Université de Franche-Comté **UFR Sciences, Techniques et Gestion de l'Industrie**

LIFC **École doctorale SPIM**

Numéro d'ordre:

Organisation et gestion de l'information dans un réseau de capteurs/actionneurs

THÈSE

sera présentée et soutenue publiquement le 3 décembre 2009

pour l'obtention du

Grade de Docteur de l'Université de Franche-Comté

École Doctorale SPIM - Spécialité Informatique

par

Kahina BOUTOUSTOUS

Composition du jury

Directeur de thèse :	Julien BOURGEOIS	Professeur à l'Université de Franche-Comté
Président :	Jean-Michel JOLION	Professeur à l'Université de Lyon
Rapporteurs :	Salvatore-Antoine TABBONE	Professeur à l'Université de Nancy
	Franck MARZANI	Professeur à l'Université de Bourgogne
Examinateurs :	Didier EL BAZ	Chargé de recherche Habilité à Diriger des Recherches au LAAS-CNRS
Encadrant :	Eugen DEDU	Maître de Conférences à l'Université de Franche-Comté

Remerciements

J'arrive à la fin de ces trois années de thèse, dont le bon déroulement et les travaux réalisés n'auraient pas pu se faire sans l'aide et le soutien de personnes que je tiens à remercier :

Tout d'abord le conseil général du Doubs pour avoir permis le financement de cette thèse.

Puis ma profonde gratitude va à Julien Bourgeois, Professeur à l'Université de Franche-Comté pour avoir accepté de me prendre en thèse, pour avoir dirigé mon travail, m'avoir judicieusement conseillée et encadrée. Ce qui m'a permis d'y voir plus clair et plus loin tout au long de mes trois années de thèse.

À Jean-Christophe Lapayre, Professeur à l'Université de Franche-Comté et Directeur du laboratoire LIFC qui a mis à ma disposition sa structure pour m'accueillir lors de ma thèse.

À François Spies, Professeur à l'Université de Franche-Comté, pour m'avoir accueillie au sein de l'équipe OMNI du laboratoire LIFC à Montbéliard.

Mes remerciements vont vers Eugen Dedu, Maître de Conférences à l'Université de Franche-Comté, pour ses précieux conseils.

Je tiens à remercier Jean-Michel Jolion, Professeur à l'Université de Lyon pour l'honneur qu'il me fait en acceptant d'être président de jury.

Je tiens à remercier Franck Marzani, Professeur à l'Université de Bourgogne et Salvatore-Antoine Tabbone, Professeur à l'Université de Nancy pour avoir accepté de rapporter ce travail, d'apporter une contribution critique à celui-ci et aussi pour toutes leurs remarques positives et constructives.

Je remercie sincèrement Didier EL Baz Chargé de recherche Habilité à Diriger des Recherches au LAAS-CNRS de Toulouse, pour avoir accepté de participer à mon jury de thèse et accepté d'examiner mon travail.

Je tiens à remercier particulièrement Laëtitia Matignon, Guillaume Laurent et Nadine Piat du laboratoire FEMTO-ST pour avoir mis à ma disposition la plate-forme Smart Surface qui était nécessaire pour tester mes algorithmes en temps réel et pour la richesse que m'a apporté le travail collaboratif avec eux.

Je tiens aussi à remercier Yves-André Chapuis, Maître de Conférences Habilité à Diriger des Recherches à l'Université de Strasbourg, pour m'avoir orientée lors de la rédaction du premier chapitre et permis de bénéficier de son expérience.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

TABLE DES FIGURES

Introduction

La déclaration "*small is beautiful*" est incontournable dans le monde de la technologie, car ce qui est petit est beau, rapide, bon marché et rentable. Ce qui explique les progrès des technologies de miniaturisation, qui ont d'importantes répercussions sur notre vie quotidienne. Les radios et les ordinateurs qui autrefois occupaient de grands espaces peuvent désormais tenir dans la paume d'une main.

La miniaturisation électronique qui a débuté au cours de la deuxième guerre mondiale, continue à changer notre monde et à engendrer une révolution dans les capteurs et les dispositifs micro-mécaniques. Les techniques de miniaturisation sont de nos jours utilisées dans divers domaines comme l'industrie, l'automobile, la médecine, la météorologie ou l'environnement. Elles permettent une meilleure mobilité et une grande souplesse dans l'utilisation des appareils.

En réalité, la miniaturisation des systèmes mécaniques a permis une convergence avec la micro-électronique. Puis s'est suivie la miniaturisation de l'ensemble. Cela a permis leur intégration sur des circuits intégrés ou des puces, permettant ainsi l'invention en 1970 des micro-systèmes avec la technologie **MEMS** (*Micro Electro Mechanical Systems*) qui associent des éléments mécaniques et électroniques.

Depuis les années 90, les MEMS ont su s'imposer dans de nombreux domaines et cela grâce à leur petite taille qui les rend moins encombrants avec de meilleures performances, une faible consommation d'énergie, une grande rapidité, une fiabilité élevée et une intégration sur des circuits intégrés.

L'arrivée des MEMS a permis de réaliser un grand pas dans la technologie de l'intégration entre la mécanique et l'électronique. De nos jours, cette technologie migre vers une nouvelle tendance qui consiste, en plus, à intégrer de l'intelligence pour évoluer vers des micro-systèmes pouvant réaliser des tâches complexes. Le problème est que cette intelligence s'implante non pas sur des systèmes conséquents mais plutôt sur des systèmes miniaturisés.

La large utilisation d'éléments de plus en plus miniaturisés a également mené au développement et à *la manipulation de micro-pièces*. Or il est difficile de manipuler des micro-pièces, étant donné qu'il n'est pas possible de les toucher directement à la main, ou avec des objets non adéquats, sans les abîmer. De plus, lors de l'utilisation, par exemple, d'une pince préhensile, il est plus facile d'attraper la micro-pièce que de la lâcher à cause des forces d'adhésion qui vont maintenir la pièce collée à la pince. Il est aussi difficile de manipuler des micro-pièces en

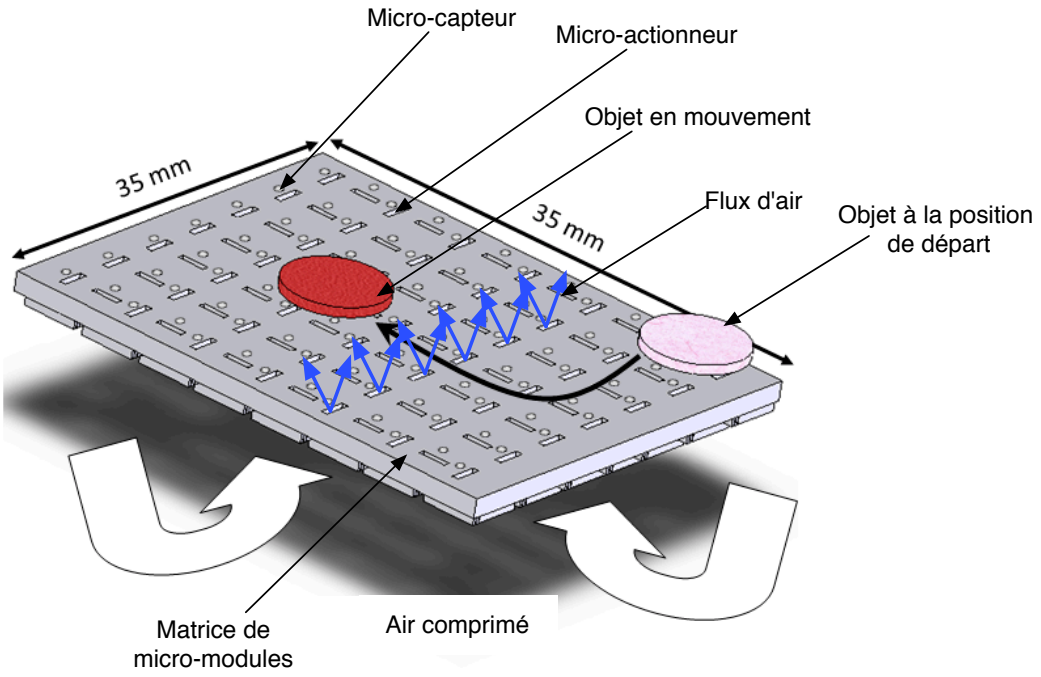


FIG. 1 – Micro-manipulateur pneumatique constitué d'une matrice de micro-modules

parallèles en utilisant des pinces préhensiles à cause du risque de collision entre les pinces.

Tout ceci a motivé les chercheurs à développer des **systèmes distribués de manipulations de micro-pièces**. De nombreux projets se sont intéressés à des systèmes MEMS permettant le convoyage de micro-pièces. Pourtant, la majorité de ces projets ont porté sur des manipulations sans retour d'information ce qui ne permet pas le tri des micro-pièces. Concrètement, le tri consiste à reconnaître et à déplacer des micro-pièces d'un point vers un autre, de manière à ce que les objets de même forme soient regroupés entre eux et donc dirigés vers le même emplacement.

D'autres projets qui prennent en compte ces retours d'information sont tributaires d'une unité de calcul externe, typiquement un PC. Cette dépendance limite l'autonomie du système de micro-convoyage et l'empêche de s'intégrer dans un système embarqué.

D'un côté, nous avons l'évolution de la technologie, apportée par les MEMS, vers une intégration de la mécanique, de l'électronique et de l'intelligence, de l'autre côté, la miniaturisation des éléments technologiques qui implique un besoin croissant de systèmes de manipulation. De là est née l'idée de concevoir un micro-manipulateur distribué et intégré sur une matrice de micro-modules intelligents (plusieurs centaines). Cette surface active permet le déplacement, le positionnement et le convoyage automatique de micro-pièces. Le convoyage est effectué **sans contact** avec la pièce grâce à un flux d'air.

Chaque micro-module sera composé d'un micro-actionneur, d'un micro-capteur et d'une unité de traitement. La coopération de ces nombreux micro-modules permettra de différencier les pièces et de commander les micro-actionneurs afin de déplacer et positionner de façon précise les micro-pièces sur la Smart Surface (voir figure 3.1).

Cette thèse intervient plus précisément dans la gestion de la coopération entre ces nombreux

micro-modules pour permettre un traitement et une gestion distribuée des informations. Elle entre dans le cadre du projet **Smart Surface** dont l'objectif est **la conception, le développement et le contrôle d'un système micro-robotique distribué pour le convoyage, le positionnement et le tri de micro-pièces à l'échelle mésoscopique (micromètre au millimètre)**.

Ce projet est financé par l'ANR[1] et regroupe plusieurs laboratoires de diverses disciplines qui travaillent en collaboration.

Une première Smart Surface a été conçue et fabriquée à *l'Institute of Industrial Science(IIS)* de l'Université de Tokyo au sein du laboratoire du *Pr. Hiroyuki Fujita* en collaboration avec Yves-André Chapuis, chercheur au laboratoire LIMMS/CNRS-IIS[2].

La Smart Surface a pour but le tri dans le cadre du micro-assemblage automatisé de micro-objets. Cela constitue un challenge important dû aux dimensions de ces pièces. Afin de bien comprendre le contexte étudié dans cette thèse, nous commencerons d'abord avec un simple exemple d'un jeu d'enfants, qui consiste à trier des pièces suivant leur géométrie. Le jeu est composé d'un ensemble de pièces (par exemple : carré, cercle, triangle) et d'une pyramide. Chaque face de la pyramide contient des trous en fonction des différentes géométries des pièces. Le but du jeu est de trier les pièces selon leur géométrie. Pour trier ces pièces il faut :

1. reconnaître la pièce ;
2. déplacer la pièce.

L'enfant doit alors reconnaître la pièce et la déplacer au trou correspondant. Mais en réalité reconnaître la pièce est un terme bien trop fort pour un enfant de cet âge qui ne peut pas reconnaître un cercle ou un carré. Il a juste la faculté de faire la différence entre les différentes pièces. On peut alors tout simplement parler de **différenciation** qui est le thème principal abordé dans cette thèse.

Nous nous sommes intéressés à la **différenciation de micro-pièces**. En d'autres termes nous essayons de répondre à la question suivante : Est-il possible de différencier une micro-pièce posée sur la Smart Surface ? Revenons à notre exemple. Pour passer la pièce dans le trou qui lui correspond, l'enfant va comparer la pièce avec les différents trous, en se basant sur la géométrie de la pièce. La géométrie d'une pièce est une caractéristique propre à la pièce. Dans ce document cette caractéristique est appelée **critère** de la pièce. Afin de différencier les pièces posées sur la Smart Surface il est nécessaire de déterminer des critères qui nous permettront de différencier les pièces. Pour cela, il nous a fallu répondre aux questions suivantes : quels sont les critères de différenciation qui permettent de différencier les pièces ? Plus encore, pour un ensemble de pièces générées exhaustivement est-il possible d'arriver à une différenciation totale ? C'est-à-dire que quelque soit le groupe de pièces, parmi un ensemble de pièces, arrive-t-on toujours à les différencier ? Dans notre exemple du jeu d'enfants, il est logique que pour trier les pièces l'enfant doit d'abord les ramasser, mais avant de les ramasser l'enfant doit les localiser. Localiser des pièces est une tâche relativement facile. Par contre cela est moins évident pour des micro-pièces. Quand une micro-pièce est posée sur la Smart Surface, elle doit être détectée. La détection est effectuée par des capteurs qui se trouvent sur la Smart Surface. Ces capteurs sont binaires et chaque micro-pièce est décrite par une matrice binaire qui correspond à sa représentation

[1] Agence nationale de la recherche (ANR-06-0009-03) et le conseil général du Doubs.
[2] Institute of Industrial Sciences.

binaire. Sur cette représentation binaire seront calculées les différentes valeurs des critères de différenciation. Nous nous sommes alors posés la question suivante : combien de capteurs faut-il mettre dans la Smart Surface afin d'atteindre son fonctionnement optimal ?

Le projet Smart Surface nécessite la collaboration de plusieurs sous-projets de différents domaines de compétence interdépendants. En effet, il est plus raisonnable de déterminer le paramétrage optimal de la Smart Surface avant de commencer à la réaliser. Par exemple, déterminer le nombre de capteurs nécessaires est une question essentielle pour arriver à bien différencier les pièces d'une part et d'autre part c'est une donnée cruciale pour la construction de la Smart Surface finale.

Plan du document

Ce document se compose de trois parties : Les chapitres 1 et 2 de la première partie traitent respectivement de l'état de l'art des systèmes de manipulation et des méthodes de différenciations. La deuxième partie contient trois chapitres. Le chapitre 3 présente le projet Smart Surface et la problématique traitée dans cette thèse. Les chapitres 4 et 5 sont quant à eux dédiés à la présentation des solutions proposées afin de répondre aux questions posées dans le chapitre 3. Enfin la troisième partie, composée des chapitres 6 et 7, est consacrée à la présentation des différentes expérimentations réalisées.

Le chapitre 1 est d'abord consacré à la définition de la manipulation des objets et plus particulièrement des micro-pièces. Plusieurs problèmes liés aux micro-pièces seront également présentés. Ensuite nous parlerons des MEMS (*Micro Electro Mechanical Systems*) technologie incontournable et prometteuse dans le monde de l'industrie. Nous terminerons ce chapitre avec un état de l'art et une synthèse des différents systèmes de manipulation.

Le chapitre 2 présente un panorama des différentes méthodes de différenciation qui est le but principal cette thèse. Une classification de ces différentes méthodes sera donnée dans ce chapitre.

Le chapitre 3 est consacré au projet Smart Surface, il présente la problématique traitée dans cette thèse. Cette dernière consiste à déterminer les paramètres de la Smart Surface essentiels à son bon fonctionnement.

Le chapitre 4 est consacré à la présentation de l'approche utilisée pour déterminer le premier paramètre de la Smart Surface. Il consiste à trouver les critères de différenciation qui arrivent à différencier tous les groupes de pièces. Ces groupes sont formés à partir d'un ensemble de pièces générées exhaustivement.

Le chapitre 5 est dédié à une autre approche permettant de trouver la taille de la grille de capteurs à intégrer dans la Smart Surface finale. Cette approche, contrairement à la précédente utilise un ensemble fini de pièces appelées *modèles de pièces*.

Le chapitre 6 est une présentation des résultats obtenus grâce à notre outil ECO (*Exhaustive Comparison Framework*) qui est une application de l'approche utilisée pour trouver le premier paramètre de la Smart Surface à savoir trouver les meilleurs critères à utiliser pour différencier les pièces. Les résultats obtenus avec cet outil nous ont permis de valider notre approche.

Le chapitre 7 est une présentation des résultats obtenus avec notre outil SNC (*Sensor Network Calibrator*). L'outil SNC est une application de l'approche proposée pour déterminer le

deuxième paramètre de la Smart Surface qui est la taille de la grille de capteurs à poser sur la Smart Surface. Les résultats obtenus montrent l'intérêt de cet outil pour construire une Smart Surface optimale.

Première partie

L'état de l'art

CHAPITRE 1

Les applications de la Smart Surface

Sommaire

1.1 Introduction

De nos jours, nous manipulons des objets de taille très réduite mais qui réalisent des tâches de plus en plus complexes. On peut citer comme exemple le téléphone portable utilisé pour la communication mais aussi pour naviguer sur le Net, consulter les mails, écouter de la musique ou encore voir des vidéos. La miniaturisation des objets a été accélérée grâce à l'avènement des MEMS. Ces derniers mesurent moins de cent millièmes de millimètres et rendent des services irremplaçables dans plusieurs domaines tels que l'automobile, la médecine, etc.

La miniaturisation des composants électroniques de l'ordre de quelques microns a rendu ces derniers impossibles à manipuler directement par les humains. En effet, un composant de quelques microns peut être facilement détérioré si l'on essaye de le tenir dans une main ou avec

des éléments non adaptés d'où la nécessité de trouver des systèmes capables de manipuler des objets de tailles misoscopiques.

De nombreux travaux se sont intéressés aux systèmes MEMS utilisés pour la manipulation de micro-pièces. La plupart de ces systèmes n'ont pas de retour d'information ce qui ne permet pas le tri des micro-pièces. D'autres travaux prennent en compte ces retours d'information mais sont tributaires d'une unité de calcul externe, typiquement un PC. Cette dépendance limite l'autonomie du système de micro-convoyage et l'empêche de s'intégrer dans un système embarqué. Les Smart Surfaces sont des systèmes dotés et composés de micro-actionneurs, de micro-capteurs et d'unités de traitement qui collaborent dans le but de transporter, de positionner et d'orienter des objets dont la taille est de l'ordre du micron. Les Smart Surfaces sont des systèmes de manipulation de micro-pièces dotés de la technologie MEMS.

Dans ce qui suit, nous définirons la manipulation de pièces, donnerons des exemples sur les macro-manipulateurs, expliquerons les différents problèmes liés à la manipulation des micro-pièces et présenterons la définition des MEMS. Enfin il s'en suivra des exemples de systèmes de micro-maniplulations distribués 2D, leur classification en deux catégories à contact et sans contact avec une étude comparative de ces derniers.

1.2 Les macro-manipulateurs de pièces

Dans le monde de l'industrie, il est indispensable de manipuler des pièces que ce soit dans une chaîne de montage ou dans une chaîne de fabrication. La manipulation de pièces se définit tout simplement comme un ensemble d'actions qui ont pour but de modifier la position ou l'orientation de pièces. Ces actions consistent à transporter, positionner ou orienter les pièces. Si en plus, on rajoute une contrainte supplémentaire à la manipulation de pièces, à savoir, selon certaines données (le poids, la couleur ou la forme, etc.) déplacer la pièce à une position ou orienter la pièce à une direction bien spécifique nous parlons alors d'une manipulation dans le but d'un repositionnement suivant certaines données. C'est la définition même du tri, qui consiste à manipuler des pièces dans le but de les trier selon certains critères.

Généralement, en industrie le nombre de pièces à manipuler est tellement grand qu'on utilise un système de manipulation des pièces. Un système de manipulation de pièces peut être une surface qui a pour fonction le contrôle des déplacements (translation, rotation) des pièces. Les plus simples de ces systèmes de manipulation sont les fonctions d'alimentation des pièces (*part feeding*). Elles consistent à déplacer une ou plusieurs pièces en groupe, et à les positionner en un point donné [22]. La fonction d'alimentation des pièces est importante dans les applications où il est nécessaire d'inspecter les pièces pour, par exemple, rejeter les pièces défectueuses. Une autre caractéristique importante, généralement associée à la fonction d'alimentation des pièces, est la position d'une pièce : une pièce qui est transférée du point A vers un autre point B, doit arriver au point B dans une position connue, afin de la préparer à la prochaine étape de fabrication. Cela s'apparente à un coureur de relais qui doit remettre son bâton à l'autre coureur, le bâton doit être dans une certaine orientation. Cela conduit à la notion de manipulation. La manipulation est un type sélectif de la part feeding : des pièces spécifiques doivent exécuter un mouvement spécifique, par exemple, lors du convoyage d'un groupe de pièces, les pièces défectueuses doivent être éjectées ou bien orientées dans une direction bien spécifique.

Une solution typique de manipulation est attrape-et-place (*pick-and-place*) : par exemple un robot avec des pinces en guise de bras, qui ramasse une pièce d'un lieu de départ, la transporte et la place vers une destination [22]. L'inconvénient de cette méthode est qu'elle est coûteuse, difficile à concevoir et à contrôler. Si plusieurs objets doivent être manipulés simultanément, cette opération est appelée *manipulation parallèle*, il est essentiel de prendre en compte la notion de contrôle. On doit tenir compte du potentiel risque de collisions entre les différentes pinces. Les problèmes de la pick-and-place ont motivé les chercheurs dans la mise au point d'un dispositif de manipulations non préhensile regroupé dans un champ appelé la manipulation distribuée [22]. Des dispositifs constitués de 100 à 1000 actionneurs répartis sur une surface ont été proposés. Chaque actionneur produit une force locale. Cette force peut être de type magnétique, mécanique, etc. Par conséquence, une rangée d'actionneurs peut produire un champ de force plus important pour agir sur une pièce qui est sur la surface. Cette manipulation distribuée permet aussi la manipulation de plusieurs pièces parallèlement. Cette conception élimine l'encombrement causé par les bras du robot, le contrôle est simplifié ainsi que la tâche d'inspecter les pièces (étant donné qu'elles sont visibles). Nous reviendrons plus en détail sur les systèmes distribués de manipulation de pièces dans la section 1.5.

Un exemple de système de manipulation distribuée est le *bol vibrant* [23, 24]. Très utilisé, il consiste à mettre des objets dans un bol qui vibre en tournant, le mouvement du bol fait monter les pièces vers le haut suivant une certaine trajectoire. Cette trajectoire contient des coupures considérées comme un filtre mécanique. Les pièces qui ont une certaine orientation vont passer le filtre, les autres vont retomber dans le bol. L'opération est réitérée pour les pièces restantes.

Nous citerons aussi les surfaces vibrantes comme le Sony's APOS fonction d'alimentation des pièces *part feeder* [23, 24], qui utilise une série de nids (pièges de silhouette) qui sont dans une plaque vibrante. Les nids et le mouvement vibratoire sont conçus de telle sorte que les pièces qui ont une orientation particulière restent dans les nids et les autres pièces traversent les nids éventuellement remplis avec des pièces dans la même orientation. Bien que le mouvement vibratoire soit sous le contrôle logiciel, les nids doivent être conçus pour chaque pièce.

Les raisons du succès et de la large utilisation des bols vibrants et des surfaces vibrantes tells que le Sony APOS part feeder résident dans leur simplicité et dans le fait qu'ils n'utilisent pas de capteurs, cependant ils ont quelques inconvénients :

- la pièce peut se coincer dans le filtre ;
- ils doivent être revus lorsque la géométrie de la pièce change ;
- ils peuvent causer des dommages sur les pièces quand elles tombent dans le bol ou les user par les vibrations.

Les travaux de Bohringer [23, 1] ont présenté un dispositif qui, grâce à un champ de force, positionne et aligne des pièces. Ce champ est créé par les vibrations d'une plaque. Le dispositif se compose d'une plaque d'aluminium fixée à un générateur de vibration capable de produire une force de 500 n (voir figure 1.1). Un signal en entrée permet de préciser l'oscillation souhaitée. Le mouvement est similaire aux vibrations transversales d'une plaque rectangulaire, fixée d'un côté et libre de l'autre. Ce mouvement vibratoire crée un champ de force qui va attirer les pièces vers des positions dont la vibration est minimale, appelées ligne nodale. En changeant les deux champs de force, l'état d'équilibre d'une pièce peut être réduit successivement jusqu'à atteindre l'état final souhaité. La figure 1.1 montre deux pièces de la forme d'un triangle et d'un trapèze,

après avoir atteint leur position d'équilibre. Afin de mieux illustrer l'effet de l'orientation, les auteurs ont tracé une courbe montrant la ligne nodale.

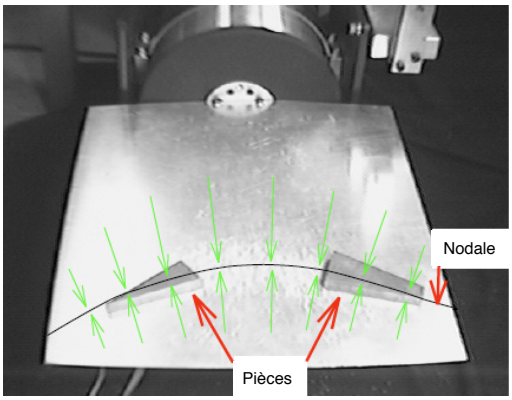


FIG. 1.1 – Exemple de champ de force d'une plaque vibrante en aluminium de taille 50 × 40 cm^2 [1].

Les *pinces mâchoires* ou *pinces préhensiles* sont obtenues avec une matrice d'actionneurs en créant un champ de force opposé qui peut orienter et déplacer la pièce [25]. Bohringer et al. [23, 2, 24, 26] ont proposé un concept appelé *champs de force programmable* qui est une extension de l'algorithme de Goldberg. L'idée de base est de réaliser une surface plane et une fois que la pièce est en contact avec la surface plane, elle est soumise à un champ de force qui va déplacer la pièce jusqu'à atteindre une position d'équilibre stable (voir figure 1.2).

Il existe dans la littérature différents systèmes de manipulation, dans le cadre de cette thèse nous nous sommes particulièrement intéressés à la manipulation des micro-pièces

1.3 Les micro-pièces

1.3.1 Définition d'une micro-pièce

De nos jours, les nouvelles technologies ne cessent de révolutionner notre monde. Parmi ces technologies, nous pouvons citer la micro-technologie, qui a su s'imposer dans notre vie quotidienne. Les micro-produits sont présents dans de nombreux domaines tels que le domaine médical, celui du son et celui des images. Leur présence est visible dans la plupart de nos objets quotidiens tels que les appareils photo numériques, cartes bancaires ou micro-ordinateurs. Mais qu'est ce qu'une micro-pièce ?

Etymologiquement, le mot *micro* se définit comme un préfixe dont l'origine greque signifie *petit*. Une micro-pièce se définit par un objet dont les dimensions qui le caractérisent se mesurent à l'échelle du micron[3].

[3] 1 micron=10^{-6} mètres.

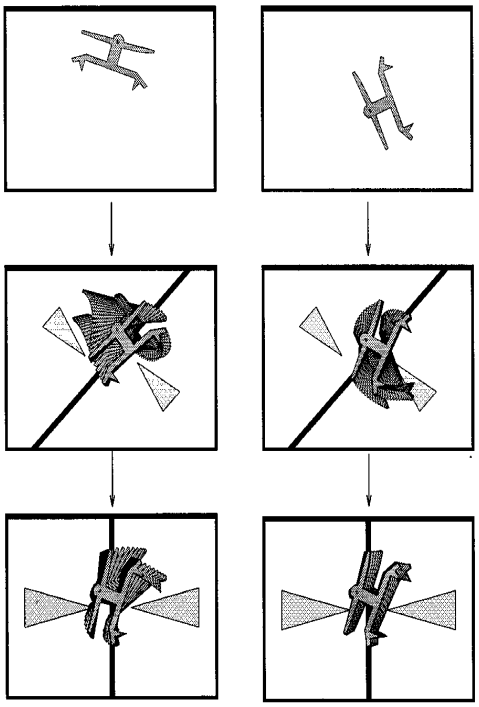


FIG. 1.2 – Orientation d'une pièce sans capteur en utilisant une séquence de champ de force, les flèches indiquent le sens du champ de force [2].

1.3.2 Problèmes spécifiques aux micro-pièces

Développer des systèmes de manipulation des micro-pièces est loin d'être une tâche facile car à cette échelle les micro-pièces sont soumises à différents phénomènes d'interaction entre le manipulateur et la micro-pièce [27, 28, 29]. À cette échelle, l'effet de la surface a un rôle plus important que le volume, certaines lois physiques sont plus influentes que d'autres.

Pour des pièces avec des masses de plusieurs grammes, les forces gravitationnelles sont généralement plus dominantes que les forces d'adhésion. Donc quand ces pièces sont attrapées par une pince, elles vont tomber dès que cette dernière est ouverte. Par contre pour des pièces avec une taille inférieure au millimètre (une masse inférieure à $10^{-}16\ kg$), les forces gravitationnelle et inertielle vont être négligeables en comparaison aux forces d'adhésion qui sont généralement proportionnelles à la surface. Donc lors de la manipulation de ces micro-pièces avec par exemple une pince, les forces d'adhésion vont maintenir la micro-pièce collée à la pince.

Exemple : comme exemple d'application numérique, on choisira une sphère en silicium avec un rayon $r = 1\ \mu m$. Les forces électrostatiques générées du fait de la proximité ou du contact de la micro-pièce et du micro-manipulateur sont définies par l'équation 1.1 :

$$F_E = \frac{\pi\rho^2(r)^2}{\varepsilon} \tag{1.1}$$

où :
ρ : la densité surfacique de charge d'une sphère.

ε : la permittivité électrique de l'air.
r : le rayon de la micro-pièce.
Pour l'application numérique on prendra : $\varepsilon = 8.854 \times 10^{-12}\ F.m^{-1}$ et $\rho = 1.6 \times 10^{-6}\ C.m^{-2}$.
On obtient :
$F_E = 9.2\ 10^{-10}\ mn$.
Les forces atomiques appelées forces de Van der Waals sont définies par l'équation 1.2 :

$$F_{vdw} = \frac{hr}{8\pi z^2} \tag{1.2}$$

où :
h : la constante de Lifshitz-Van der Waals.
z : la distance entre les atomes des éléments.
r : le rayon de la micro-pièce.
Pour l'application numérique on prendra : $h = 2.5\ eV$ et $z = 1\ nm$. On obtient :
$F_{vdw} = 1.59\ 10^{-5}\ mn$.
Une tension, définit par l'équation 1.3, est créée entre les surfaces des éléments en contact suivant le degré d'humidité des micro-objets.

$$F_T = 4\pi r\gamma \tag{1.3}$$

où :
γ : la tension de la surface.
r : le rayon de la micro-pièce.
Pour l'application numérique on prendra : $\gamma = 73\ mn.m^{-1}$. On obtient :
$F_T = 9,16\ 10^{-4}\ mn$.
Dans l'environnement tous les objets sont soumis à la loi de gravité qui, pour une sphère, se définit par l'équation 1.4 :

$$F_g = \frac{4}{3}\pi r^3 \rho g \tag{1.4}$$

où :
r : le rayon de la sphère.
ρ : la masse volumique.
g : constante gravitationnelle.
Pour l'application numérique on prendra : $\rho = 2,3 \times 10^3\ kg.m^{-3}$ (masse volumique du silicium).
On obtient :
$F_g = 9,6\ 10^{-11}\ mn$.

En classant les résultats des différentes forces on obtient :
$F_g \ll F_E \ll F_{vdw} \ll F_T$

On remarque sur cet exemple que pour une sphère de rayon 1 μm la force de gravité devient plus faible. Dans [27] est présentée une étude de l'amplitude des forces électrostatiques, forces de Van Der Waals, forces de tentions de surface et forces gravitationnelles en fonction de la taille de la micro-sphère. On remarque que, pour des sphères de rayon inférieur à 10 μ, la force de gravité est alors la plus faible.

Dans la micro-manipulation, les forces de gravité sont négligeables par rapport aux forces électrostatiques, aux forces de Van Der Waals et aux tentions de surface (elles sont plus influentes

que les forces de gravité). Ces dernières vont augmenter la force d'adhésion lors d'un contact avec la micro-pièce ce qui va causer par exemple des difficultés à reposer la micro-pièce après la manipulation (pour plus de détails voir [30]).

Les micro-pièces sont aussi très fragiles à manipuler et peuvent facilement s'abîmer. Elles ne peuvent donc pas être manipulées par de grands dispositifs préhensiles pouvant générer des forces plus importantes que la résistance mécanique des micros-pièces.

Nous allons voir plus loin que ces pièces, par leur petite taille, posent aussi des défis dans leur reconnaissance et c'est là que se trouvent certaines de nos contributions.

Face aux problèmes spécifiques des micro-pièces, il est important de concevoir des systèmes de manipulation de micro-pièces bien adaptés.

1.4 Les systèmes de micro-manipulation MEMS

1.4.1 Définition des MEMS

C'est vers les années 70 que la technologie MEMS a été développée. Plus connue sous le nom de *Micro Electro Mechanical Systems*, les micro-systèmes sont des *systèmes très petits* ou bien *fabriqués à partir de très petits composants* pouvant réaliser une tâche de détection (grâce à des micro-capteurs) ou bien d'agir (grâce à des micro-actionneurs). Leur taille est comprise entre 1 μm et 1 mm [3] comme le montre la figure 1.3.

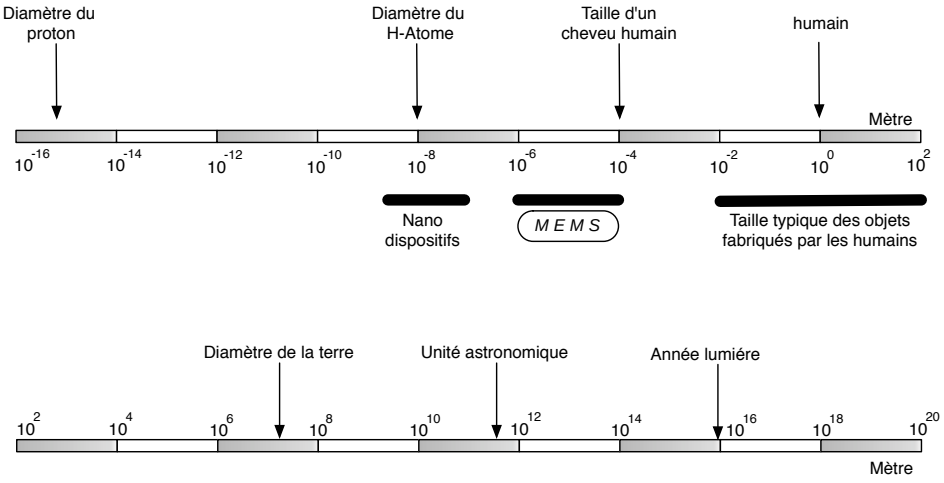


FIG. 1.3 – Échelle de comparaison des ordres de grandeur des objets [3].

1.4.2 Structure générale des MEMS

Leur nom n'a aucune relation avec la méthode spécifique de leur fabrication ou leur type particulier de fonctionnement. Mais du terme Micro Electro Mechanical Systems (MEMS) on peut déduire les significations suivantes [31] :

- **Micro :** donne une information sur leur dimension ;
- **Electro :** signifie l'implication soit de l'électricité, soit de l'électronique ou bien des deux ;

- Mechanical : suggère un déplacement de pièces mécaniques sous l'effet d'un signal électrique.

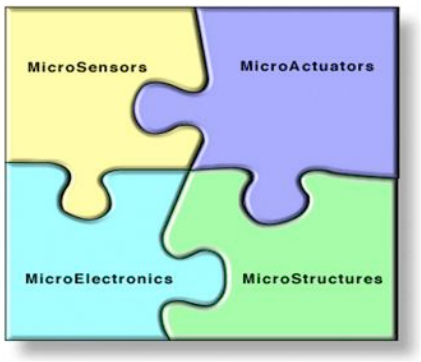


FIG. 1.4 – Présentation de l'ensemble des éléments constituants un MEMS.

Les MEMS sont donc une intégration de micro-capteurs (*MicroSensors*), de micro-électroniques (*MicroElectronics*), de micro-actionneurs (*MicroActuators*) et de micro-structures (*MicroStructures*) sur une couche de silicium grâce à la technologie de micro-fabrication (voir figure 1.4[4]).

Les micro-capteurs sont considérés comme les sens du système MEMS. Ils ont comme rôle de détecter et de recueillir les informations sur l'environnement par le biais de mesures mécaniques, thermiques, biologiques, chimiques, optiques et de phénomènes magnétiques.

Les informations récoltées par les micro-capteurs sont alors transmises aux éléments micro-électroniques. Les éléments micro-électroniques sont considérés comme le cerveau du système MEMS. Ils ont la capacité de prendre des décisions, de diriger et d'agir. Les éléments micro-électroniques vont traiter les données reçues par les micro-capteurs et par leurs capacités de prise de décision vont agir sur les micro-actionneurs [32].

Les micro-actionneurs vont réagir par le déplacement, le positionnement ou bien l'activation d'un dispositif extérieur.

Grâce à la micro-fabrication, la construction de tous petits composants sur une couche de silicium est possible. Ces petits composants sont appelés micro-structures.

1.4.3 Application des MEMS

Les MEMS sont des micro-systèmes électromécaniques composés de capteurs pouvant identifier des paramètres physiques de leur environnement (pression, accélération, température etc) et/ou des actionneurs pouvant agir sur cet environnement. Plus encore, le MEMS est extrêmement présent dans la fabrication de micro-systèmes complexes et diversifiés : pompes, moteurs, etc. Cette présence a permis leur large commercialisation dans divers domaines allant de l'habitation, de la surveillance de la pression sanguine à des systèmes de suspension active pour les automobiles, etc. Cette commercialisation, qui a débuté vers les années 90, et la diversité de leurs domaines d'application ont engendré une activité économique considérable de l'ordre de plusieurs millions de dollars (voir figure 1.5[5]).

[4] `http://faculty.bus.olemiss.edu/breithel/b620s02/riley/MEM.jpg`
[5] `http://www.isuppli.com/News/Pages/Global-MEMS-Market-Suffers-First-Ever-Decline-in-2008.aspx`

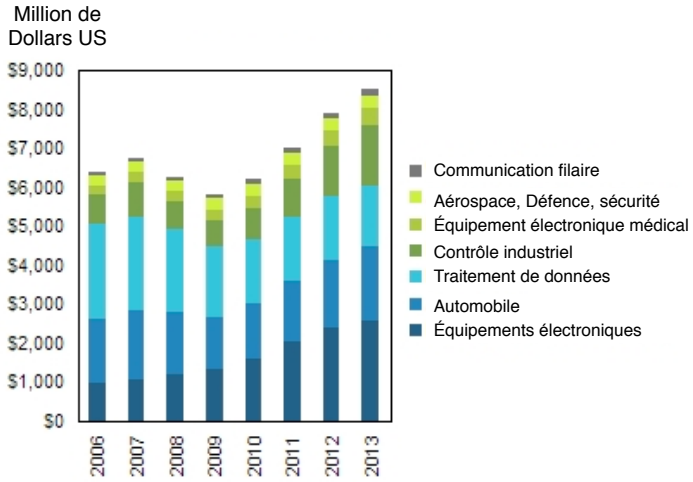


FIG. 1.5 – Marché global en millions de dollars selon divers domaines d'applications des MEMS au cours de ces dernières années.

Il existe de nombreuses applications des MEMS. Un exemple est le vidéo projecteur DMD *Digital Micromirror Device*, développé par Texas Instruments, à base d'une matrice de 800×600 de miroirs élémentaires disposés sur une plaque de silicium formant ainsi un système de projection [33].

Il y a aussi les MEMS pour airbags, dont l'unité de détection comprend deux micro-peignes imbriqués l'un dans l'autre, l'un fixe l'autre mobile. Dans le cas d'un choc important, une variation brutale de capacité apparaît. L'élément de détection, l'électronique de traitement et de déclenchement du coussin sont intégrés sur une puce de quelques millimètres carrés, produite industriellement à faible coût [34].

Les MEMS sont utilisés dans divers domaines tels que l'informatique avec les cartouches d'imprimantes à jet d'encre, l'automobile, la communication sans fil, la surveillance de la santé, la navigation personnelle et la surveillance de l'environnement [35]. La figure 1.6[6] montre quelques exemples des domaines d'applications des MEMS.

Les progrès des MEMS au cours des dernières décennies ont entraîné une expansion rapide des techniques pour construire des systèmes pour la manipulation et l'analyse de très petites pièces ainsi que pour des applications spécifiques des capteurs et des actionneurs. Ils ont également contribué au développement des systèmes de micro-manipulation.

1.5 Les systèmes de micro-manipulation distribués 2D

Un micro-actionneur ne peut agir qu'avec des mouvements simples et une force réduite. Il est donc évident qu'il ne peut pas accomplir des tâches complexes. Par contre, coordonner le mouvement simple de plusieurs micro-actionneurs permet d'accomplir une tâche complexe. Cette coordination se définit en répartissant une tâche complexe en plusieurs petites tâches simples

[6] `http://www.alaide.com/dico.php`

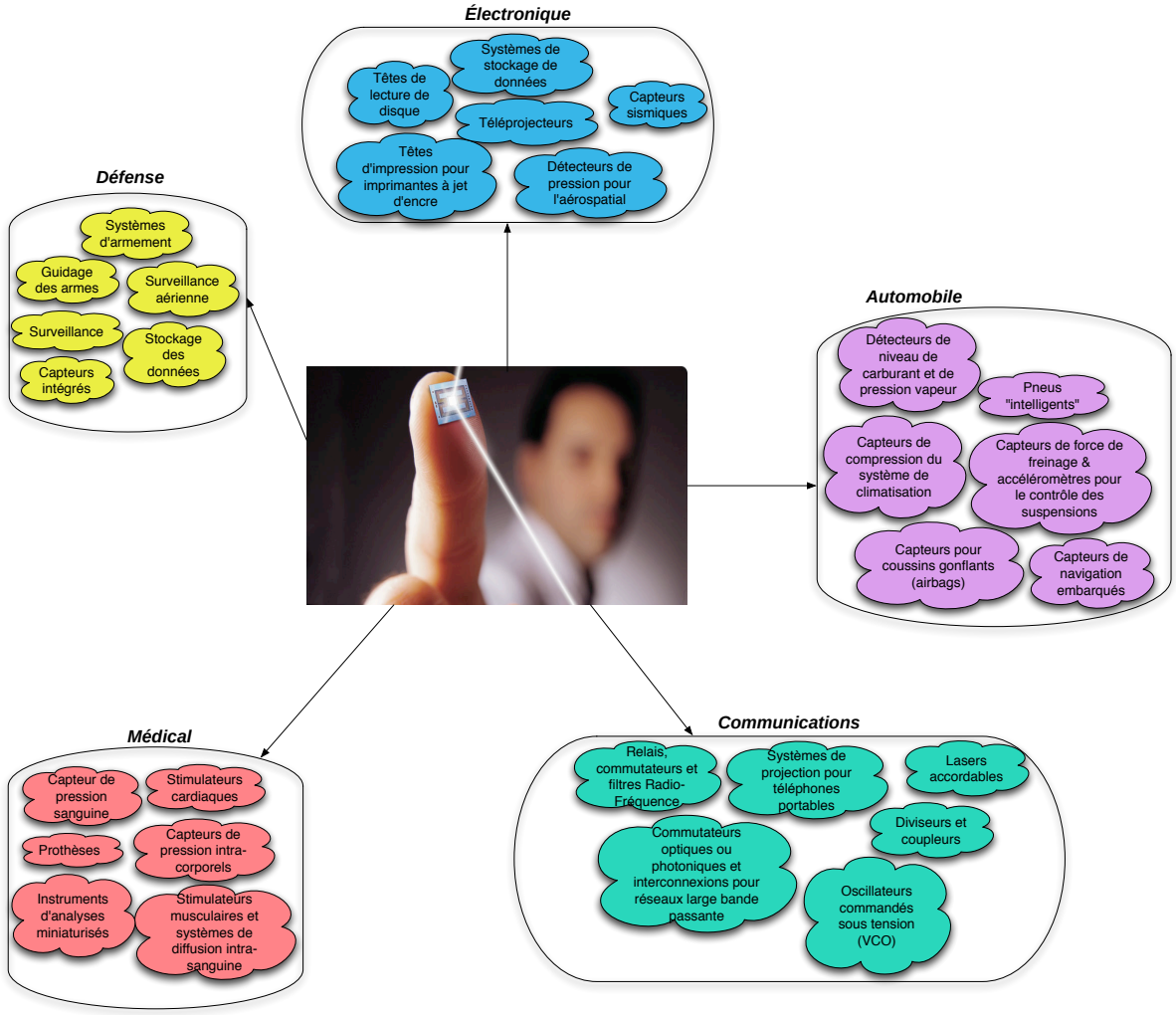


FIG. 1.6 – Exemple de domaines d'application des MEMS.

attribuées à chaque actionneur [12]. C'est le principe même des micro-manipulateurs distribués. Les systèmes de manipulation peuvent être classés suivant trois caractéristiques (voir figure 1.7) :

1. le type de pièces qu'ils manipulent :

   -micro-pièces : ce sont des pièces à l'échelle micro ;

   -macro-pièces : ce sont des pièces à l'échelle macro[7].

2. le type de manipulateurs qu'ils utilisent :

   -micro-manipulateur : sont des actionneurs à l'échelle micro ;

   -macro-manipulateur : sont des actionneurs à l'échelle macro.

3. le type de manipulation qu'ils utilisent [36] :

   -à contact : s'il y a un contact entre l'objet et l'actionneur ;

   -sans contact : s'il n'y a pas de contact entre l'objet et l'actionneur.

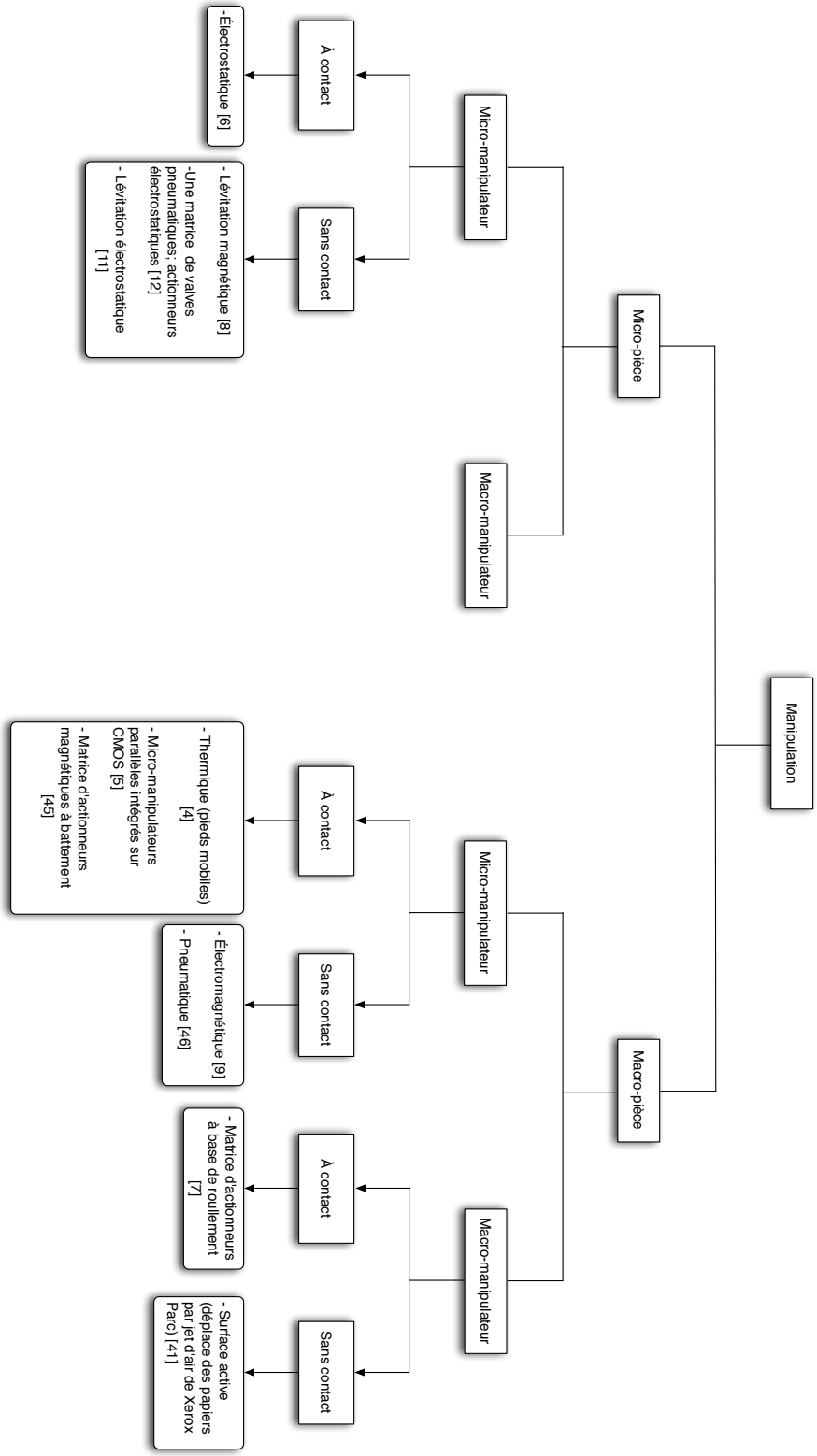


FIG. 1.7 – Schéma de classification des micro-manipulateurs.

Dans le cadre de ce travail de thèse, nous traitons la manipulation des micro-pièces grâce à un micro-manipulateur sans contact. Dans ce qui suit, nous allons détailler les systèmes de manipulation en nous basant sur le troisième critère, à savoir les systèmes de manipulation à contact et sans contact.

1.5.1 Les systèmes de manipulation à contact

Les micro-manipulateurs à contact se définissent comme un réseau de manipulateurs qui sont en contact direct avec l'objet à manipuler.

Fujita et al [4] ont développé un prototype de micro-manipulateurs thermiques à l'institut des sciences industrielles de l'université de Tokyo [4]. C'est une matrice constituée de plusieurs rangs de bimorphes thermiques (voir figure 1.8). Ces actionneurs sous effet thermique reproduisent le mouvement des cils des organismes vivants. La matrice est constituée de 512 bimorphes sur une surface de 1cm^2.

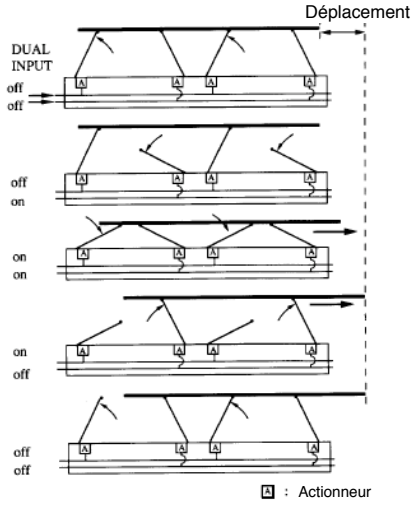


FIG. 1.8 – Image d'une cellule de la matrice de micro-manipulateurs thermiques [4].

Les actionneurs ont une longueur de 500 μm, une largeur de 100 μm, une épaisseur de 6 μm, une amplitude selon la verticale de 150 μm et une amplitude selon l'horizontale de 80 μm. Ce système permet le convoyage de petites pièces en silicium de quelques millimètres de surface avec un poids d'environ 2,4 mg avec une puissance en entrée de 4 mW pour chaque actionneur.

Bohinger et al [5, 37, 38] ont proposé la première matrice de micro-actionneurs ciliaires intégrée sur un circuit CMOS. Cette réalisation a vu le jour à l'université de Stanford et de Washington. Cette matrice est composée de quatre puces dont chacune contient une matrice de 8 cellules permettant de combiner 256 cellules de forces distribuées sur une surface de 16 mm^2 (voir figure 1.9(a)). Chaque cellule est composée de quatre actionneurs disposés de manière orthogonale dans le but de converger vers le centre de la cellule. Chaque actionneur est de forme triangulaire (voir figure 1.9(b)). Au repos ces actionneurs subissent une déflexion, et sous l'effet du courant électrique, ils vont chauffer et fléchir en direction de l'objet.

[7]Supérieur au millimètre.

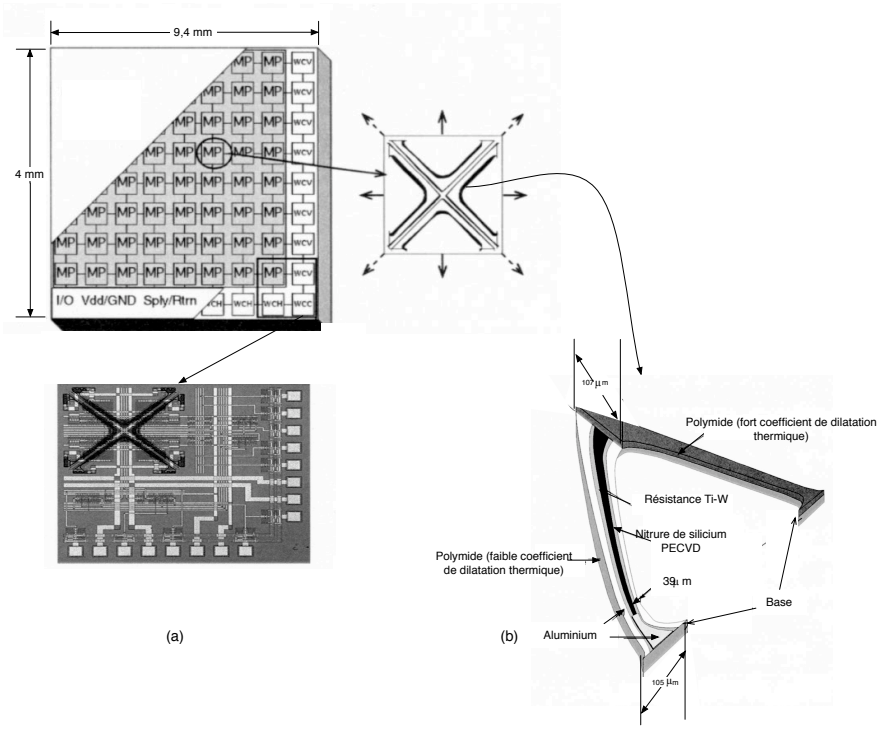


FIG. 1.9 – (a) Architecture générale d'un micro-actionneur ciliaire intégré sur un circuit CMOS. (b) Illustration d'un micro-actionneur ciliaire élémentaire [5].

Cette surface a donné de bons résultats lors des tests pour convoyer des objets en silicium ayant une épaisseur de 100 μm nécessitant une alimentation électrique entre $3,5V$et $6V$. Mais pour des objets d'épaisseur de quelques centaines de microns des problèmes de convoyage sont apparus.

Dans [6, 39] les auteurs ont proposé une matrice de micro-actionneurs électrostatiques gravés dans du silicium monocristallin. Ce système génère un champ de force dont le but est de déplacer des petits objets. Le calcul et l'expérimentation ont montré que la matrice d'actionneurs est assez puissante pour déplacer de petits objets plats.

Chaque actionneur peut approximativement générer une force égale à 10 μN et un déplacement de 5 μm. Cette surface a été construite pour atteindre une taille de 10 cm^2 avec $1,5 \times 10^5$ actionneurs (voir figure 1.10). Des essais pour déplacer de petits morceaux de verre et de petits bouts de papier ont été effectués sur ce dispositif mais sans résultat satisfaisant à cause du manque de force de frottement entre l'objet et l'actionneur.

Dans [7, 40] les auteurs ont développé un système de manipulation contenant un ensemble de 20 cellules capables de transporter des objets. Chaque cellule consiste en une paire orientée de roues (*orthogonale motorized roller wheels*), capable de produire une force pour déplacer les objets. Chacune de ces cellules est connectée à une large plaque. L'ensemble constitue une matrice de micro-manipulateurs (voir figure 1.11(a)). En coordonnant les actionneurs des cellules, l'objet positionné au-dessus des cellules peut être transporté et manipulé. En substance, ce système est une amélioration par rapport au système de convoyage traditionnel. Les objets peuvent être

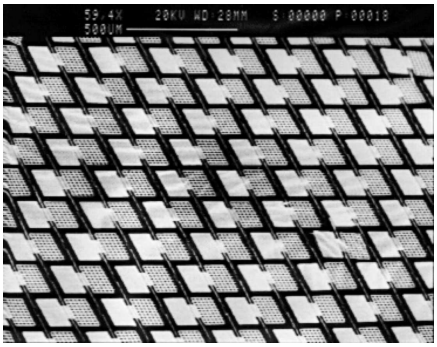


FIG. 1.10 – Une matrice d'actionneurs unidirectionnels [6].

réorientés et déplacés. Un tel système peut manipuler et transporter plusieurs objets en même temps. Le système est coordonné de manière distribuée où chaque cellule possède son propre contrôleur et chaque contrôleur communique avec ses voisins (voir figure 1.11(b)).

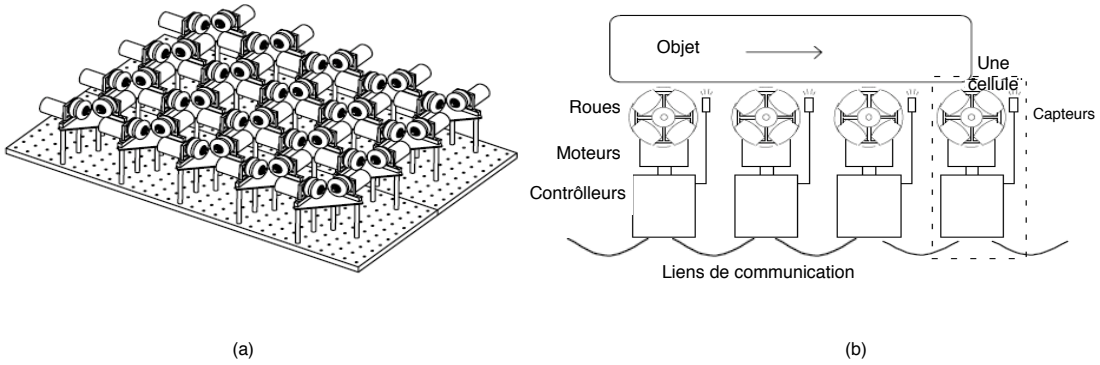


FIG. 1.11 – (a) Image d'une matrice de cellules à roulement. (b) Exemple du principe de manipulation à base de roulement pour le convoyage d'une pièce [7].

Les micro-manipulateurs à contact sont très performants pour la prise en charge d'objets assez lourds. Ils permettent de réaliser un positionnement précis. Leur fabrication est à faible coût mais les objets se déplacent à une faible vitesse.

1.5.2 Les systèmes de manipulation sans contact

Les systèmes sans contact se définissent comme un réseau de manipulateurs (pneumatique, électrostatique, électromagnétique, etc) en contact indirect avec l'objet, évitant ainsi tout contact, frottement mécanique et risque d'usure de l'objet à manipuler.

Fujita et al [8] ont proposé un système de micro-manipulateurs à lévitation magnétique. Le micro-manipulateur se compose de trois parties : Supraconducteur, électrodes et élément mobile intégrant quatre aimants (voir figure 1.12).

Ce dispositif a donné un convoyage rapide (la vitesse est de $2,8 \times 10^2$ $\mu m/s$ sur une distance de 2.8 mm) avec un positionnement précis.

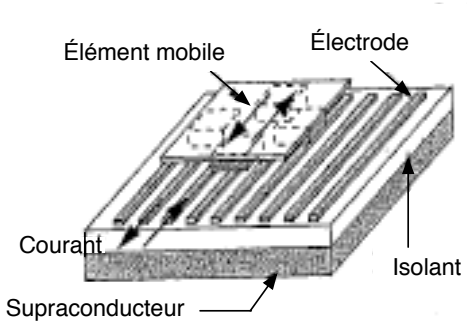


FIG. 1.12 – Une matrice d'actionneurs à lévitation magnétique [8].

Les auteurs de [9] ont présenté un micro-manipulateur constitué de deux étages matriciels de 40 mm^2 intégrant des bobines électromagnétiques planes, ainsi qu'une plate-forme mobile de 5 mm^2 (voir figure 1.13(a)). L'objet est déplacé sous l'effet d'une force magnétique, le dispositif se base sur une méthode d'excitation appliquée sur les bobines qui sont à côté de l'objet, réduisant ainsi la perte de la force magnétique dans le sens de la verticale. Quand un objet est à une position Y_1 (voir figure 1.13(b)), en excitant successivement les bobines U1,U2, etc, il va se déplacer sous l'effet de la force générée par les bobines U1,U2, etc. Ce dispositif a été testé sur des objets de taille 5x5x1 mm^3 en fer plat, magnétique ferrite et magnétique NdFeB. Un convoyage maximal a été obtenu pour les objets en NdFeB dont la masse a atteint 1,2 g avec un courant de 1 A/phase et une force de poussée de l'ordre de 40 mn. La vitesse maximale de convoyage à vide de la surface est de 30 mm/s.

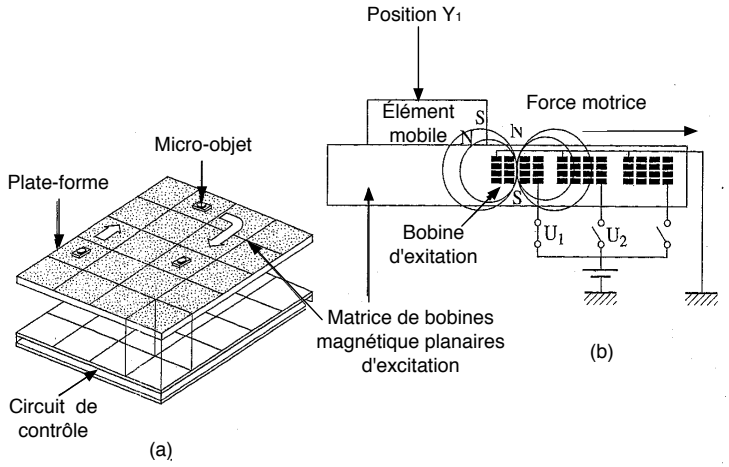


FIG. 1.13 – Un dessin conceptuel de la matrice d'actionneurs électromagnétiques [9].

Nous pouvons aussi citer les travaux de [41] qui présentent un prototype de surface active

(Xerox PARC) pour transporter des objets grâce à un flux d'air (levitating media transport). La surface active est composée de valves électrostatiques et de capteurs optiques. Elle est basée sur un système de contrôle qui est une matrice de $30,48 \times 30,48\ cm^2$ de jet d'air dirigé pour soulever et contrôler avec précision le déplacement de pièces plates, comme des feuilles de papier, sans contact avec elles.

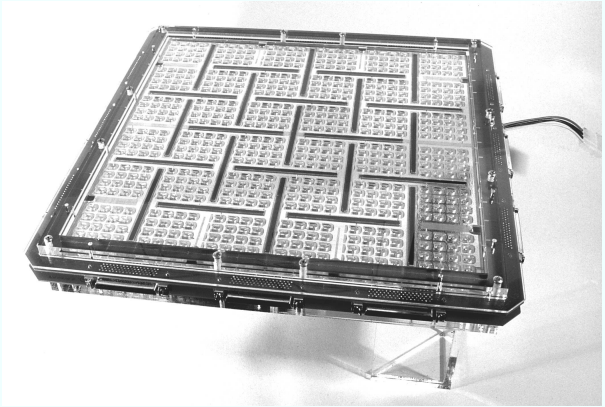


FIG. 1.14 – Image d'une surface active de taille $30,48 \times 30,48\ cm^2$ [10].

Cette surface active se compose d'une matrice de $3,2 \times 10^4$ capteurs optiques (photodiodes) qui détectent la position de la pièce en temps réel et de 1152 valves pneumatiques sur une surface de 929,03 cm^2 (matrice de $30,48 \times 30,48\ cm^2$). Quand les actionneurs de jet d'air (*valve pneumatique*) appliquent une force de contrôle distribuée sur l'objet, cette force va déplacer l'objet. Dans la surface active (voir figure 1.14), les actionneurs sont rangés sur deux plans superposés de telle façon qu'une matrice d'actionneurs située au-dessous de l'objet à transporter envoie l'air sous l'objet et une autre matrice d'actionneur positionnée par dessus l'objet envoie l'air sur l'objet. Ils fonctionnent en mode synchrone grâce à une valve de contrôle des jets d'air. Les actionneurs du bas et du haut reçoivent le même signal de contrôle. Ils appliquent donc simultanément le jet d'air sous et sur l'objet à transporter. La valve (actionneur) opère sous une force électrostatique. Quand elle s'ouvre elle libère le flux d'air en direction du papier ou de l'objet à transporter. Comme elle est très mince, elle change de sens très rapidement, de l'ordre de 1-2 kHz. Le système inclut une plaque de capteurs qui contient un tableau (bars) de 25 capteurs optiques (CMOS), dont chacun contient $1,28 \times 10^3$ CMOS photodiodes. Eclairé par un rétroprojecteur, le capteur, grâce à une lentille SELFOC qui est montée au-dessus de lui, détecte le niveau de gris (l'ombre de l'objet). C'est ce qui permet de détecter les bords de la feuille ou de l'objet à transporter.

Nous pouvons aussi citer les travaux de [11, 42] qui ont proposé un micro-actionneur (voir figure 1.15) capable de déplacer différents objets en 3 dimensions. La partie fixe du micro-actionneur est une plaque fine fabriquée en silicium. Cette plaque contient des orifices et des plaques conductrices. Les orifices permettent le passage de l'air, de manière à créer un jet d'air permettant de soulever des objets minimisant ainsi les frottements avec la plaque en silicium. Les plaques conductrices quant à elles, sous l'effet d'un champ électrique, se déplacent sur le côté, générant ainsi un flux d'air orientable, ce qui permet de déplacer l'objet selon l'horizontale. Le déplacement vertical est contrôlé par la pression de l'air appliquée aux objets et le déplacement

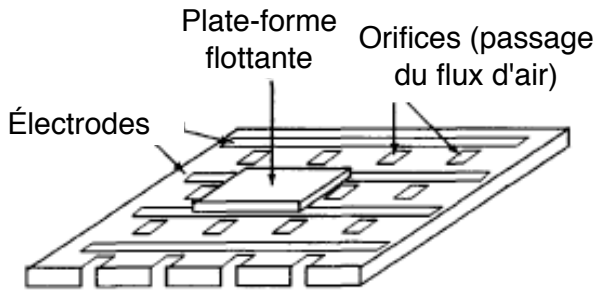


FIG. 1.15 – Architecture du système de manipulation à lévitation électrostatique [11].

horizontal est contrôlé par la force du champ électrique appliquée aux plaques conductrices. Ce micro-actionneur permet de soulever des objets d'épaisseur allant de 2 μm à 1 mm avec une force électrostatique de l'ordre de 1 à 10 nN avec un potentiel de 2 v.

D'autres réalisations suivront comme celles effectuées par Fujita et al [12] qui ont développé une matrice de micro-actionneurs pour un convoyage d'objet. La matrice de convoyage (voir figure 1.16) est composée de micro-actionneurs fluidiques qui maintiennent un objet en lévitation et le transportent grâce à un flux d'air contrôlé.

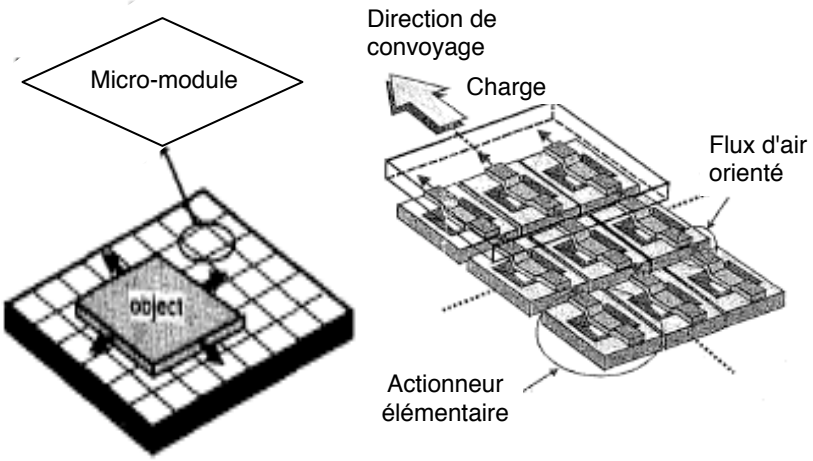


FIG. 1.16 – Image d'une matrice de convoyage basée sur le contrôle du flux d'air [12].

Le flux d'air provient des nombreux micro-actionneurs, chaque micro actionneur est de dimension 200×150 μm^2 et composé de deux orifices. L'actionneur peut déplacer les objets dans deux directions via un jet d'air. La direction du jet d'air est contrôlée grâce à des électrodes positionnées au voisinage de la canalisation (voir figure 1.17(a)). Lors de l'application d'une tension électrique entre les électrodes, sous l'effet de cette tension électrique la canalisation se referme, le flux d'air s'échappe alors par la canalisation opposée (voir figure 1.17(b)). La matrice est composée de 9 lignes de micro-actionneurs, chaque ligne contient 7 micro-actionneurs ce qui

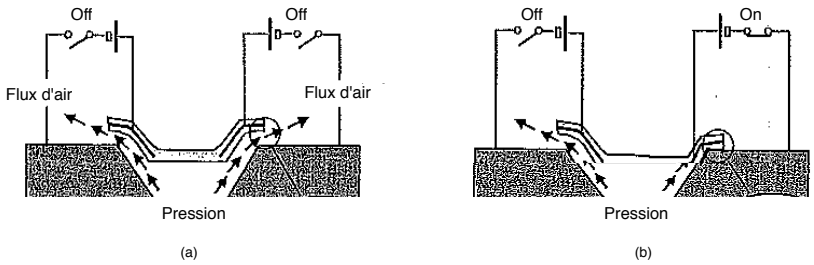


FIG. 1.17 – Mécanisme du mouvement produit par l'actionneur. (a) Situation normale absence de tension sur l'électrode. (b) Lorsqu'une tension est appliquée sur l'électrode [12].

fait un total de 63 micro-actionneurs disposés sur une surface de $2 \times 3\ mm^2$. L'expérimentation de cette approche a donné de bons résultats de convoyage de petits objets plats de surface 1 mm^2 ayant un poids de 1 mg avec une pression d'air de l'ordre de 2 kPa. L'expérience de déplacer des objets à une position souhaitée grâce à ce système a donné de bons résultats.

Fujita et al [36] ont continué à travailler depuis plus d'une dizaine d'années dans le domaine de la micro-fabrication et proposé une technologie pour la conception et la fabrication d'une surface de micro-actionneurs pneumatiques. Ils ont validé leur approche en proposant au final un dispositif composé d'une surface de $35 \times 35\ mm^2$ contenant 560 micro-actionneurs. Chaque actionneur est composé de deux couches avec une valve mobile qui se déplace sous l'effet d'un micro-actionneur électrostatique. Ils ont également proposé un contrôle distribué pour le déplacement. Chaque actionneur détecte sa position par rapport aux bords de la pièce grâce à une caméra positionnée au-dessus de la surface. Suivant la position des actionneurs par rapport à la pièce, le flux d'air est envoyé pour déplacer la pièce ou non.

Puis s'est suivi l'idée de continuer à travailler sur le concept de Smart structure en intégrant des micro-capteurs, micro-actionneurs et des unités de traitement, pour réaliser au final une Smart Surface autonome (nous reviendrons plus en détail sur la Smart Surface dans le chapitre 3).

Les systèmes de manipulateurs sans contact ont plusieurs avantages tels que la vélocité, aucun problème de frottement et potentiellement un niveau élevé de miniaturisation.

1.6 Comparaison entre les différents systèmes de manipulation

Le tableau 1.1 donne une comparaison des différents systèmes de manipulation. Les systèmes de manipulation sans contact sont réalisés en utilisant des forces pneumatiques, électrostatiques ou électromagnétiques pour déplacer l'objet en lévitation. La lévitation magnétique peut être obtenue en utilisant des aimants permanents ou un supraconducteur. Le principal avantage des systèmes de manipulation sans contact est qu'il n'y a pas de frottement, mais ils ont l'inconvénient d'avoir une grande sensibilité à l'épaisseur de la couche qui permet le convoyage d'objet en lévitation et d'avoir en général une faible capacité de charge des objets à convoyer étant donnée la difficulté du contrôle de l'épaisseur de la couche [43, 44].

Les systèmes de manipulation à contact sont réalisés à base de pieds montés sur une surface de silicium. Les pieds sont mis en marche en utilisant différents principes tels que thermique,

magnétique ou électrostatique. Ils ont l'avantage de déplacer une plus grande variété d'objets (contrairement aux systèmes de manipulation sans contact). Cependant l'augmentation de la température des pieds provoquée par leurs contacts avec l'objet à déplacer peut induire des limitations dans quelques applications.

Tab. 1.1 – Tableau comparatif des systèmes de manipulation.

Type	Descriptif du système de manipulation	Capacité de charge	Vitesse	Précision de positionnement
À contact	Une matrice de micro actionneurs électrostatiques [6]	Faible	Lente	Bonne
	Thermique (pieds mobiles) [4]	Élevée	Moyenne	Assez bonne
	Micro-manipulateurs parallèles intégrés sur CMOS [5]	Élevée	Assez élevée	Assez bonne
	Matrice d'actionneurs magnétiques à battement [45]	Élevée	Assez élevée	Non précisée
	Matrice d'actionneurs à base de roulement [7]	Élevée	Non précisée	Non précisée
Sans contact	Lévitation magnétique [8]	Faible	Lente	non précisée
	Une matrice de valves pneumatiques ; actionneurs électrostatiques [12]	Faible	Assez bonne	Assez bonne
	Lévitation électrostatique [11]	Faible	Très rapide	Assez bonne
	Électromagnétiques [9]	Elevée	Rapide	Non précisée
	Pneumatique [46]	Elevée	Rapide	Non précisée
	Surface active (déplace des papiers par jet d'air de Xerox Parc) [41]	Elevée	Non précisée	Non précisée

1.7 Conclusion

Dans ce chapitre nous avons présenté les systèmes de manipulation, défini la notion de micropièce et présenté la difficulté de leurs manipulations. Nous avons aussi présenté les MEMS, qui sont utilisés dans les technologies de fabrication indispensable pour les systèmes de manipulation à cette échelle.

Nous avons par la suite présenté différents systèmes de manipulation qui peuvent être classés en deux familles : les systèmes à contact et sans contact. Les systèmes à contact permettent

un déplacement précis des objets, ont un faible coût de construction et peuvent de déplacer de lourds objets. Leurs inconvénients majeurs sont les frottements qui peuvent endommager les objets. La deuxième famille représente les systèmes sans contact. Ces derniers ont l'avantage de ne pas être en contact avec l'objet et donc une probabilité nulle de l'endommager. Néanmoins, ils ont plusieurs inconvénients dont le coût élevé de fabrication, un déplacement peu précis des objets et la difficulté de déplacement d'objets lourds. C'est particulièrement dans cette deuxième famille que s'inscrivent nos travaux de recherche qui ont pour but la conception d'une surface intelligente autonome et distribuée à base de MEMS capable de réaliser un déplacement et un positionnement automatique de pièces de taille misoscopique.

Ce chapitre nous a permis de bien comprendre la problématique du micro-monde et des micro-manipulateurs ce qui nous a permis de bien poser le problème sur le choix des algorithmes simples de manipulation des micro-pièces. Le chapitre 2 est consacré à l'état de l'art des différentes méthodes de manipulation des micro-pièces.

La principale caractéristique de ces systèmes est leur composition de plusieurs capteurs et actionneurs. Ces derniers vont collaborer afin de réaliser l'objectif du système qui consiste à déplacer un objet et à le positionner.

Le rôle du capteur est de relever des indicateurs sur la présence de la pièce et donc de sa position. Dans le chapitre 5, nous allons voir comment cela est réalisé.

CHAPITRE 2

État de l'art sur les méthodes de différenciation

Sommaire

2.1 Introduction

Les images occupent une place prépondérante au sein de notre société. Cela implique le besoin croissant en traitement d'images numériques. Le traitement d'images numériques a pour but d'étudier les images et leurs transformations, afin d'améliorer leur qualité, d'en extraire de l'information pour pouvoir l'utiliser ou encore pour aider à la compréhension d'un besoin spécifique. C'est vers les années 50 que remonte l'origine du traitement d'images numériques [19, 47]. Il a été sollicité dans de nombreuses applications qui ont rencontré des problèmes de dégradation de signaux lors de leur transmission ou réception. À cette époque, un changement radical a été le déclencheur d'importants développements dans le domaine du traitement d'images numériques ; ce changement est dû aux équipes du JPL (*Jet Propulsion Laboratory*, Etats-Unis), qui ont numérisé l'image avant de la corriger, inversement à la traditionnelle théorie du signal qui corrige le signal avant de le numériser. C'est vers la fin des années 60, début des années 70 et grâce aux micro-ordinateurs et aux cartes spécialisées que dans le domaine de l'industrie apparaissent

des applications complètes allant de l'acquisition à la décision. On parle aussi d'analyse d'images avec notamment les travaux de détection de contour, de segmentation, d'analyse de texture, etc. C'est vers la fin des années 70 et au milieu des années 80, après l'arrivée de l'analyse d'images, qu'est apparue naturellement l'interprétation d'images et dans les années 80 qu'a eu lieu le grand boom de la robotique avec l'apparition de la vision par ordinateur, les études sur le mouvement, l'évolution vers l'analyse et le traitement d'images. À ce jour, il reste encore des travaux à faire dans le domaine de la vision par ordinateur, de la perception des formes, du mouvement et de la reconnaissance des objets, etc.

Dans ce vaste et intéressant monde du traitement d'images numériques, nous nous sommes intéressés à la forme d'une image, car le but de notre travail est de reconnaître des formes. Une forme est une entité dont la géométrie caractérise un objet. En réalité, la reconnaissance n'est pas vraiment le terme approprié. La reconnaissance implique une description sémantique de l'objet. Si l'on prend comme exemple un jeu d'enfants qui doit insérer des pièces de différentes formes dans les trous correspondants, l'enfant n'a pas une description sémantique des objets il arrive juste à associer la pièce au trou qui lui correspond. On parle alors de *différenciation*.

Si l'on considère un ensemble de formes (qui peuvent tomber sur la Smart Surface), notre objectif est alors non pas de reconnaître la pièce mais plutôt de l'associer à un type de pièces. Cela revient à *différencier* toutes les formes de cet ensemble (pour les trier).

Étant donné que la forme est une propriété fondamentale de l'objet, pour l'utiliser, il est essentiel de segmenter l'image afin de détecter les objets ou les contours d'objets.

Dans la littérature, il existe plusieurs méthodes de différenciation des formes et chaque méthode s'effectue généralement en deux étapes (voir figure 2.1) : une représentation de la forme est effectuée grâce à différentes techniques de description des formes, puis une comparaison entre les représentations est effectuée.

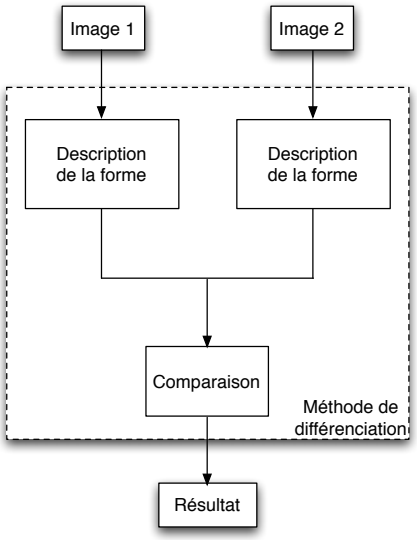


Fig. 2.1 – Organigramme des différentes méthodes de différenciation.

Le principal objectif des techniques de description des formes dans la reconnaissance d'objets

est de mesurer les attributs géométriques de l'objet qui peuvent être utilisés pour la classification et la reconnaissance d'objets. Dans la littérature, on retrouve différentes classifications des techniques de description des formes [48]. Par exemple, dans [13] les auteurs ont présenté une classification des techniques de description des formes qui se divise en deux approches [49, 50] : l'approche basée sur le *contour* et l'approche basée sur la *région*. L'approche basée sur le contour utilise seulement les contours de l'objet ; l'approche basée sur la région utilise l'intérieur de l'image en plus des contours. L'approche basée sur la région est ensuite divisée en deux-sous domaines : domaine spatial (*spatial domain*) et domaine transformé (*transform domain*). Cette sous-division dépend de l'utilisation des mesures faites directement sur la forme (*domaine spatial*) ou l'utilisation d'une transformation appliquée à la forme (*domaine transformé*) (voir figure 2.2). Néanmoins cette classification ne présente aucune autre sous-division de l'approche basée sur le contour, ce qui aurait été plus intéressant à avoir.

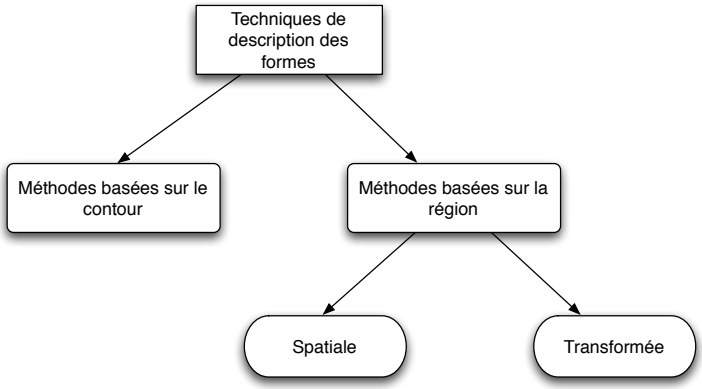


Fig. 2.2 – Organigramme des différentes méthodes de classification selon Mehtre et al [13].

Dans une autre étude [14] les auteurs ont présenté une autre classification des techniques de description des formes qui se divisent en deux approches : l'approche basée sur la transformation (*transformation based*), allant de la transformation fonctionnelle jusqu'à la transformation structurelle telle que le code de Chain, et l'approche basée sur les mesures (*measurement based*), allant des simples mesures telles que la surface jusqu'aux mesures plus sophistiquées telles que les moments invariants. L'approche basée sur la transformation est ensuite divisée en deux sous-catégories : catégorie fonctionnelle (*functional categorie*) et catégorie structurelle (*structural categorie*) (voir figure 2.3). Cette classification ne fait aucune distinction entre l'approche basée sur les régions et l'approche basée sur les contours.

Enfin, dans [15] les auteurs ont proposé une classification plus riche et globale en comparaison avec les deux précédentes étant donné qu'elle traite les inconvénients constatés précédemment, à savoir la classification des méthodes basées sur le contour et des méthodes basées sur la région et la sous-division de l'approche basée sur le contour. Des techniques supplémentaires ont été ajoutées dans cette approche (voir figure 2.4).

Étant donné notre besoin de différencier des micro-pièces à l'échelle mésoscopique, détectées par des capteurs optiques binaires, nous ne traiterons que les descripteurs de *forme* de la figure 2.4. De plus, parmi toutes les méthodes de différenciation présentées (voir figure 2.4), nous

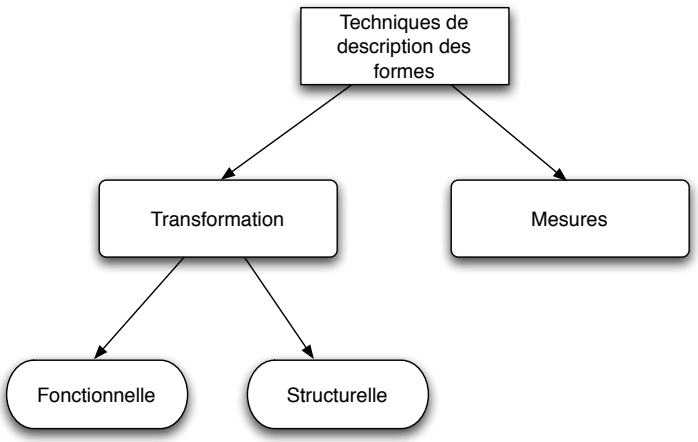


FIG. 2.3 – Organigramme des différentes méthodes de classification selon Tao et al [14].

ne présenterons dans ce chapitre que quelques-unes d'entre elles, les plus largement utilisées, les plus connues, celles considérées comme les meilleures et les plus simples.

En outre, dans le cadre du projet Smart Surface, nous avons besoin d'une méthode pour la différenciation des formes et non pas pour la reconnaissance. Cette méthode doit être très performante et très rapide. Dans cette optique nous avons opté pour des méthodes basées sur le contour et sur la région et nous ne traiterons pas leurs sous-divisions.

2.2 Approche basée sur le contour

Dans l'approche basée sur le contour, seuls les pixels de la frontière sont pris en considération. Cette approche a donné de bons résultats pour quelques types d'applications tels que la reconnaissance de caractères ou le codage d'objets. Le descripteur de contour le plus utilisé est le descripteur de Fourier [51, 52, 16].

2.2.1 Descripteur de Fourier

Dans cette approche, l'image est définie par une fonction à une dimension, appelée *signature de forme*, qui est une représentation compacte de l'image [53]. Puis une transformée de Fourier est appliquée à cette signature (voir figure 2.5).

Signature de forme

Plusieurs méthodes ont été développées pour calculer la signature d'une forme. Nous allons décrire les trois signatures de forme les plus utilisées, qui sont : *distance centroide, chordLength signature* et *fonction de la surface.*

a. Distance centroide : cette fonction se définit par la distance entre le centre de la forme (x_o, y_o) et les points se trouvant sur la frontière [16, 54]. Les points sur la frontière sont définis

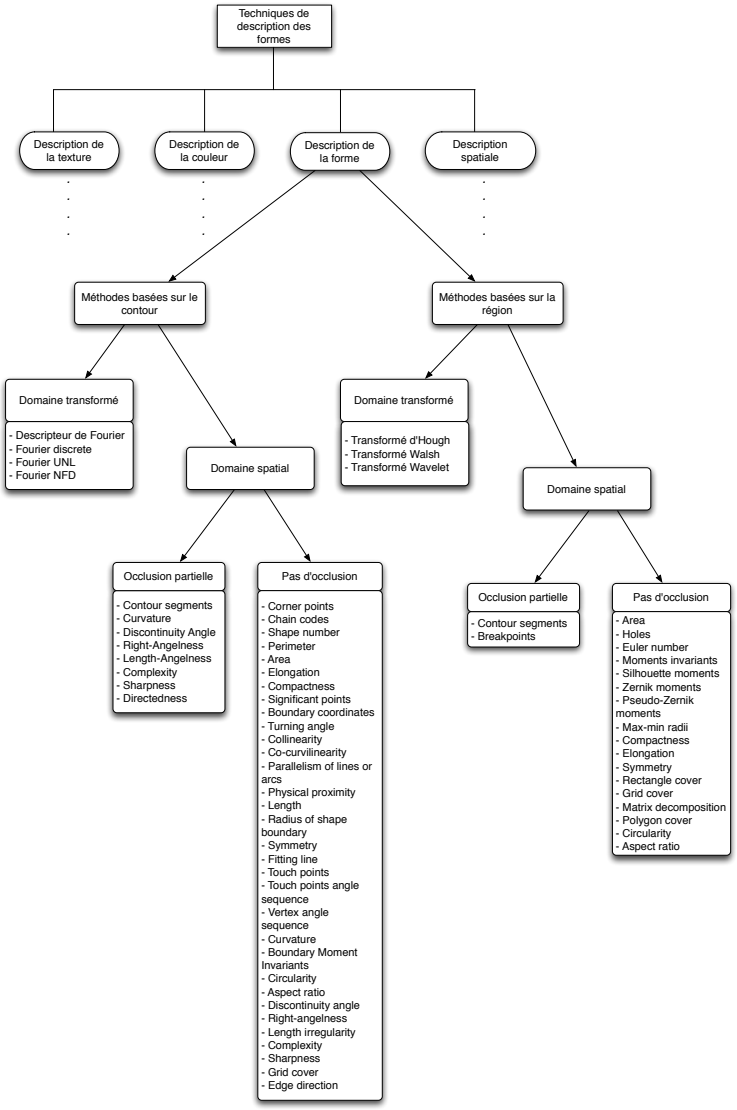


FIG. 2.4 – Organigramme des différentes méthodes de classification selon Safar et al [15].

de telle sorte que la distance entre eux soit identique (voir figure 2.6). La signature distance centroide se définit par l'équation 2.1 :

$$Z(t) = [x(t) - x_c] + i[y(t) - y_c] \tag{2.1}$$

où (x_c, y_c) sont les coordonnées du barycentre de la forme définie par l'équation 2.2 :

$$x_c = \frac{1}{N}\sum_{t=0}^{N-1} x(t), y_c = \frac{1}{N}\sum_{t=0}^{N-1} y(t) \tag{2.2}$$

et $x(t), y(t)$ sont les coordonnées du point t, N le nombre de points de la forme, et $\mathrm{i}^2 = -1$.

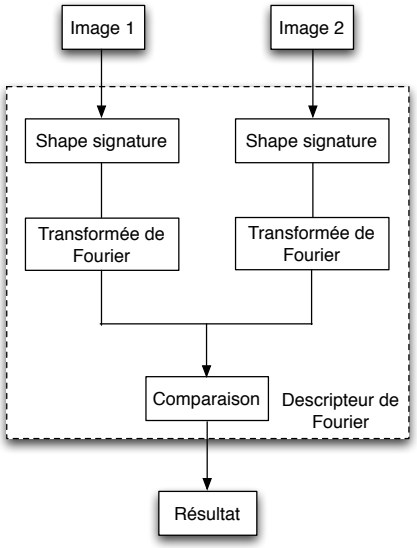


FIG. 2.5 – Organigramme de la méthode de différenciation en utilisant le descripteur de Fourier.

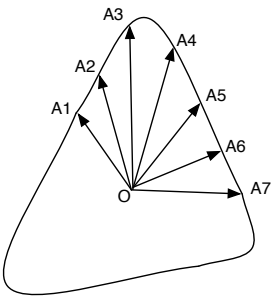


FIG. 2.6 – Distance centroide [16].

b. Signature ChordLength : le problème de l'approche distance centroide est qu'elle nécessite un point de référence et elle ne peut pas être appliquée sur des formes non convexes. Pour cela, la signature ChordLength a été proposé [17]. La signature ChordLength $r^*(t)$ est une dérivée de la distance centroide sans point de référence. Pour chaque point de la frontière P (voir figure 2.7), la $r^*(t)$ est calculée. La $r^*(t)$ représente la distance entre P et un autre point de la frontière $P^{'}$ tel que $PP^{'}$ soit perpendiculaire au vecteur tangent en P. S'il existe plusieurs points candidats pour P (dans notre exemple P_1 est un point candidat pour P) alors le point milieu de PP_1 est calculé et P_1 est éliminé car P_2 est à l'extérieur de la forme. Si P_2 est à l'intérieur de la forme, le milieu de PP_2 et le milieu de P_1P_2 sont calculés et ce processus est répété jusqu'à ce que le point milieu se trouve à l'extérieur de la forme.

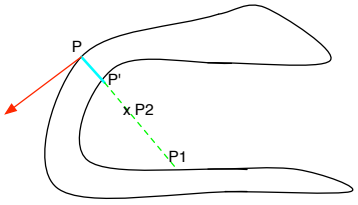


FIG. 2.7 – Exemple d'application de la signature ChordLength sur une forme non convexe [17].

c. Fonction de la surface : cette fonction se définit par la surface de la forme, par exemple le triangle formé par O, P_1 et P_2 représente la fonction de la surface (voir figure 2.8), avec l'équation 2.3 :

$$A(t) = \frac{1}{2} \mid x_1(t)y_2(t) - x_2(t)y_1(t) \mid \tag{2.3}$$

Dans la fonction de la surface, pour chaque point de la frontière, la surface du triangle formé par les points de la frontière et le centre de la forme est calculée [17].

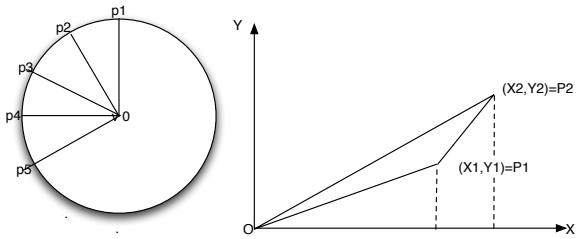


FIG. 2.8 – Fonction de la surface pour une image en forme de pomme [17].

Transformée de Fourier

Une fois la shape signature $u(t)$ calculée, la transformée de Fourrier est appliquée [55, 56]. La transformée de Fourier est définie par l'équation 2.4 :

$$x_n = \frac{1}{N} \sum_{t=0}^{N-1} u(t) exp(-i2\pi nt/N), \tag{2.4}$$

$n = 0, 1, ..., N-1$, $\mathrm{i}^2 = -1$.

Les coefficients x_n, $n = 0, 1, ..., N-1$, sont généralement appelés descripteurs de Fourier de la forme. Ces descripteurs représentent la forme de l'objet dans un domaine fréquentiel. Les fréquences faibles des descripteurs contiennent des informations globales sur les caractéristiques de la forme. Bien que le nombre de coefficients générés de la transformée soit souvent large, un ensemble de coefficients est suffisant pour capturer l'ensemble des caractéristiques de la forme. Les fréquences hautes des descripteurs décrivent les détails de la forme, qui ne sont pas utiles dans la discrimination des formes et par conséquent, ils peuvent être ignorés. La transformée de Fourier est invariante à la translation, à l'homothétie, à la rotation et au point de départ.

2.2.2 Code de Chain

Le code de Chain (aussi appelé le code de Freeman) peut être utilisé pour représenter la frontière d'une quelconque forme. La frontière est codée selon les quatre directions (voir figure 2.9(a)) ou les huit directions (voir figure 2.9(b)) [18, 19]. Le codage s'effectue en parcourant la frontière de l'image à partir d'un point de départ et à coder, au fur et à mesure, des déplacements élémentaires permis, suivant un sens donné (dans le sens des aiguilles d'une montre ou le sens contraire).

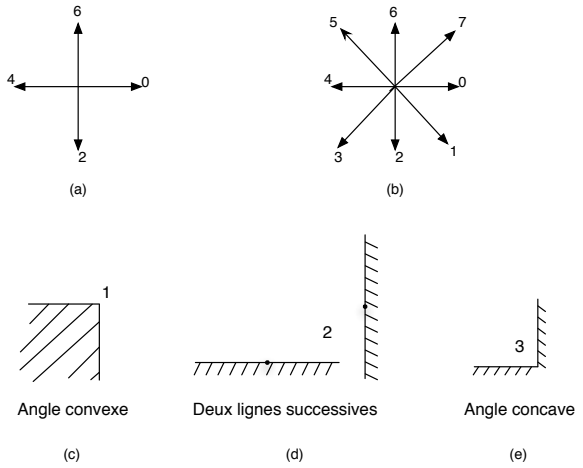


FIG. 2.9 – Illustration du code de Chain et de ses variations [13, 18].

Dans la figure 2.10, le point le plus à gauche et le plus haut est choisi comme point de départ. À partir de ce point, le code obtenu suivant les huit directions de déplacement est 00022222444446600066.

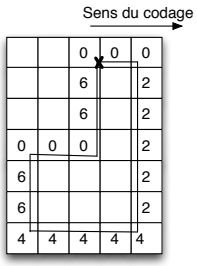


FIG. 2.10 – Exemple d'application du code de Chain à huit directions sur une forme.

Le code de Chain est invariable à la translation. Il est aussi invariable à une rotation de $k \times 90°$ pour la 4-connectivité ou $k \times 45°$ pour la 8-connectivité [57, 58].

Un autre code appelé *code de Chain dérivé* a été développé par Bribiesca et Guzman [59], qui au lieu d'utiliser l'orientation pour le codage, utilisent "la dérivation du code de Chain". Il est formulé comme suit [13] :

– un angle convexe est codé par 1 (voir figure 2.9(c)) ;
– deux lignes successives dans la même direction sont codées par 2 (voir figure 2.9(d)) ;
– un angle concave est codé par 3 (voir figure 2.9(e)).

Le code associé à une forme doit être de même ordre. Pour obtenir un code unique d'une forme, toutes les permutations cycliques du code sont calculées. Le code unique est l'une de ces permutations telle que le nombre obtenu soit le plus petit. Le code de Chain associé à la figure 2.10 devient donc 00066000222224444466.

Le contour d'un objet est représenté par un code de Chain qui est considéré comme une chaîne de caractères qui décrit l'objet. Pour comparer des images entre elles, une comparaison des codes de Chain, qui les caractérisent, est effectuée. Cette comparaison peut être effectuée en calculant des *mesures de distance*. Il existe trois types de mesures de distance : distance pondérée de Levensthein « *weighted Levensthein distance* »(WLD), correspondance non linéaire« *nonlinear elastic matching* »(NEM) et distance étendue « *extended distance* »(ED), voir figure 2.11.

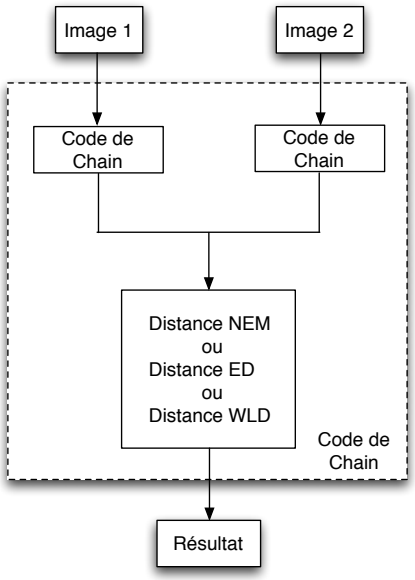


FIG. 2.11 – Organigramme de la méthode de différenciation utilisant le code de Chain.

Dans ce qui suit, chacune de ces mesures sera détaillée. Soit A et B deux codes de Chain :

$$A = \{a_1, a_2, ..., a_n\},\ B = \{b_1, b_2, ..., b_m\} \tag{2.5}$$

avec n, m la taille des codes de chain A et B.

Si s_k, i_k et d_k substitutions, insertions et suppressions sont nécessaires pour réécrire le code A dans B et p, q et r sont respectivement les coûts relatifs aux substitutions, insertions et suppressions, alors :

1. La distance WLD entre les codes A et B est définie par l'équation 2.6 :

$$D_{WLD} =_k^{min} \{ps_k + ki_k + rd_k\} \tag{2.6}$$

2. La distance NEM est définie par l'équation 2.7 :

$$D_{NEM} = l(m,n) \tag{2.7}$$

où
$l(i,j) = min(l((i-1),j-1) + w_s, l(i,j-1) + w_i, l(i-1,j) + w_d)$
Pour $1 \leq i \leq m$ *et* $1 \leq j \leq n$ avec

$$w_s = \begin{cases} 0 & \text{si } a_i = b_i \\ p & \text{sinon} \end{cases}$$

avec condition sur la frontière
$l(i,0) = ir \quad i = 0,1,2,...,m$
$l(0,j) = jq \quad j = 0,1,2,...,n$

3. La distance ED peut aussi se définir par l'équation 2.8 :

$$D_{ED} = l(m,n) \tag{2.8}$$

avec

$$w_s = \begin{cases} 0 & \text{si } a_i = b_j \\ \alpha p & \text{si } ((a_{i-1} = a_i = b_{j-1}) \, et \, (a_{i+1} = b_j = b_{j+1})) \text{ où} \\ & \text{si } ((a_{i-1} = b_j = b_{j-1}) \, et \, (a_{i+1} = a_i = b_{j+1})) \\ p & \text{sinon} \end{cases}$$

$$w_i = \begin{cases} \beta q & \text{si } a_i = b_{j-1} = b_j \\ p & \text{sinon} \end{cases}$$

$$w_i = \begin{cases} \gamma r & \text{si } a_{i-1} = a_i = b_j \\ r & \text{sinon} \end{cases}$$

où α, β et γ sont des nombres réels appartenant à $[0;1]$.

2.2.3 Squelettes

Dans cette méthode, le squelette de la forme X est défini par tous les centres des plus grands cercles strictement contenus dans X [19, 18] (voir figure 2.12). En d'autres termes, un point s

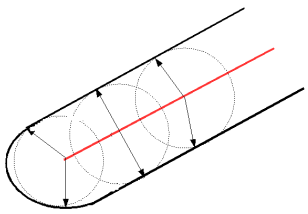


FIG. 2.12 – Définition du squelette [19].

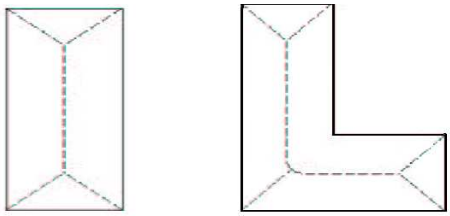


FIG. 2.13 – Squelettes des différentes formes.

fait partie du squelette $Sq(X)$ si est seulement si s est équidistant de deux points distincts du contour (voir figure 2.13).

Il est clair que le squelette dépend entièrement de la distance par rapport au contour et de sa régularité. Le squelette n'a pas de bonnes propriétés. En fait, le squelette d'un cercle légèrement déformé sur le contour n'est plus un point mais une ligne droite qui relie le centre du cercle à la déformation (voir figure 2.14).

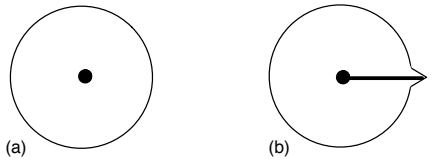


FIG. 2.14 – (a) Squelette d'un cercle. (b) Squelette d'un cercle avec une légère déformation [19].

Cette grande sensibilité au bruit est le problème principal de la méthode du squelette. Elle est aussi très coûteuse en calcul car elle nécessite le calcul de la distance entre tous les points à l'intérieur de l'objet avec les points du contour.

L'extraction du squelette est illustrée par la propagation d'un feu de prairie. Le feu est allumé en même temps sur l'ensemble du contour de la prairie et se propage à la même vitesse vers le centre. Le feu s'éteint en raison du manque de carburant le long d'une ligne d'extinction correspondant à la forme de squelette de la prairie (voir figure 2.15).

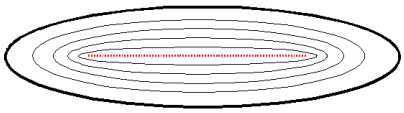


FIG. 2.15 – Illustration de la méthode squelette : technique de feu de prairie [19].

Pour obtenir le squelette, la méthode la plus couramment utilisée est l'amincissement jusqu'à convergence, c'est-à-dire lorsque l'amincissement ne génère plus aucune modification.

Sans parler de sa sensibilité au bruit, la méthode de squelette n'est pas très intéressante pour notre étude car nous travaillons sur des micro-pièces où l'amincissement n'apporte pas beaucoup d'informations.

Les deux méthodes, descripteur de Fourier et code de Chain, présentées ci-dessus, quant à elles sont très efficaces et largement utilisées pour de grandes images où le contour diffère nettement de l'intérieur de l'image. Dans notre étude, ces méthodes ne sont pas très intéressantes étant donné que nous travaillons sur de micro-pièces où le contour est très proche de la surface.

De plus, nous avons besoin de capter les caractéristiques d'une forme avec un algorithme à complexité linéaire ; tous les pré-traitements avec des algorithmes exponentiels sont très coûteux et ne sont donc pas adaptés à nos besoins.

Ainsi nous terminons notre étude sur les méthodes basées sur le contour. Dans la prochaine section, nous allons parler des méthodes basées sur la région.

2.3 Approche basée sur la région

Dans l'approche basée sur la région, on prend en considération les pixels à l'intérieur de la forme. Les meilleurs descripteurs de cette catégorie sont ceux qui sont basés sur les moments. Ces descripteurs sont devenus de plus en plus populaires grâce à leur description compacte, à leur performance et à leur possibilité de varier le niveau de détail à atteindre, ce qui explique leur présence dans diverses applications [60].

2.3.1 Méthode Grid-based

Dans cette méthode [16, 61], on trace une grille de cellules, de taille fixe, sur l'image, comme le montre la figure 2.16.

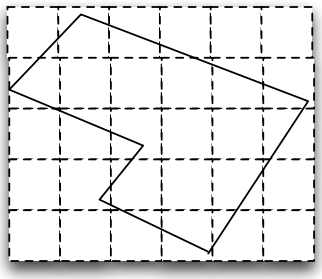


Fig. 2.16 – Tracé d'une grille sur une image.

On parcourt notre grille de gauche à droite et de haut en bas, on affecte la valeur 1 pour chaque cellule qui est totalement ou partiellement couverte par la forme et 0 sinon [61, 20].

On obtient ainsi un nombre binaire, qui est la représentation de notre forme. Exemple : la représentation binaire de la forme de la figure 2.16 est : 01100011111000111001110000100. La différence entre deux pièces est donnée par le XOR entre leurs représentations binaires.

Il est évident qu'une telle représentation binaire est très sensible à la rotation, à la translation et à l'homothétie, ce qui nécessite un processus de pré-normalisation [20]. Le processus de pré-normalisation s'effectue en trois étapes : normalisation pour la rotation, pour la translation et pour l'homothétie.

Normalisation pour la rotation

Lorsqu'une image subit une rotation, celle-ci change la position de la forme dans la grille, donc modifie sa représentation binaire. Pour palier ce problème, avant de parcourir la grille pour écrire le nombre, on effectue une série de rotations sur la forme jusqu'à ce que l'axe reliant les deux points les plus éloignés de la forme soit parallèle à l'axe des abscisses (voir figure 2.17). Cet axe est appelé axe majeur (*major axis*).

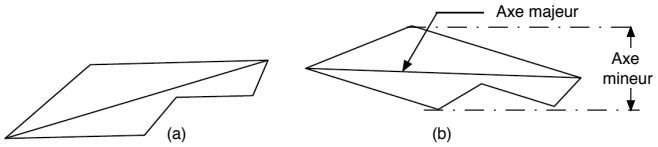


FIG. 2.17 – Normalisation pour la rotation (a) pièce avant la normalisation, (b) pièce après la normalisation [20].

Normalisation pour l'homothétie

Lorsque la taille de la forme change, sa représentation binaire change aussi. Une longueur d'axe est arbitrairement fixée. Cet axe est considéré comme un axe principal aussi appelé axe principal normalisé. On applique une homothétie sur la forme (après avoir effectué une normalisation pour la rotation) jusqu'à ce que l'axe majeur de la forme soit égal à l'axe principal normalisé (voir figure 2.18).

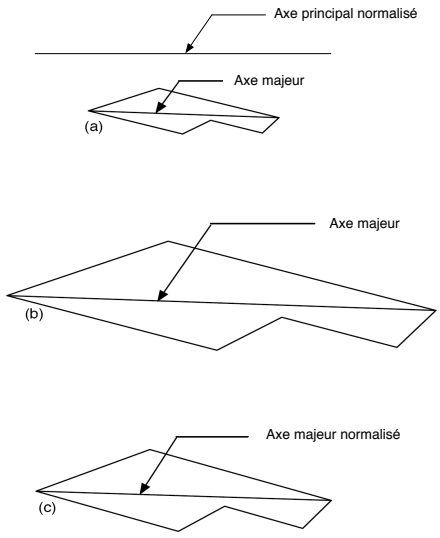


FIG. 2.18 – Deux formes similaires : (a) et (b) avant la normalisation pour l'homothétie, (c) après normalisation pour l'homothétie [20].

Les formes dans les figures 2.18(a) et (b) sont identiques mais à une échelle différente. Dans

la figure 2.18(c) la forme est normalisée, elle est obtenue à partir de la forme de la figure 2.18(a) ou la figure 2.18(b). Cette forme normalisée est obtenue en appliquant des changements d'échelle sur la forme, le long de l'axe des abscisses et l'axe des ordonnées jusqu'à ce que l'axe majeur soit égal à l'axe principal normalisé. Afin de préserver la perception de la forme après application de l'homothétie, on applique l'homothétie sur la forme proportionnellement à l'axe mineur. L'axe majeur et l'axe mineur sont perpendiculaires et définissent le plus petit carré dans lequel la forme est contenue.

Normalisation pour la translation

Une fois la forme normalisée en rotation et en homothétie, l'invariance à la translation est obtenue par le fait de tracer une grille sur la forme.

Après avoir appliqué le processus de normalisation, on balaye la grille et on génère le code binaire. Pour la mesure de similitude entre deux pièces le XOR est calculé entre leurs représentations binaires. Le nombre binaire résultant représente la différence entre les deux pièces (voir figure 2.19). Cette méthode est très intéressante à appliquer pour différencier des images très

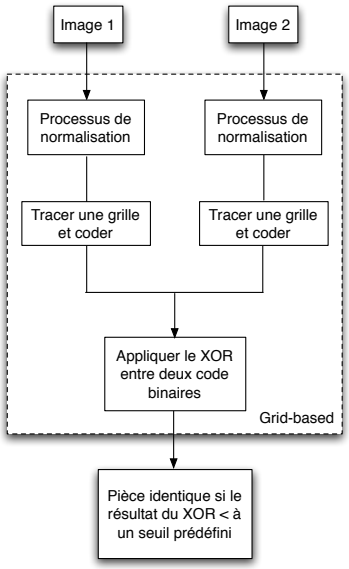


FIG. 2.19 – Étape de la méthode de Grid-based.

proches car elle se base sur la représentation de l'image.

2.3.2 Moments d'Hu

Dans cette méthode [62, 63, 64] l'image est représentée par les moments invariants. Le moment M d'une image I à deux dimensions d'ordre $(p+q)$ avec $p, q > 0$ est défini par la formule 2.9 :

$$M_{pq} = \sum_{x=0}^{m} \sum_{y=0}^{m} x^p y^q I(x,y) \qquad p,q = 0,1,2,3... \tag{2.9}$$

où :
M_{pq} : le $(p+q)^{\text{ème}}$ moment de l'image
$m * m$: la taille de l'image
$I(x,y)$: la fonction décrivant l'image en chacun de ses points de coordonnées (le niveau de gris d'un pixel de l'image)

Invariance à la translation

On définit le moment moyen invariant à la translation par la formule 2.10 :

$$\mu_{pq} = \sum_{X} \sum_{Y} (x - \bar{x})^p (y - \bar{y})^q \tag{2.10}$$

avec :

$$\bar{x} = \frac{M_{10}}{M_{00}}, \bar{y} = \frac{M_{01}}{M_{00}}$$

x et y : les centroïdes ou barycentre de l'image, M_{00}, M_{01}, M_{10} : respectivement les moments d'ordre 0,1.

Invariance à l'homothétie

Les moments moyens normalisés et invariants aux changements d'échelle sont obtenus par la formule 2.11 :

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}} \tag{2.11}$$

avec :

$$\gamma = \frac{p+q}{2} + 1$$

Invariants à la rotation

Les moments réduits décrits dans les formules précédentes sont invariants à la translation et à l'homothétie. Afin d'obtenir l'invariance à la rotation, H. Ming-Kuel a proposé en 1963 un ensemble de 7 descripteurs appelés *invariants de Hu* [65]. Ces descripteurs sont construits par une combinaison non linéaire des moments jusqu'à l'ordre 3. Ces moments sont représentés ainsi :
$M_1 = (\eta_{20} + \eta_{02})$,
$M_2 = (\eta_{20} - \eta_{02})^2 + 4{\eta_{11}}^2$,
$M_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$,
$M_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$,
$M_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})$
$(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$,
$M_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$,
$M_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] - (\eta_{30} + 3\eta_{12})$

$(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$.

Ces descripteurs sont utilisés comme des vecteurs d'entrée pour la méthode de classification. Il existe plusieurs méthodes de classification, parmi elles, les réseaux de neurones sont les plus utilisés pour leur tolérance aux fautes, leur capacité de classification et leur généralité. L'architecture du réseau est la suivante :

- les entrées : le réseau reçoit sept valeurs en entrée ;
- la couche cachée : qui définit l'ensemble des neurones intermédiaires ;
- la couche de sortie : qui définit la réponse des neurones par rapport aux entrées.

Une autre méthode peut être appliquée pour comparer deux images, qui consiste à calculer la distance euclidienne entre les codes d'Hu des images. Souvent les valeurs des moments d'Hu sont très petites, et doivent être normalisées pour la comparaison. Cette normalisation est effectuée en calculant les limites de chaque moment invariant.

La distance d_m entre deux vecteurs M^Q et M^I associés respectivement aux images Q et I est obtenue en calculant la distance euclidienne avec l'équation 2.12 :

$$d_m(M^Q, M^I)) = \sqrt{\sum_{i=0}^{n}(M_i^Q - M_i^I)^2} \tag{2.12}$$

Des images sont dites identiques ou similaires si la valeur de d_m est égale ou très proche de zéro.

La figure 2.20 présente un schéma de différenciation basée sur les moments d'Hu.

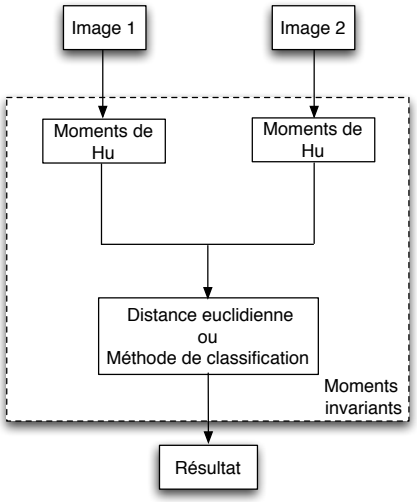


FIG. 2.20 – Diagramme de différenciation basée sur les moments d'Hu.

Les moments invariants peuvent être considérés comme une projection de la fonction $I(x, y)$ sur l'espace des polynômes engendrés par x^p, y^q $(p, q) \in N^2$. L'inconvénient est que cet espace n'est pas orthogonal, ce qui donne une représentation sous-optimale. Il existe une extension de cette approche qui est basée sur des fonctions orthogonales, par exemple les Moments de Zernike.

2.3.3 Moments de Zernike

Les moments de Zernike ont vu le jour en 1934 (proposé par Zernike). Ils sont largement connus pour leur invariance au bruit et leur redondance d'information et se définissent en utilisant un ensemble de polynômes complexes qui forment une base orthogonale normale définie à l'intérieur du cercle unitaire, c'est-à-dire tel que $x^2 + y^2 \leq 1$. La forme de ces polynômes est donnée par l'équation 2.13 [60] :

$$V_{nm}(x, y) = V_{nm}(\rho, \theta) = R_{nm}(\rho) \cdot \exp^{(im\theta)} \tag{2.13}$$

où
n : un entier positif ou nul
m : un entier tel que $\|m\| \leq n$
ρ : longueur du vecteur entre l'origine et le pixel (x,y)
θ : angle entre le vecteur p et x
$\mathrm{i}^2 = -1$
R_{nm} : polynôme radial définit par l'équation 2.14 :

$$R_{nm}(\rho) = \sum_{s=0}^{\frac{(n-|m|)}{2}} (-1)^s \frac{(n-s)!}{s!(\frac{(n+|m|)}{2} - s)!(\frac{(n-|m|)}{2} - s)!} r^{n-2s} \tag{2.14}$$

Pour calculer les moments de Zernike d'une image, le centre de l'image est pris comme pixel d'origine et les cordonnées des pixels sont calculées suivant un cercle unitaire. Les pixels se trouvant à l'extérieur du cercle ne sont pas pris en considération. Les moments de Zernike sont invariants à la rotation mais sensibles à la translation et à l'homothétie. Pour une utilisation efficace des moments de Zernike dans la reconnaissance des formes, l'image doit être normalisée pour la translation et l'homothétie. Un autre facteur important à noter est que les moments de Zernike ne prennent en compte que l'image contenue dans un cercle unitaire, cela implique la réduction de l'image au cercle unitaire pendant le processus de normalisation. L'invariance à la translation peut être obtenue en déplaçant l'origine au centre de chaque image donnée.

Pour l'invariance à l'homothétie, l'image est réduite à un carré de la taille de M pixels et la distance entre les pixels est divisée par $\sqrt{2M^2}$, de manière à ce que l'image tienne dans le cercle d'unité. Avec le changement d'échelle de l'image en un carré d'unité, il y a perte d'informations et notamment de l'aspect ratio[8]. Il n'est pas possible de distinguer un cercle d'une ellipse avec cette échelle. Afin de conserver les informations, il est nécessaire de calculer l'aspect ratio avant le changement d'échelle.

Une image peut être dans n'importe quelle orientation, il en résultera différentes formes avec des valeurs différentes du moment de Zernike. Pour éviter ce problème et afin de le rendre invariant à l'orientation, il est nécessaire de calculer le rectangle de frontière minimal.

Soit $A = \{a_1, a_2, ..., a_n\}$ et $B = \{b_1, b_2, ..., b_n\}$ les vecteurs de moments de Zernike pour les deux images A et B avec n la taille des vecteurs. Soit $R = \{r_1, r_2, ..., r_n\}$ le vecteur rapport de A et B avec $r_i = \frac{a_i}{b_i}$. La variance R est calculée comme une distance de similitude Sim_{dist} entre

[8] L'aspect ratio correspond au rapport largeur/hauteur d'une image.

deux images avec l'équation 2.15 :

$$Sim_{dist} = \frac{1}{n}\sum_{1}^{n}(r_i - R_{avg})^2 \tag{2.15}$$

où :
$R_{avg} = -\frac{1}{n}\sum_{1}^{n} r_i$

Le Sim_{dist} est égal ou presque égal à zéro pour des images identiques (ou similaires). Il est plus grand pour des images non similaires.

Nous pouvons aussi citer les moments pseudo-Zernike qui sont une variante des moments de Zernike [66]. Les polynômes pseudo-Zernike sont une série de polynômes orthogonaux à x, y, r, qui diffèrent de ceux de Zernike car le polynôme radial est défini par l'équation 2.16 :

$$R_{nm}(\rho) = \sum_{s=0}^{(n-|m|)} (-1)^s \frac{(2n+1-s)!}{s!(n-|m|-s))!(n+|m|+1-s))!} r^{n-2s} \tag{2.16}$$

où $n = 1, ..., \infty$, et m est un entier tel que $m \leq n$. Cette série de polynômes de pseudo-Zernike contient $(n+1)^2$ polynômes linéairement indépendants de degré n, alors que l'ensemble des polynômes Zernike ne contient que $(n+1)(n+2)/2$ polynômes linéairement indépendants de degré n.

La mesure de similarité pour les moments pseudo-Zernike est la même que celle utilisée dans les moments de Zernike.

Les moments invariants sont largement utilisés dans les modèles à trois dimensions ou pour de grandes images qui ont besoin d'être compactées. Les moments de Zernike et pseudo-Zernike sont aussi largement utilisés et très connus pour leur robustesse aux diverses perturbations de l'image. Il n'est pas nécessaire d'appliquer ces méthodes dans notre cas car nous travaillons avec des images très petites.

2.3.4 La méthode de la matrice de forme

Cette méthode est définie en plaçant une grille radiale de taille $M \times N$ composée de M cercles et de N secteurs au centre de la région (voir figure 2.21), puis en affectant 1 ou 0 dans les cases de la grille selon que le point d'intersection[9] entre les rayons et le cercle de cette grille soit à l'intérieur de la forme ou non [21].

Positionner la grille au centre de la région rend la méthode robuste à la translation. Pour la robustesse à l'homothétie, le rayon du cercle est défini comme le plus grand rayon du cercle contenant la forme tandis que les nombres de secteurs et d'anneaux de la grille sont fixes. Pour la robustesse à la rotation, la direction doit être établie. Cette direction peut être définie par le point le plus éloigné du centre de la forme.

Cette méthode s'applique pour compresser des images complexes, les points d'intersection entre les rayons et les cercles sont enregistrés dans une matrice qui est une représentation compacte de la forme. Une telle méthode n'est pas très intéressante dans notre étude étant donné que nous travaillons sur des petites images.

[9]Comparée à la méthode Grid-based cette méthode utilise le point d'intersection et non le contenu de la cellule.

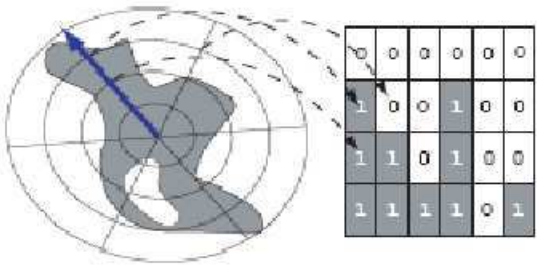


FIG. 2.21 – Matrice de forme [21].

2.4 Comparaison des différentes techniques

Nous avons présenté quelques méthodes des deux approches, la première basée sur le contour et la deuxième basée sur la région. Nous constatons (voir le tableau 2.1) que la méthode Grid-based est sensible à la taille de la grille de cellules. En effet, utiliser une grille de petite taille permet de générer un nombre binaire plus précis et donc une meilleure performance. Cependant, l'utilisation d'une petite grille de cellule implique l'augmentation de l'espace de stockage et les coûts de calcul, notamment celui de la mesure de similarité (comparaison) entre deux formes. Il est aussi à noter que la méthode Grid-based n'est pas très robuste au bruit. Cette sensibilité est due à l'utilisation de l'axe majeur dans le processus de normalisation de la rotation (voir figure 2.22). Une légère déformation de la frontière de la forme peut impliquer un changement dans le choix de l'axe majeur. Dans [20], les auteurs ont proposé de remplacer dans le processus de normalisation de la rotation la méthode d'axe majeur en utilisant une méthode d'élongation. Cette dernière se calcule en se basant sur un ensemble de points de la frontière. Dans le cas de perturbations, la méthode d'élongation s'est avérée moins robuste que la méthode basée sur l'axe majeur. Cela s'explique par le fait que la méthode d'élongation est affectée par la perturbation de n'importe quel point de son ensemble. Alors que la méthode basée sur l'axe majeur est affecté seulement par le point qui va impliquer un changement dans l'axe majeur.

FIG. 2.22 – Deux formes similaires avec une légère aspérité donnant deux axes majeurs différents.

La méthode de descripteur de Fourier est plus robuste au changement du nombre de coefficients de Fourier. Cela s'explique par le fait que les grandes fréquences contiennent des informations sur les détails fins de la forme et non sur les caractéristiques globales de la forme. C'est pour cela qu'il est plus intéressant d'utiliser les faibles fréquences des coefficients de Fourier seulement pour indexer et mesurer la similarité des formes dans le but d'optimiser l'espace de

TAB. 2.1 – Comparaison des différentes méthodes.

Méthode	Performance	Coût mémoire	Bruit	Inconvénient
Grid-based [61]	Très bonne, particulièrement pour les formes similaires	Très coûteuse	Robuste	Sensible à la taille de la grille de cellules utilisée
Descripteur de Fourier [53]	Bonne	-	Bonne robustesse	-
Moments d'Hu [64]	Faible	-	Robuste au bruit gaussien	Pas très performante et très sensible au bruit
Moments de Zernike [60]	Performant, particulièrement lors de l'occultation	-	-	Sensible à la modification de fond.
Code de Chain [59]	Oui	Non	Non	Très sensible à la résolution de l'image et la robustesse des contours
Squelette [19]	Assez performante	Coûteuse	Non	Très sensible au bruit et a l'occultation.
Signature fonction de la surface [17]	Très performante	-	-	Nécessite un point de référence
Fonction distance centroide [16]	Performante	-	-	Nécessite un point de référence
Signature Chord-Length [17]	Assez bonne	Coûteuse	Pas robuste	Similaire aux fonctions distance centroide et fonction de la surface pour des formes pas très complexes

stockage nécessaire et le coût de calcul.

La méthode Grid-based donne en moyenne de meilleurs résultats que les méthodes basées sur les descripteurs de Fourier et les moments d'Hu. Cependant les performances de la méthode de descripteur de Fourier sont très proches de celles de la Grid-based. Néanmoins les moments d'Hu ont la plus faible performance. De plus la méthode Grid-based est plus robuste au bruit que la méthode de descripteur de Fourier ou la méthode des moments d'Hu. La méthode des moments d'Hu est robuste au bruit gaussien.

La méthode des moments de Zernike est sensible au changement de fond mais présente une bonne robustesse lors de l'ajout du bruit ou de l'occultation.

La méthode de descripteur de Fourier dérivée de la distance centroide et celle dérivée de la fonction de la surface sont les plus performantes en comparaison des transformées de Fourier dérivées des autres signatures. Cependant, ces deux signatures utilisent toutes les deux un point de référence qui peut être biaisé sous l'effet de déformation. Pour palier ce problème, la signature ChordLength élimine le point de référence mais a des conséquences sur la robustesse au bruit. Cette signature donne des résultats très similaires aux résultats des signatures de forme et fonction de la surface lorsqu'elle est appliquée sur des formes pas très complexes. La signature ChordLength est coûteuse et pas très robuste au bruit.

2.5 Conclusion

Il existe dans la littérature différentes méthodes de description de forme. Elles peuvent être classées en deux catégories : celles basées sur la région et celles basées sur le contour. Par la suite, nous avons fait une comparaison entre ces différentes méthodes en termes de performance, robustesse au bruit et coût mémoire.

Pour différencier nos pièces, nous nous sommes inspirés de la méthode Grid-based. Cette dernière repésente l'image par une matrice binaire. La comparaison de deux images s'effectue par le calcul du XOR entre leur représentation binaire. Dans notre cas, nous décrivons également une image par sa réprésentation binaire. Mais pour différencier deux images, nous utilisons une méthode de comparaison qui calcule de simples critères et les compare. Ces critères sont basés sur la région et sur le contour. Leur définition sera présentée au chapitre 4.

Cette méthode de comparaison a été développée pour se substituer au XOR qui est une méthode coûteuse en mémoire et en temps de calcul. Pour la validation de ce choix, une étude comparative en terme de coût mémoire et temps d'exécution entre le XOR et nos critères a été effectuée et sera présentée dans le chapitre 6.

Deuxième partie

Paramétrage de la Smart Surface

CHAPITRE 3

Introduction au projet Smart Surface

Sommaire

3.1 Introduction

Le projet Smart Surface a pour objectif la conception, le développement et le contrôle d'un système micro-robotique distribué pour le convoyage, le positionnement et le tri de micro-pièces à l'échelle mésoscopique[10].

La Smart Surface est une Surface de 3,5 cm×3,5 cm en silicium composée de plus de 500 micro-actionneurs, micro-capteurs et unités de traitement qui travaillent en collaboration pour le déplacement, le tri et le convoyage de micro-pièces. Le convoyage de ces micro-pièces est réalisé grâce à un flux d'air continu orientable et sans contact avec la micro-pièce.

3.2 Principe de la Smart Surface à base de MEMS

Un premier prototype à base de MEMS d'une Surface active a été conçu au LIMMS[11]/CNRS-IIS[12] de l'Université de Tokyo au sein du laboratoire du Pr. Fujita par Yves-André Chapuis, maître de conférences HDR à l'Université de Strasbourg et il a été mis à notre disposition au laboratoire FEMTO-ST[13] courant 2008.

[10] Du μm au mm.
[11] Laboratory of Integrated Micro Mechatronic Systems.
[12] Institute of Industrial Science.
[13] Département AS2M-Automatique et Systèmes Micro-Mécatronique.

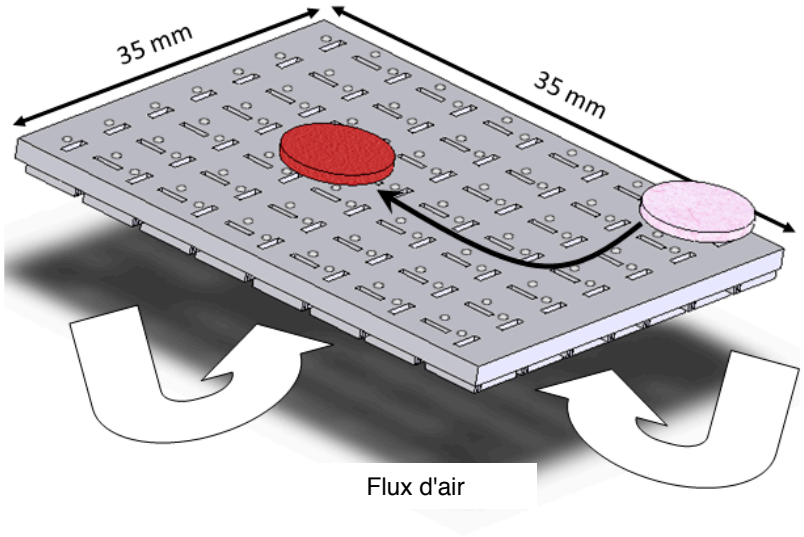


FIG. 3.1 – Surface active pneumatique constituée d'un réseau de micro-actionneurs.

La surface active est une matrice de 35×35 mm^2 qui contient 560 micro-actionneurs fluidiques, repartis sur la Surface (voir la figure 3.1). Le but de cette Surface est de réaliser un déplacement et un positionnement d'objet en distribué et sans contact. Le principal avantage de cette approche est que l'actionneur, qui est l'élément le plus fragile du dispositif, est protégé car il n'y a pas de frottement entre l'objet et l'actionneur contrairement aux systèmes de micro-manipulation à contact qui utilisent ce frottement pour déplacer les objets [36].

Le flux d'air arrive par les micro-valves situées au-dessous de la Smart Surface et génère à la surface un flux d'air bidirectionnel, qui va maintenir et déplacer l'objet en lévitation. L'orientation du jet d'air est contrôlée par un micro-actionneur électrostatique.

Un micro-actionneur se compose de deux couches (voir la figure 3.2). :

- une couche supérieure avec des orifices. Le flux d'air arrive par les micro-valves situées au-dessous du dispositif et passe à travers les orifices générant un flux d'air à la Surface (voir la figure 3.2(a,b,c)). Une force électrostatique est fournie grâce à deux électrodes alignées avec la micro-valve et fixées sur la couche supérieure ;
- une couche inférieure composée d'une valve mobile actionnée par un micro-actionneur électrostatique (voir la figure 3.2(d,e,f)). La micro-valve se déplace sous tension vers le coté gauche ou droit.

La fabrication de ces deux couches est réalisée avec du wafers SOI[14]. Elles se composent de deux substrats en silicium de même épaisseur 100 μm. Le micro-actionneur est de dimension 1×1 mm^2 [67].

Quand aucune tension n'est appliquée la micro-valve est bien centrée sous l'orifice permettant ainsi le passage d'un faible flux d'air qui va maintenir l'objet en lévitation comme le montre la figure 3.2(a). Par contre, si on applique une tension sur l'électrode de droite la micro valve va se

[14]Silicon On Insulator.

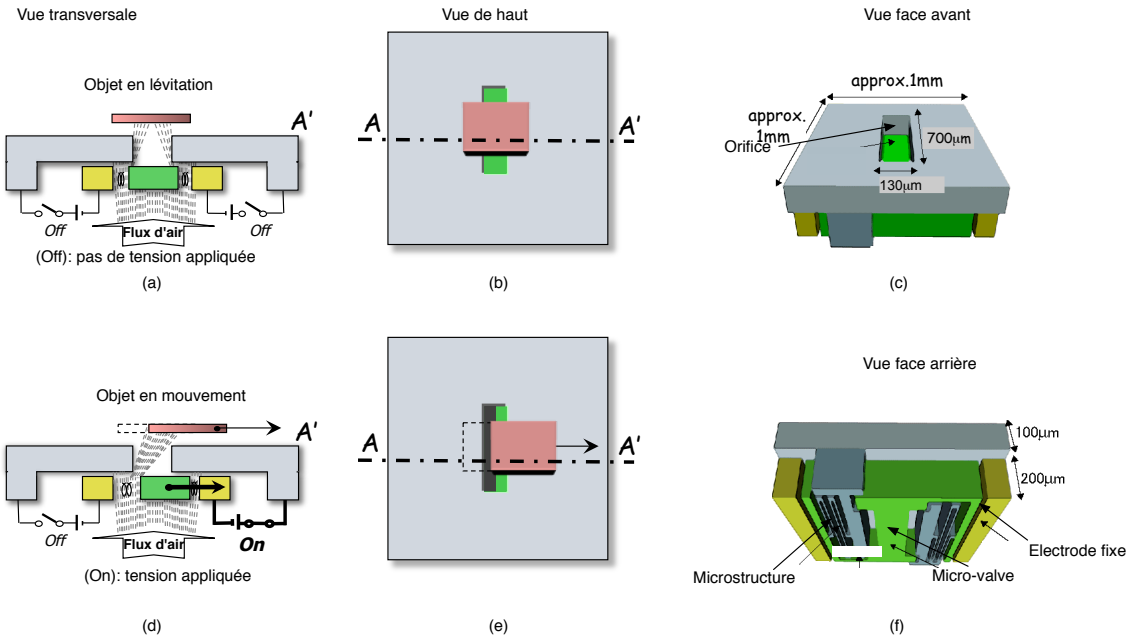


FIG. 3.2 – Image du fonctionnement d'un micro-actionneur : (a,d) vue transversale, (b,e) vue de haut, (c) vue de face avant et (f) vue de face arrière.

déplacer vers la droite, dégageant la partie gauche de l'orifice ce qui va augmenter le flux d'air vers la gauche. Donc cela va produire un jet d'air orientable qui va déplacer l'objet en lévitation vers la droite. Comme le montre la figure 3.2(d), on distingue deux types d'actionneurs, verticaux et horizontaux.

La figure 3.3 présente une maquette de la Smart Surface. Les micro-actionneurs sont disposés de manière alterner entre les micro-actionneurs verticaux et horizontaux pour le convoyage dans une direction quelconque d'un objet sur la Surface active.

Les différents systèmes de manipulateurs ont donné de bons résultats de convoyage. Mais leur utilisation dans l'industrie nécessite l'intégration d'intelligence et de capteurs. Une amélioration de ces systèmes est nécessaire.

L'objectif du projet Smart Surface est justement de concevoir un micro-manipulateur à base de MEMS *intelligent, distribué et autonome* composé d'une matrice de micro-modules afin de réaliser *un déplacement et un positionnement automatique des pièces*. Chaque micro-module sera composé d'un micro-actionneur, micro-capteur et d'une unité de traitement (voir la figure 3.4).

La coopération de ces micro-modules, grâce au réseau intégré, permettra à l'aide des micro-capteurs de différencier les pièces et de contrôler les micro-actionneurs pour déplacer et positionner avec précision les pièces sur la Smart Surface.

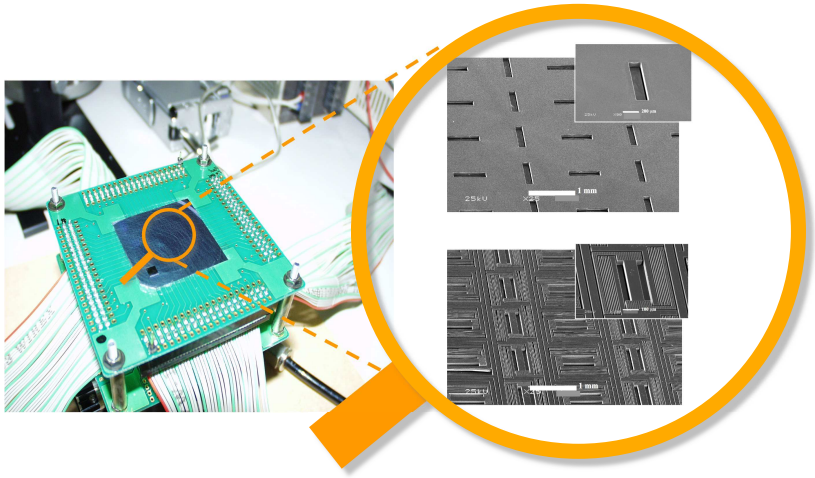


FIG. 3.3 – Image de la maquette de la Smart Surface actuelle.

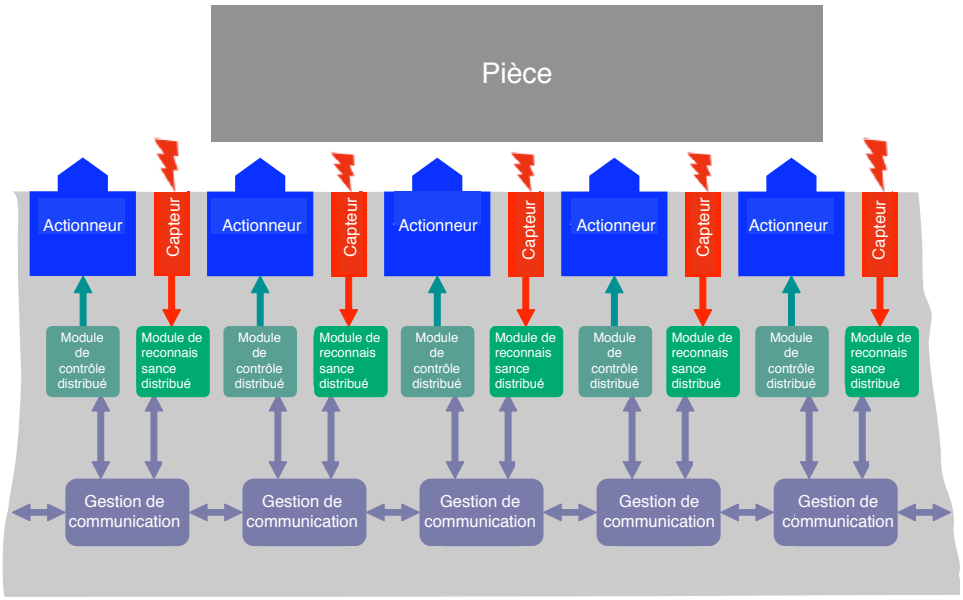


FIG. 3.4 – Concept de la Smart Surface.

3.3 Collaboration entre diverses disciplines

Pour la réalisation d'un tel dispositif autonome et distribué ce projet a été divisé en quatre sous-projets qui travaillent en collaboration.

SP1 : ce sous-projet est consacré à la *conception* et la *fabrication* de la matrice de capteurs/actionneurs ; le développement d'une structure micro-robotique distribuée appelée Smart Surface nécessite une approche de conception innovatrice, l'utilisation et le développement des technologies MEMS pour la réalisation d'une structure parallèle et une totale intégration (des composants).

SP2 : ce sous-projet est consacré à la reconnaissance (ou plus exactement à la *différenciation*) des micro-pièces. L'objectif de ce travail consiste à gérer les informations entre les modules de micro-capteurs/micro-actionneurs. Le micro-capteur retourne une information binaire sur la présence ou l'absence d'une micro-pièce. L'objectif est alors de déterminer quel type de pièce est au dessus de chaque micro-capteur en utilisant un algorithme distribué. Deux défis doivent être résolus afin de mener à bien cet objectif. Le premier est l'absence d'un point central de calcul. Ceci implique l'utilisation d'un algorithme distribué. Le deuxième est le manque de ressources de calcul sur les modules de micro-capteurs/micro-actionneurs. En effet, comme la Smart Surface travaille sur des micro-pièces, les modules doivent être aussi petits que possible, ce qui limite les ressources de calcul. Par conséquent, l'objectif est de proposer un algorithme de reconnaissance de formes complètement distribué, intégré dans la Smart Surface qui utilise le moins de ressources possible. Les modules de micro-capteurs/micro-actionneurs communiquent fréquemment pour échanger des informations sur la présence de la pièce et pour contrôler le déplacement de la micro-pièce. Il n'y a pas de point central sur la Smart Surface qui organise et route l'échange des messages, un module de micro-capteur/micro-actionneur est seulement lié à ses voisins. L'objectif est alors de concevoir un module dédié dans le but d'organiser le processus de communication globale. La distribution de la gestion de l'information à l'intérieur de la Smart Surface est basée sur le développement des algorithmes distribués qui devraient être mis en œuvre dans les unités de traitement à l'intérieur du micro-module.

SP3 : ce sous-projet est dédié au développement d'une *commande* distribuée et adaptative ; l'objectif de ce sous-projet est de piloter la Smart Surface pour *positionner* ou *déplacer* de manière automatique les objets placés dessus. Chaque micro-actionneur (valve) est capable de réaliser une poussée élémentaire sur un objet. La difficulté est de piloter efficacement les centaines d'actionneurs de la Smart Surface pour générer des déplacements contrôlés de l'objet. L'approche retenue ici est fondée sur l'apprentissage par renforcement dans le cadre des systèmes multi-agents, pour plus de détails voir [68].

SP4 : ce dernier traite les méthodes de conception et de *validation* ; la simulation et la validation à différents niveaux d'abstraction (du niveau comportement au niveau physique) sont nécessaires pour la conception correcte d'un système intégré aussi complexe.

La force de notre projet est la collaboration multidisciplinaire entre cinq laboratoires spécialisés dans leur domaine et plus de vingt chercheurs. Nous sommes responsables de la gestion de l'information dans la Smart Surface, plus précisément dans la différenciation distribuée de la pièce et l'infrastructure de communication (SP2).

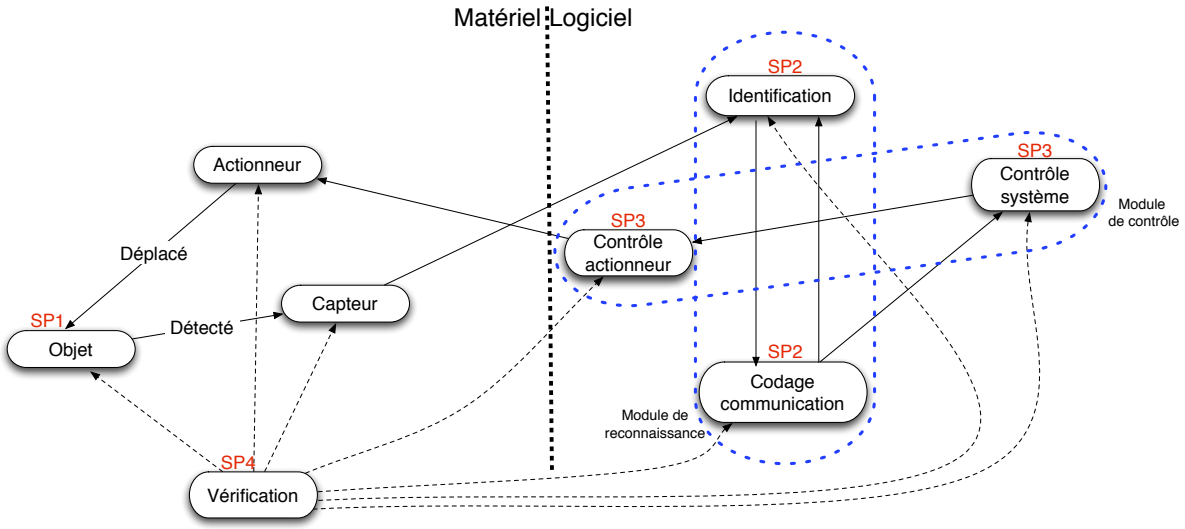


FIG. 3.5 – Exemple d'interaction entre les différents composants de la Smart Surface.

3.4 Scénario de collaboration

Le projet Smart Surface fait appel à une collaboration multidisciplinaire. Nous pouvons résumer l'interaction des différents sous-projets comme suit (voir la figure 3.5). Pour tout l'aspect matériel c'est le sous-projet 1 (SP1) qui a la tâche de la réalisation de la Smart Surface. Pour l'aspect fonctionnel, le scénario de collaboration peut être résumé comme suit : quand une pièce est sur la Smart Surface, elle va être détectée par les capteurs. Cette détection se fait par retour d'information binaire :

Capteur retourne 1 : présence de la pièce sur la Smart Surface (le capteur est couvert par la pièce) ;

Capteur retourne 0 : absence de la pièce sur la Smart Surface (le capteur n'est pas couvert par la pièce).

Les capteurs vont envoyer l'information au module d'identification. Grâce à ce dernier, les capteurs communiquent avec leurs voisins pour reconstruire la représentation binaire de la pièce qui est sur la Smart Surface. Une fois que la représentation binaire de l'image est construite, le module de reconnaissance va procéder à la différenciation de la pièce, cette différenciation est effectuée par le biais de calcul d'un ensemble de critères simples appelés *critères de différenciation* (nous reviendrons plus en détail sur la différenciation des pièces dans le chapitre 4), tout ceci est réalisé par le sous-projet 2 (SP2).

Une fois la pièce différenciée, le type de la pièce et la direction de son convoyage sont transmis au module de contrôle qui va envoyer une commande aux actionneurs. Ces derniers vont réagir en orientant le jet d'air dans des directions spécifiques afin de déplacer la pièce vers la direction souhaitée. Toute cette partie est réalisée par le sous-projet 3 (SP3).

Pour que cette collaboration multidisciplinaire fonctionne le mieux possible le sous-projet 4 (SP4) vérifie la coordination entre tous les sous-projets et valide les algorithmes utilisés.

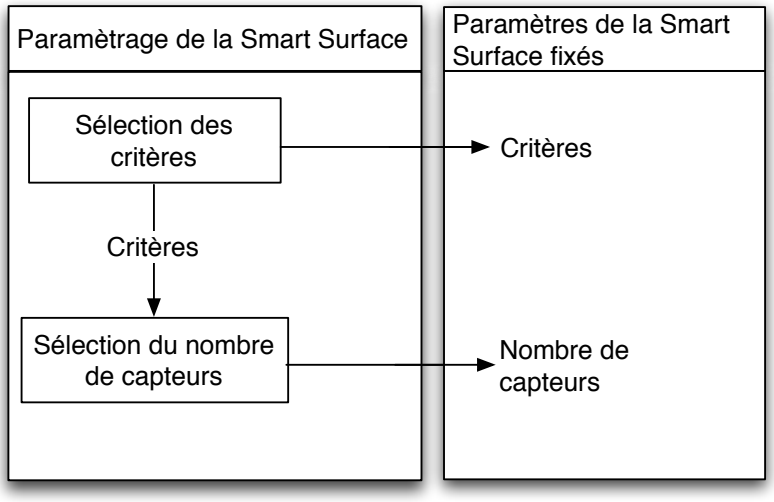


FIG. 3.6 – Vue globale du paramétrage de la Smart Surface.

3.5 Définition des paramètres de la Smart Surface

Le but du projet est de réaliser une Smart Surface pour le convoyage distribué des micro-pièces. Mais avant de commencer la phase de réalisation de la Smart Surface, il nous faut d'abord bien fixer ses paramètres, c'est une étape cruciale étant donné l'espace réduit de la Smart Surface. En effet, des paramètres mal fixés causeront des problèmes lors de son utilisation.

Le but de cette thèse est de bien fixer ces paramètres en répondant à une question très importante : comment fixer les paramètres de la Smart Surface pour un fonctionnement optimal ?

Pour répondre à cette question, il nous faut d'abord bien définir ces paramètres. Nous nous sommes donc intéressés à deux d'entre eux (voir figure 3.6) :

1. la différenciation des pièces se fait grâce au calcul de simples critères de différenciation. Un des paramètres nécessaire au bon fonctionnement de la Smart Surface est donc de trouver les meilleurs critères qui permettent de différencier les pièces. En d'autres termes il nous faut répondre à la question suivante : quels sont les meilleurs critères à utiliser pour une bonne différenciation des pièces ?
2. la détection des pièces s'effectue par les capteurs, qui ont pour fonction de détecter la présence ou l'absence de pièces sur la Smart Surface. De plus, étant donné l'espace réduit sur la Smart Surface, il nous faut déterminer comme deuxième paramètre le nombre de capteurs nécessaires pour un fonctionnement optimal, c'est-à-dire de répondre à la question suivante : quel est le nombre de capteurs nécessaire pour un bon fonctionnement de la Smart Surface ?

3.6 Conclusion

La Smart Surface est une Surface autonome composée de micro-actionneurs, micro-capteurs et d'unités de traitement qui travaillent en collaboration pour le convoyage de micro-pièces.

La construction de cette surface nécessite d'impliquer plusieurs acteurs de diverses disciplines qui collaborent dans le but de résoudre plusieurs problématiques. Ces problématiques impliquent notament la gestion de différents verrous scientifiques dans chaque discipline.

Dans cette thèse nous nous sommes particulièrement intéressés au fonctionnement optimal de la Smart Surface qui nécessite un paramètrage rigoureux. Ces paramètres consistent à déterminer les critères de différenciation des pièces et le nombre de capteurs nécessaires pour un fonctionnement optimal de la Smart Surface (nous y reviendrons plus en détail dans les chapitres 4 et 5).

CHAPITRE 4

Sélection des critères

Sommaire

4.1 Introduction

Avant de réaliser la Smart Surface, il est primordial de bien définir ses paramètres afin d'en assurer un fonctionnement optimal.

Ce chapitre est dédié à déterminer les critères nécessaires pour différencier les pièces.

Afin de pouvoir atteindre notre objectif, il nous faut :

1. un ensemble de critères : ces critères doivent être simples à implémenter ;
2. un ensemble de pièces : ces pièces représentent un jeu de tests pour tester nos critères.

Dans ce qui suit, nous allons présenter un outil qui permet de comparer les critères de différenciation des pièces. Il est basé sur une génération *exhaustive* des pièces. Le but de cet outil est de répondre à la question suivante : *quels sont les meilleurs critères de différenciation pour différencier trois pièces quelconques de taille inférieure à* 500×500 μm^2 ? Les pièces peuvent couvrir au maximum 4×4 capteurs de la Smart Surface. Dans la suite nous donnerons la taille d'une pièce grâce au nombre de capteurs qu'elle couvre soit 2×2, 3×3 ou 4×4.

Nous différencions seulement les groupes de trois pièces, car étant donnée l'application pratique de la Smart Surface qui consiste à trier des pièces, on suppose qu'on aura des pièces qui arrivent d'un côté de la Smart Surface (exemple côté bas) une fois ces pièces différenciées elles seront déplacées vers un des côtés de la Smart Surface (Haut, gauche ou droit).

4.2 Outil de sélection des critères

Afin de tester la série de critères que nous avons définis, nous avons développé l'outil ECO (*Exhaustive Comparison Framework*) [69]. Cet outil reçoit en entrée (voir figure 4.1) :

1. l'ensemble des critères de différenciation à comparer ;
2. la taille maximale de la pièce P_{taille} ;
3. le *nombre* de pièces à différencier G_{taille}.

Cet outil va ensuite générer en sortie :

1. toutes les pièces uniques dont la taille est inférieure ou égale à P_{taille} ;
2. un arbre de comparaison des critères.

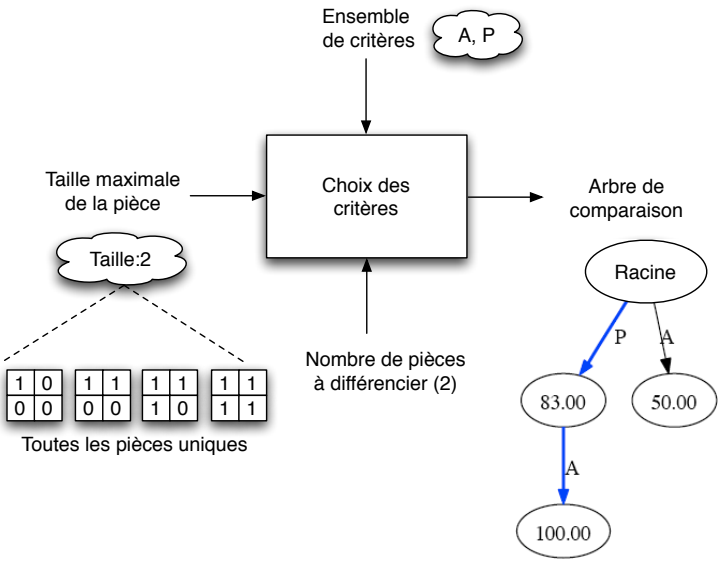


FIG. 4.1 – Vue globale de l'outil.

4.3 Hypothèses de travail

À ce stade de la thèse, nous nous sommes d'abord intéressés à vérifier la faisabilité du problème, en répondant à la question suivante : est-il possible de différencier n'importe quelle pièce de taille 3×3 ou 4×4 qui se trouve sur la Smart Surface ? En d'autres termes, existe-t-il des critères de différenciation qui arrivent à 100 % de différenciation, permettant de différencier à coup sûr une pièce parmi toutes les autres ? Partant de cette large hypothèse, nous avons fixé quelques contraintes qui seront assouplies plus loin (voir chapitre 5). Ces contraintes sont :

- les pièces peuvent être tournées par multiples de 90° seulement ;
- il n'y a pas d'erreur de communications entre capteurs ;
- il n'y a aucune défaillance de capteurs ;
- nous travaillons avec des *familles de pièces*. Nous définissons une famille comme l'ensemble des pièces qui ont la même forme. Par exemple, la lettre typographique L et L (avec et sans empattement) ont la même forme car leurs différences sont trop petites pour être détectées par nos capteurs.

4.4 Quelques définitions

4.4.1 Définition d'un critère de différenciation

Un critère de différenciation se définit comme étant une fonction mathématique qui associe à une pièce donnée, une valeur (voir figure 4.2).

Comme l'espace est réduit dans la Smart Surface et que tous les traitements informatiques s'effectuent dans la Smart Surface, les critères doivent être simples à calculer afin d'avoir un temps de réponse optimal et occuper le moins d'espace possible en mémoire. Dans le chapitre 6, nous reviendrons plus en détail sur les critères que nous avons définis.

FIG. 4.2 – Définition d'un critère.

Ces critères de différenciation sont utilisés pour différencier les pièces entre elles. Cela s'effectue en calculant la valeur d'un critère associé à deux pièces différentes, puis en effectuant une simple comparaison entre les valeurs obtenues. Mathématiquement, cela revient à :
Soit C_1 un critère donné et P_1, P_2 deux pièces quelconques. $C_1(P_1) = V_1$, la valeur du critère C_1 appliqué à la pièce P_1. $C_1(P_2) = V_2$, la valeur du critère C_1 appliqué à la pièce P_2. Alors :

$$\begin{cases} V_1 = V_2 \Rightarrow P_1 \text{ et } P_2 \text{ sont identiques} \\ V_1 \neq V_2 \Rightarrow P_1 \text{ et } P_2 \text{ sont différentes} \end{cases}$$

Si un critère n'est pas suffisant pour différencier une pièce, alors il est possible d'utiliser une combinaison de critères de différenciation. Par exemple, si $C = \{A, S, P\}$ est l'ensemble des cri-

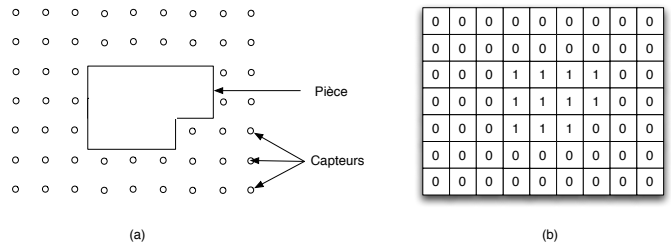
tères de différenciation, les combinaisons générées sont $CC = \{\{A\}, \{S\}, \{P\}, \{AS\}, \{AP\}, \{SP\}, \{ASP\}\}$.

Dans la section 4.6 nous reviendrons plus en détail sur le calcul de la différenciation entre les pièces selon un critère ou une combinaison de critères.

4.4.2 Définition d'une pièce

Nous nous intéressons seulement à la *forme* d'une pièce, et donc nous ne prenons pas en compte d'autres caractéristiques telles que la couleur, le poids, etc.

Quand une pièce est sur la Smart Surface (voir figure 4.3(a)), elle est détectée par des capteurs qui retournent une information binaire. Si la pièce est au-dessus du capteur, celui-ci renvoie 1, sinon 0 (voir figure 4.3(b)).

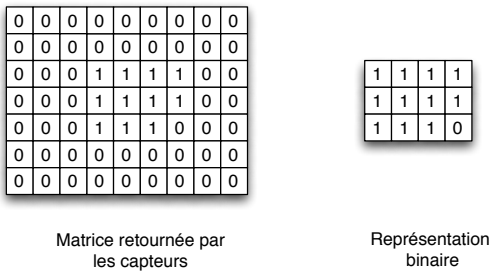


0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	0	0
0	0	0	1	1	1	1	0	0
0	0	0	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

FIG. 4.3 – (a) Exemple d'une pièce posée sur un réseau de capteurs. (b) Représentation binaire de la pièce fournie par les capteurs.

4.4.3 Définition du masque

En réalité la représentation binaire d'une pièce est le plus petit carré de capteurs dans lequel la pièce est contenue. Pour l'obtenir, on calcule le *masque* de la représentation binaire qui se définit comme étant une matrice générée à partir de la matrice initiale après avoir supprimé les colonnes et les lignes contenant que des 0 (voir la figure 4.4). Pour notre outil une pièce est une matrice binaire (représentation binaire de la pièce).



0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	1	0	0
0	0	0	1	1	1	1	0	0
0	0	0	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0

1	1	1	1
1	1	1	1
1	1	1	0

FIG. 4.4 – Calcul d'une représentation binaire.

4.4.4 Définition d'une composante connexe

Voisinage

Le voisinage d'un point varie selon le type de connexité utilisé. Il en existe deux basés sur un maillage carré : 4-connexité et 8-connexité. Selon la 4-connexité, chaque point a 4 voisins (voir figure 4.5(a)). Selon la 8-connexité, chaque point a 8 voisins (voir figure 4.5(b)).

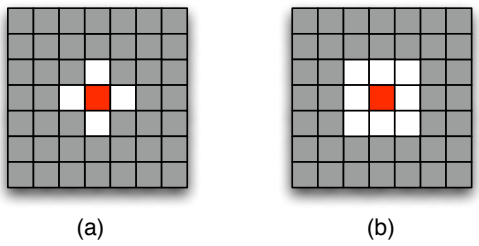


FIG. 4.5 – Les voisins d'une pièce pour : (a) 4-connexité, (b) 8-connexité.

Chemin connexe

On appelle chemin connexe entre deux points P_0 et P_n, une suite de points $(P_0, P_1, ..., P_n)$ tels que P_{i-1} et P_i soient adjacents (voir figure 4.6).

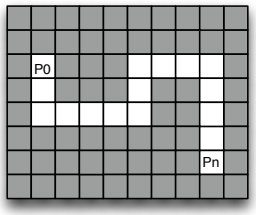


FIG. 4.6 – Exemple d'un chemin connexe.

Composante connexe

Soit P et Q deux points de l'image et soit S un sous-ensemble de points contenant P et Q. On dit que P et Q sont connectés, si et seulement si, il existe un chemin connexe inclus dans S entre P et Q. S est une composante connexe si quelques soient P et Q appartenant à S, ils sont connectés. La détection d'une composante connexe dépend du type de connexité utilisé. Exemple dans la figure 4.7(a) la pièce est connexe au sens de la 4-connexité et de la 8-connexité. La pièce de la figure 4.7(b) n'est pas connexe au sens de la 4-connexité mais connexe au sens de la 8-connexité. Enfin dans la figure 4.7(d) la pièce n'est pas connexe ni au sens de la 4-connexité ni au sens de la 8-connexité.

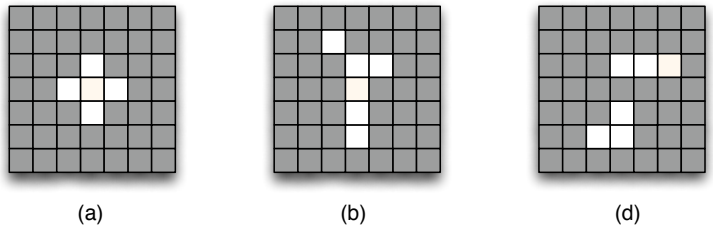


FIG. 4.7 – (a) Composantes connexes selon 4-connexité. (b) Composantes non connexes selon 4-connexité mais connexes selon 8-connexité. (c) Composantes non connexes selon la 4-connexité et la 8-connexité.

4.5 Génération exhaustive des représentations binaires des pièces

À partir d'une taille maximale de la pièce P_{taille}, l'outil ECO va d'abord générer de manière exhaustive toutes les représentations binaires. Ceci s'effectue comme suit :

- une pièce va être représentée dans un carré de taille maximale $P_{taille} \times P_{taille}$;
- ce carré va contenir des informations retournées par des capteurs binaires, donc il contiendra des 0 ou 1. Il existe en réalité $2^{P_{taille} \times P_{taille}}-1$ représentations binaires possibles[15].
- pour avoir toutes les représentations binaires des pièces de taille maximale P_{taille} il suffit de déterminer toutes les représentations binaires des nombres de 1 à $2^{P_{taille} \times P_{taille}}-1$ dans une matrice de taille $P_{taille} \times P_{taille}$.

Si on prend comme exemple une taille maximale de la pièce égale à 2 ($P_{taille} = 2$). Notre outil va générer toutes les représentations binaires des pièces de taille maximale $P_{taille} \times P_{taille} = 2 \times 2 = 4$. En fait, cela revient à déterminer $2^{P_{taille} \times P_{taille}}-1 = 2^4-1 = 15$ matrices (voir figure 4.8).

En examinant l'ensemble des représentations binaires obtenues, nous remarquons qu'il existe des pièces qui apparaissent plusieurs fois. Pour supprimer les pièces en double, nous allons appliquer sur cet ensemble différents traitements afin de le réduire à un ensemble de représentations binaires uniques.

4.5.1 Filtrage par connexité

Dans l'ensemble des représentations binaires générées beaucoup d'entres elles ne sont pas *connexes*[16]. En réalité, il y a deux pièces au lieu d'une (voir figure 4.9). Le contrôle de la connexité se fait grâce à un algorithme de détection de composantes connexes. Cet algorithme peut être décrit comme suit :

1: Soit NbCC, le nombre de composantes connexes initialisé à 1.
2: **repeat**
3: parcours de l'image
4: **until** pixel blanc (ou noir) p non marqué
5: Empiler p dans la pile et le marquer par *
6: **while** la pile est non vide **do**

[15]La représentation binaire du nombre 0 n'est pas prise en compte.
[16]Nous considérons la 4-connexité.

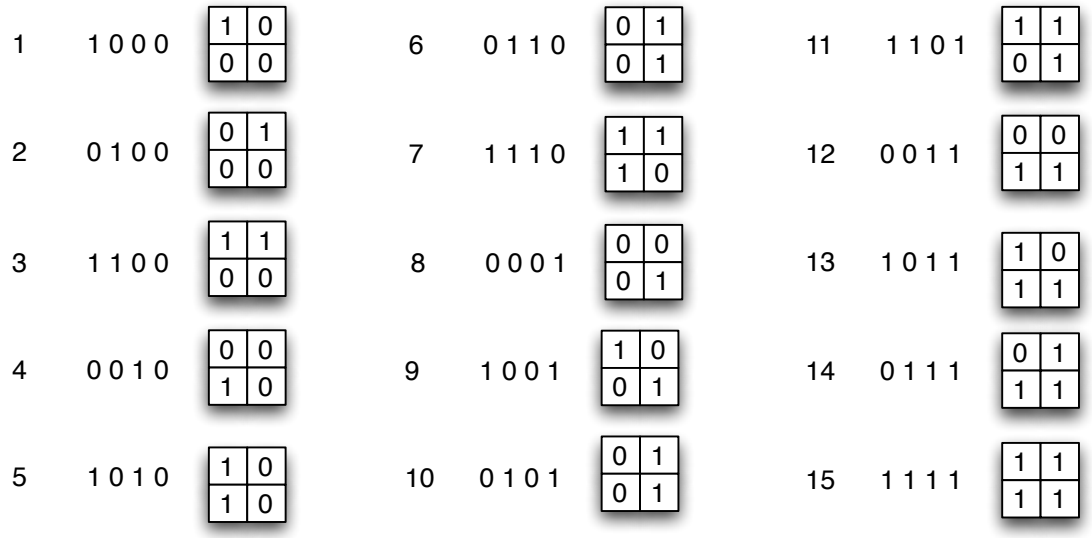


FIG. 4.8 – Exemple de génération exhaustive de toutes les représentations binaires des pièces de taille maximale $P_{taille} = 2$.

FIG. 4.9 – Exemple de représentation binaire non connexe, qui montre deux pièces en 4-connexité.

7: p = Dépiler le sommet de la pile
8: Empiler les voisins de p non marqués (selon la 4 ou 8 connexités) et les marquer par *
9: Marquer le pixel p par NbrCC
10: **end while**
11: **if** il existe dans l'image un pixel noir non marqué **then**
12: la pièce n'est pas connexe
13: **else**
14: la pièce est connexe
15: **end if**

A la fin de l'algorithme, si tous les pixels sont marqués alors la pièce est connexe, sinon la pièce n'est pas connexe (voir figure 4.10).

4.5.2 Filtrage par translation

Nous remarquons aussi que sur l'ensemble des représentations binaires générées (voir figure 4.8), il y en a plusieurs qui sont redondantes. En réalité c'est la même représentation binaire qui est juste translatée (voir figure 4.11).

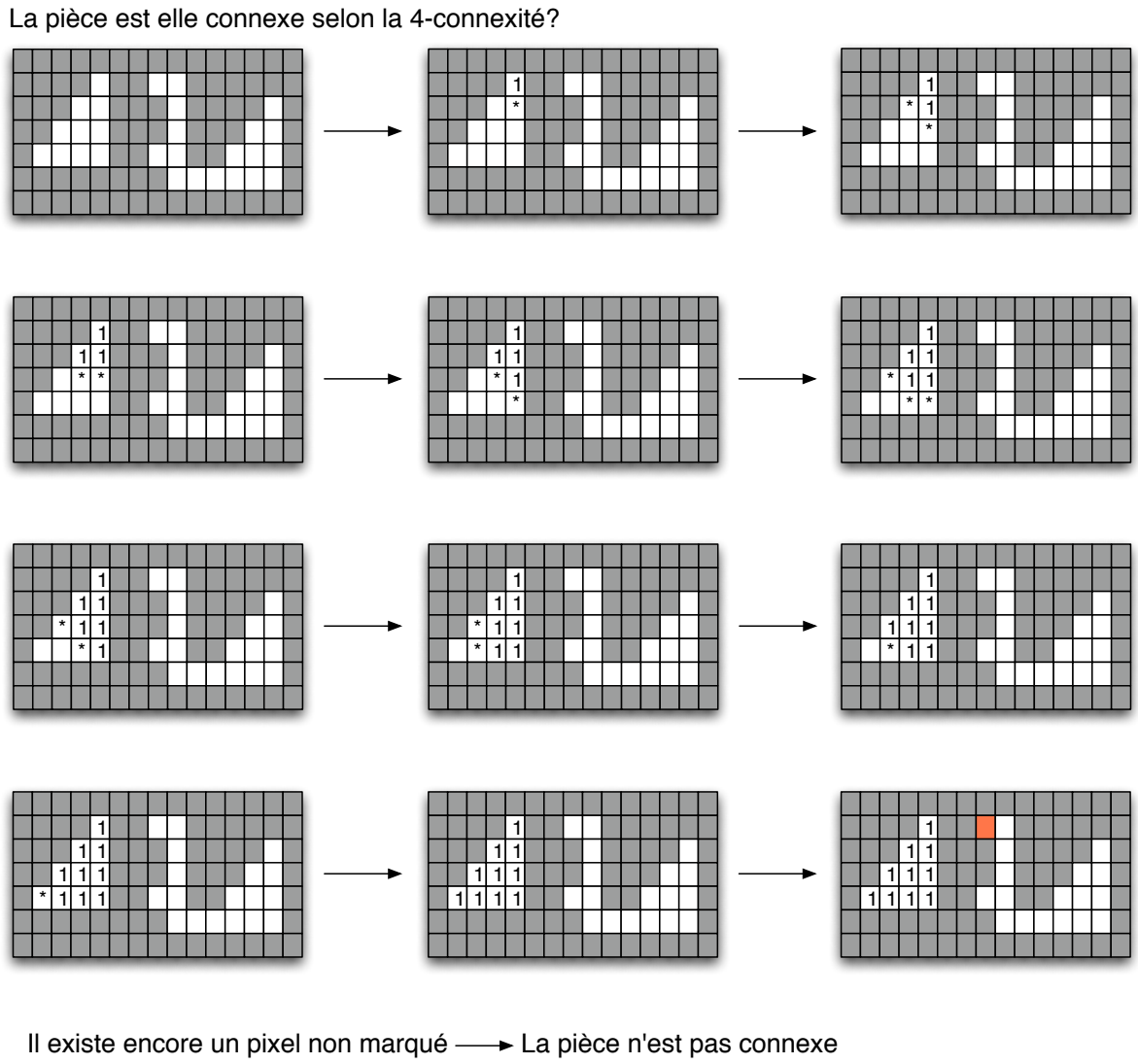


FIG. 4.10 – Étiquetage composante connexe.

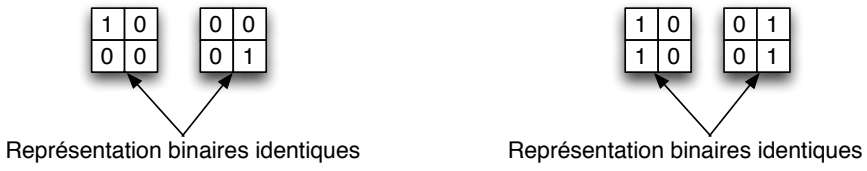


FIG. 4.11 – Exemple de représentations binaires identiques par translation.

Pour détecter qu'une représentation binaire est en réalité la translation d'une autre, il suffit de comparer leur masque (voir la figure 4.12).

L'algorithme du calcul du masque est :

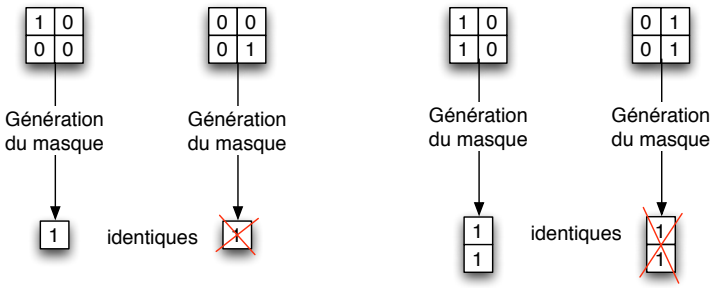


FIG. 4.12 – Exemple de génération de masque des représentations binaires.

1: Masque$_{imin} = P_{taille}$, Masque$_{jmin} = P_{taille}$, Masque$_{imax} = 0$, Masque$_{jmax} = 0$
2: **for** chaque i ∈1, 2 , 3,..., P_{taille} **do**
3: **for** chaque j ∈1, 2 , 3,..., P_{taille} **do**
4: **if** $R_{i,j} = 1$ élement de la représentation binaire **then**
5: **if** i < Masque$_{imin}$ **then**
6: Masque$_{imin} = i$
7: **end if**
8: **if** i > Masque$_{imax}$ **then**
9: Masque$_{imax} = i$
10: **end if**
11: **if** j < Masque$_{jmin}$ **then**
12: Masque$_{jmin} = j$
13: **end if**
14: **if** i > Masque$_{jmax}$ **then**
15: Masque$_{imax} = j$
16: **end if**
17: **end if**
18: **end for**
19: **end for**

Le résultat de cet algorithme permet de déterminer le plus petit rectangle contenant la pièce. Le rectangle est représenté par les deux points P_1(Masque$_{imin}$, Masque$_{jmin}$) et P_2(Masque$_{imax}$, Masque$_{jmax}$).

4.5.3 Filtrage par rotation et miroir

Parmi l'ensemble des représentations binaires générées (voir figure 4.8), nous remarquons aussi qu'il y a des représentations binaires identiques mais tournées à différents degrés de rotation[17] : 90°, 180° et 270°. Nous remarquons également que certaines représentations binaires sont des miroirs pour d'autres représentations binaires.

Au lieu de générer toutes les représentations binaires et ensuite de les comparer deux à deux,

[17]Le pas de rotation est fixé à 90°, il sera assoupli dans le chapitre 5.

afin de supprimer les pièces identiques par rotations (seules les rotations à 90°, 180° et 270° sont considérées, voir figure 4.14) et miroirs (voir 4.13), on applique l'algorithme de filtrage des rotations et des miroirs qui permet de générer un ensemble de représentations des pièces uniques.

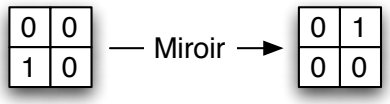


FIG. 4.13 – Exemple de miroir des représentations binaires des pièces.

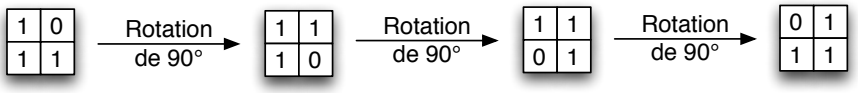


FIG. 4.14 – Exemple de rotation à 90° d'une représentation binaire d'une pièce.

L'algorithme de filtrage des rotations et des miroirs est :

1: RBU l'ensemble des représentations binaires uniques
2: RotAndMirRBU : l'ensemble des rotations et miroirs des représentations binaires uniques
3: RBU ={⊘}
4: RotAndMirRBU ={⊘}
5: **for** chaque représentation binaire RB générée **do**
6: L : l'ensemble des rotations et miroirs de RB
7: **if** RB $\notin$ RBU et L $\cap$ RBU = $\oslash$ et L $\cap$ RotAndMirRBU= $\oslash$ **then**
8: ajouter RB dans RBU
9: ajouter L dans RotAndMirRBU
10: **end if**
11: **end for**

A la fin de cet algorithme on obtient un ensemble de représentations binaires uniques du point de vue de la rotation et des miroirs (voir figure 4.15).

4.5.4 Génération de tous les groupes de trois représentations binaires

Dans le cadre applicatif de la Smart Surface qui stipule qu'il existe potentiellement trois sorties pour les pièces, nous avons choisi de regrouper les représentations binaires des pièces uniques par groupes de trois pièces. À partir des représentations binaires des pièces uniques obtenues (voir figure 4.15), tous les groupes de trois représentations binaires des pièces sont générés (voir figure 4.16).

Étant donné que, dans notre projet, nous travaillons avec des pièces à l'échelle mésoscopique, qui recouvriraient 3×3 ou 4×4 capteurs sur la Smart Surface, nous avons généré toutes les

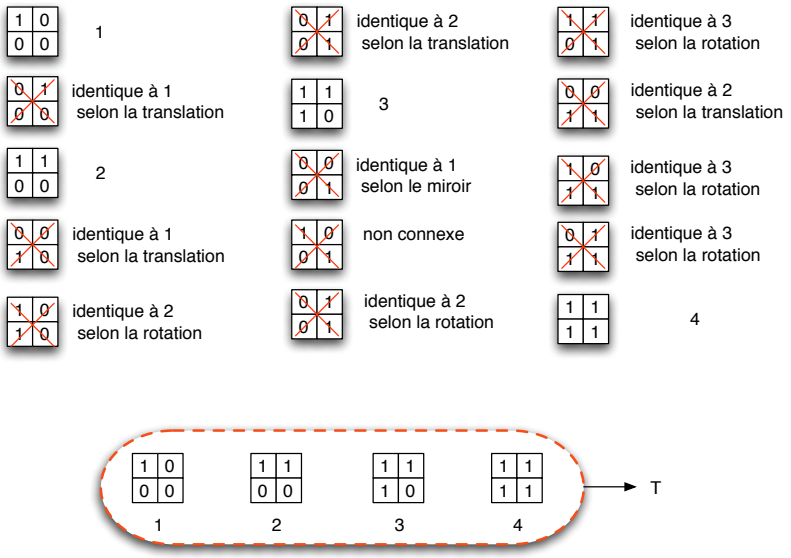


FIG. 4.15 – Représentations binaires uniques obtenues de taille $P_{taille} = 2$.

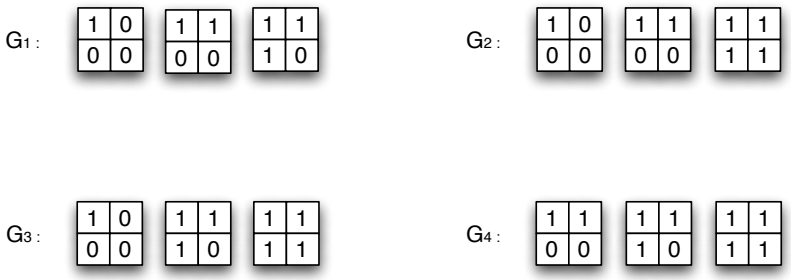


FIG. 4.16 – Tous les groupes de trois représentations binaires générés à partir des représentations binaires uniques de taille $P_{taille} = 2$.

représentations binaires uniques des pièces de taille $P_{taille} = 3$ et $P_{taille} = 4$. Le tableau 4.1 représente un récapitulatif du nombre total T de représentations binaires des pièces uniques obtenues pour $P_{taille} = 3$ et $P_{taille} = 4$.

TAB. 4.1 – Le nombre de représentations binaires uniques des pièces et le nombre de groupes de trois représentations binaires des pièces générées.

Taille maximale de la pièce	Nombre de pièces générées	Nombre de pièces uniques (T)	Nombre de groupes
3×3	512	35	$C^3_{35} = 6545$
4×4	65536	1280	$C^3_{1280} = 348706560$

4.6 Différenciation des pièces

4.6.1 Différenciation des pièces par un critère

Pour chaque groupe G_i, pour différencier les représentations binaires la matrice de différenciation D selon le critère C_j est calculée (voir figure 4.17). La matrice D sert à comparer les représentations binaires des pièces deux à deux. Affecter 1 dans la matrice de différenciation D selon le critère C_j signifie qu'avec le critère C_j on est capable de différencier les deux représentations binaires des pièces entre elles. Par contre affecter 0, dans la matrice de différenciation D selon le critère C_j signifie que les deux représentations binaires des deux pièces ne peuvent pas être différenciées en utilisant le critère C_j. La matrice D est triangulaire supérieure car comparer entre P_1, P_2 ou P_2, P_1 donne exactement le même résultat. Aussi est-il inutile de comparer P_1 à P_1 (voir figure 4.17).

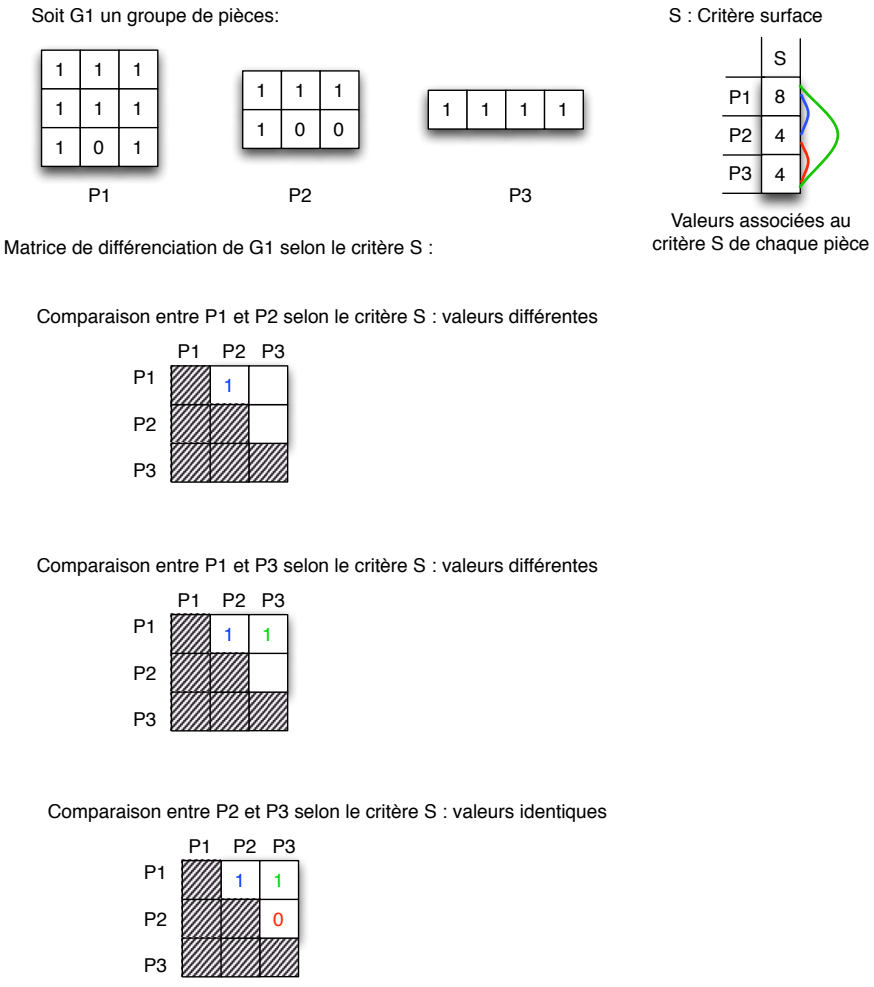


FIG. 4.17 – Exemple de calcul de la matrice de différenciation D selon le critère $C_j = \{S\}$.

4.6.2 Différenciation des pièces par une combinaison de critères

Dans le cas d'une combinaison de plusieurs critères CC_j, la matrice de différenciation D selon la combinaison de critères CC_j est obtenue en calculant le OU logique entre les matrices de différenciation de chaque critère appartenant à la combinaison CC_j (voir figure 4.18).

$$D_{G_i,CC_j} = \cup_{k \in CC_j} D_{G_i,C_k}, \text{ avec } CC_j = \{C_1, C_2, ..., C_m\} \tag{4.1}$$

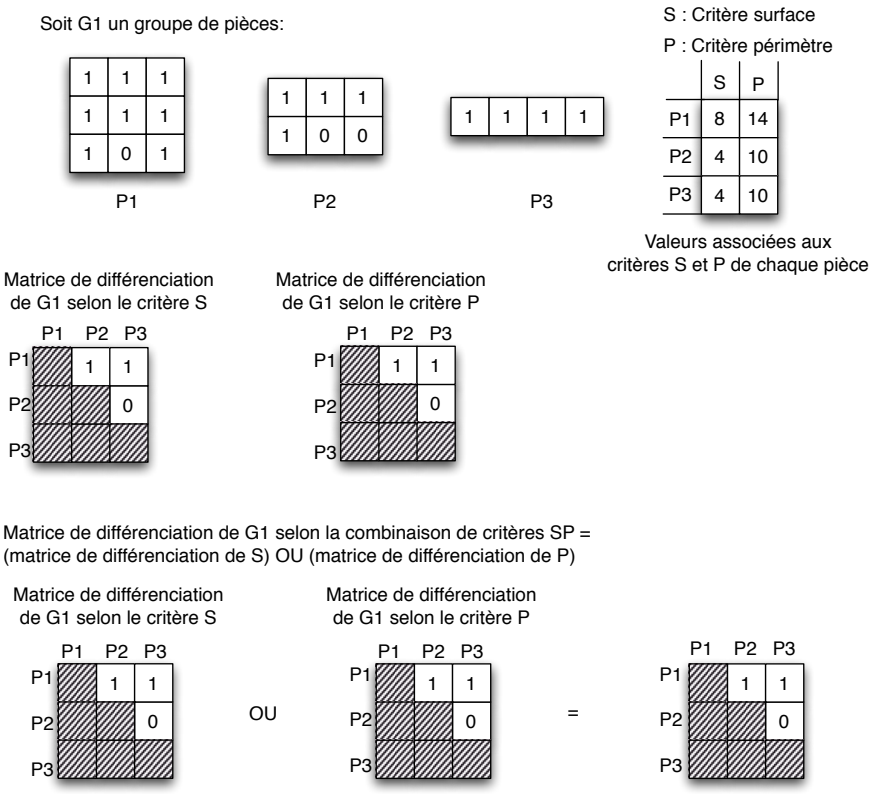


FIG. 4.18 – Exemple de calcul de la matrice de différenciation D selon la combinaison de critères $CC_j = \{S, P\}$.

4.6.3 Résultats de la différenciation des pièces

Résultats de différenciation pour un groupe

Si la matrice D résultante ne contient que des 1, les représentations binaires des pièces sont dites différenciées grâce à cette combinaison de critères. Cela signifie, que quelque soit la représentation binaire de pièce appartenant à ce groupe, on arrive à la différencier.

$$\begin{gathered} D_{G_i,C_j}(k,l) = 1, \forall k,l, k \neq l \in G_i \\ \Leftrightarrow \text{le critère } C_j \text{ différencie toutes les pièces du groupe } G_i \end{gathered} \tag{4.2}$$

La figure 4.19 est un exemple de calcul de la matrice de différenciation de $G_1 = \{P_1, P_2, P_3\}$ pour la combinaison de critères $CC_1 = \{ASP\}$. Comme dans la matrice de différenciation D résultante il n'existe que des 1, cela signifie que pour le groupe G_1 on arrive à différencier toutes les représentation binaires des pièces entre elles en utilisant la combinaison de critères $CC_1 = \{ASP\}$.

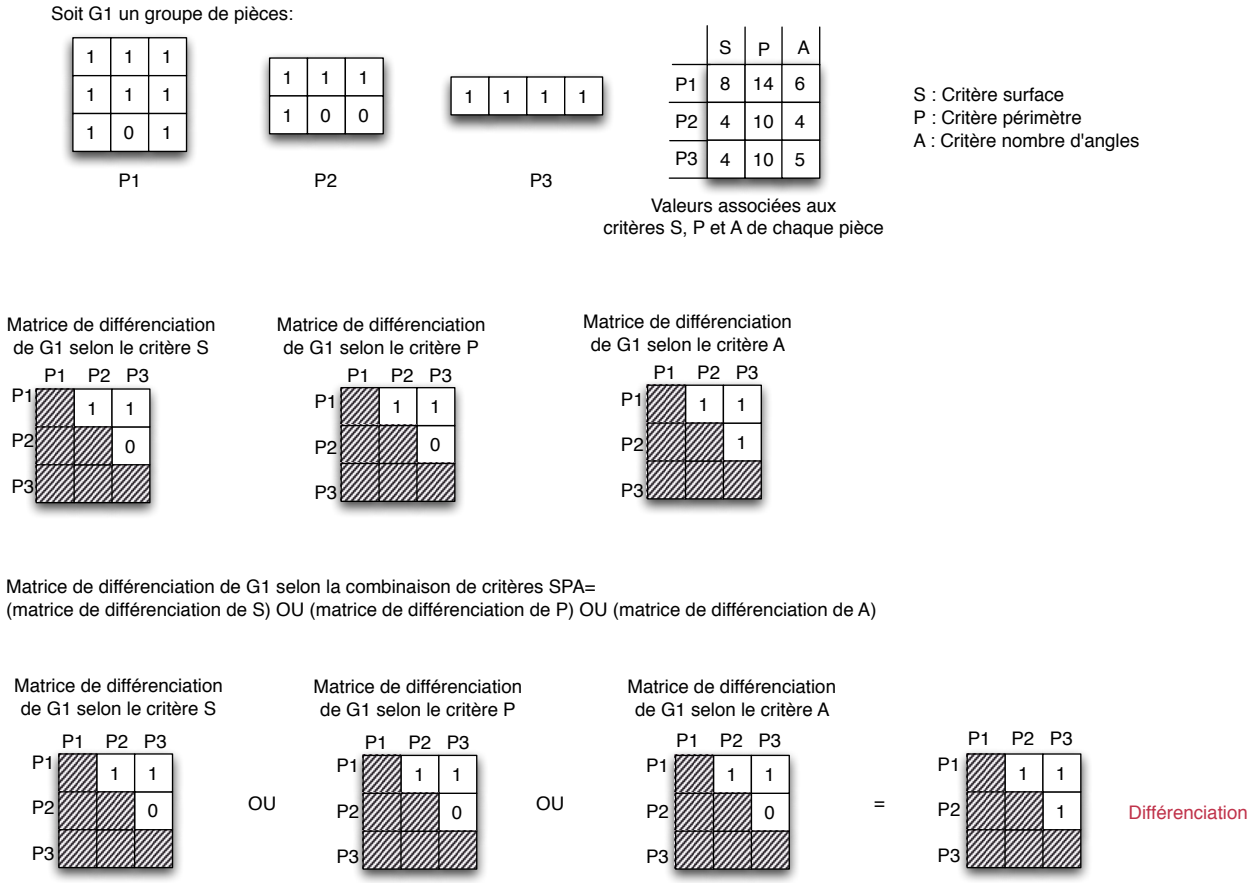


FIG. 4.19 – Exemple de calcul de la matrice de différenciation selon la combinaison de critères $CC_j = \{S, P, A\}$.

Résultats de différenciation pour tous les groupes

Notre travail consiste à trouver les critères de différentiation qui arrivent à différencier tous les groupes des représentations binaires des pièces. Il nous faut alors calculer le taux de différenciation t pour chaque critère C ou combinaison de critères CC_j. Ce taux représente le pourcentage des matrices D_{G_i,CC_j} qui contiennent que des 1. Soit NbrTotalGroupe le nombre total de groupe de trois représentations binaires générés. L'algorithme du calcul du taux de différenciation est :

1: $NbrMat = 0$
2: **for** chaque groupe de représentation binaire des pièces générées **do**

3: calcul de la matrice de différenciation selon un critère C_j ou une combinaison de critères CC_j
4: **if** D_{G_i,CC_j} ne contient que des 1 **then**
5: le nombre de groupes différenciés est $NbrMat = NbrMat + 1$
6: **end if**
7: **end for**
8: le taux de différenciation est $t = \frac{NbrMat \cdot 100}{NbrTotalGroupe}$

Une différenciation est dite totale (ou différenciation à 100%) si *toutes* les représentations binaires des pièces sont différenciées pour *tous* les groupes possibles.

Le défi majeur est de trouver une combinaison de critères qui arrive à une différenciation totale (pour tous les groupes de représentations binaires des pièces).

Le processus de différenciation peut être décrit par l'algorithme suivant : Soit n le nombre de pièces qui doivent être différenciées. L'outil génère tous les groupes de n représentations binaires des pièces uniques. Si T est le nombre total de représentations binaires des pièces uniques, il y aura NbrTotalGroupe=C_T^n groupes. L'algorithme de différenciation est :

1: **for** chaque CC_j = sous ensemble des critères $\{C_1, C_2, ..., C_m\}$ **do**
2: **for** chaque groupe G_i sous ensemble de n éléments dans T **do**
3: **if** CC_j est un seul critère **then**
4: remplir la matrice de différenciation D_{G_i,CC_j}
5: **else**
6: $\{CC_j$ est une combinaison de critères$\}$
7: **for** chaque C_k dans CC_j **do**
8: remplir la matrice de différenciation $D_{G_i,CC_j} = OU D_{G_i,C_k}$
9: **end for**
10: **end if**
11: **if** D_{G_i,CC_j} ne contient que des 1 **then**
12: le nombre de groupes différenciés est $NbrMat = NbrMat + 1$
13: **end if**
14: **end for**
15: calcul du taux de différenciation $t = \frac{NbrMat \cdot 100}{NbrTotalGroupe}$
16: **end for**
17: construire l'arbre de comparaison

4.7 Construction de l'arbre de différenciation et de coût

Afin de visualiser l'importance de chaque critère, les résultats de différenciation pour chaque critère sont donnés sous la forme d'un arbre appelé *arbre de comparaison* (voir figure 4.20). Cet arbre, de type général, a été simplifié car le chemin XY qui est le même que YX a été supprimé. Chaque nœud représente le taux de différenciation exprimé en pourcentage. Un arc représente un critère de différenciation. Une branche de l'arbre (un chemin de la racine vers une feuille) représente la combinaison de critères utilisée pour la différenciation. Le taux est obtenu en utilisant tous les critères du chemin à partir de la racine de l'arbre. Enfin, un coût (temps d'exécution, mémoire utilisée etc.) peut être associé à chaque branche (voir figure 6.18).

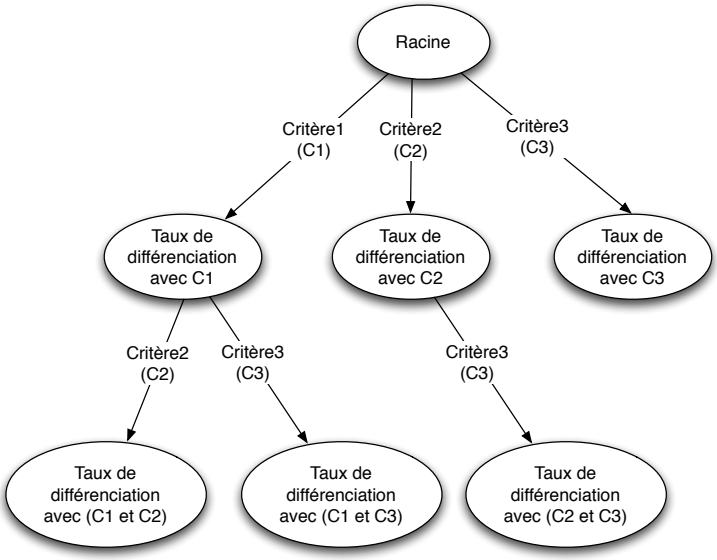


FIG. 4.20 – Exemple d'arbre de comparaison généré.

4.7.1 Coût mémoire

La Smart Surface a une mémoire limitée en raison de son intégration à l'échelle micro. Cette section formalise la mémoire nécessaire pour calculer chaque critère. Tout d'abord, la mémoire nécessaire pour une cellule (micro-module) utilisant un seul critère est calculée. Chaque cellule doit connaître tous les modèles M_j (pièces connues par la Smart Surface). Soit n la taille du groupe (les pièces du groupe qui doivent être différenciées), par exemple $n = 3$ et soit $m(C_i, M_j)$ la mémoire nécessaire (en bits) pour stocker la valeur du critère C_i de la pièce du modèle M_j. Par conséquent, tous les modèles nécessitent $M_1 = \sum_{j=1}^{n} m(C_i, M_j)$ bits de mémoire.

Quand une pièce P est sur la Smart Surface, elle est d'abord reconstruite par chaque cellule. L'algorithme de reconstruction est :

1: **for** chaque image **do**
2: phase de communication
3: phase de calcul
4: **end for**

Dans la phase de communication, les capteurs s'échangent leur état et dans la phase de calcul chaque capteur reconstruit l'image de la pièce. Nous reviendrons plus en détail en section 5.8.4. La pièce occupe $M_2 = P_{taille} \times P_{taille}$ bits, parce que la taille maximale d'une pièce est de P_{taille} par P_{taille} cellules. La valeur d'un critère de la pièce P est $M_3 = m(C_i, P)$. La section 6.5 donne quelques valeurs pratiques pour M_3. La somme de ces valeurs donne la mémoire totale nécessaire à une cellule d'un critère C_i :

$$m = M_1 + M_2 + M_3 \tag{4.3}$$

Soit CC_j l'ensemble des combinaisons de critères qui arrivent à une différenciation totale. Le

meilleur ensemble du point de vue de la mémoire est :

$$M = \{c_k / c_k = \min_{C_i \subset CC_j} M(C_i)\} \tag{4.4}$$

La mémoire nécessaire par la Smart Surface est de $X \cdot M$, où X est le nombre de cellules de la Smart Surface.

4.7.2 Temps d'exécution

Différencier les pièces se fait d'une manière distribuée par chaque cellule :

1: la pièce est reconstruite par chaque cellule
2: **repeat**
3: calcul de la valeur du critère (de la pièce qui est sur la Smart Surface)
4: comparaison avec tous les modèles de pièce
5: **if** non différenciation **then**
6: déplacer la pièce
7: **end if**
8: **until** différenciation

L'objectif de l'outil ECO est de trouver les meilleurs critères. Le temps nécessaire pour la reconstruction et le déplacement de la pièce est identique pour tous les critères et dépend de l'algorithme distribué utilisé. Il n'est donc pas pris en compte. Supposons que CC_i soit la meilleure combinaison de critères. Le pire cas est quand *tous* les critères inclus dans CC sont nécessaires pour reconnaître la pièce.

Soit $t_1(C_j)$ le temps d'exécution du critère C_j. Des valeurs pratiques du temps d'exécution de critères sont données dans la section 6.5. Le temps d'exécution total pour calculer la valeur de tous les critères est :

$$T_1 = \sum_{C_j \subset CC_i} t_1(C_j) \tag{4.5}$$

La valeur de chaque critère est un nombre. La comparaison de deux nombres est très rapide, soit t_2 le temps de comparaison (constant). n est le nombre de pièces à différencier. Le temps d'exécution de toutes les comparaisons est :

$$T_2 = \sum_{C_j \subset CC_i} n \times t_2 \tag{4.6}$$

Le temps d'exécution total est la somme des deux temps T_1 et T_2 :

$$T = \sum_{C_j \subset CC_i} (t_1(C_j) + n \times t_2) = n \times t_2 |CC_i| + \sum_{C_j \subset CC_i} t_1(C_j) \tag{4.7}$$

où $|CC_i|$ est le cardinal (nombre d'éléments) de CC_i.

4.8 Conclusion

Dans ce chapitre, nous avons présenté l'outil ECO qui permet de déterminer les meilleurs critères de différenciation. Cette différenciation s'effectue sur un ensemble de représentations

binaires générées de manière exhaustive. Cet ensemble de représentations binaires a été travaillé dans le but d'éliminer les représentations binaires identiques, les représentations binaires non connexes (selon la 4-connexité), les translations et les miroirs, afin d'obtenir un ensemble de représentations binaires uniques du point de vue de la rotation, de la translation et des miroirs. À partir de cet ensemble, tous les groupes de trois représentations binaires sont générés afin de pouvoir tester les critères de différenciation définis. Nous reviendrons plus en détail au chapitre 6.

Dans ce chapitre, nous sommes partis d'un cas exhaustif et général, à savoir essayer de trouver des critères de différenciation totale, c'est-à-dire qu'à partir de notre groupe de pièces, quelque soit le groupe de représentations binaires, la différenciation est possible. Pour arriver à ce résultat global nous avons posé quelques hypothèses (voir section 4.3).

Dans le chapitre 5, nous allons lever quelques contraintes et travailler sur un ensemble réduit de modèles de pièces. Nous allons également travailler directement sur les modèles et non sur leur représentation binaire et enfin nous serons dans un environnement à rotation libre.

Nous verrons dans les expérimentations (voir chapitre 6) que la différenciation totale est possible pour les pièces de taille 3×3 et 4×4.

CHAPITRE 5

Sélection du nombre optimal de capteurs pour la Smart Surface

Sommaire

5.1 Introduction

Dans ce chapitre, nous allons nous intéresser à la détermination du deuxième paramètre nécessaire pour la réalisation d'une Smart Surface performante. Il consiste à déterminer le nombre de capteurs nécessaires au fonctionnement de la Smart Surface.

Nous allons également essayer de répondre à différentes questions toutes liées entre elles : le nombre de capteurs a-t-il une influence sur la différenciation des pièces ? Si oui, quel est le nombre optimal de capteurs à positionner sur la Smart Surface ? Existe-t-il un seuil au-delà duquel les performances de la différenciation diminueraient ?

5.2 Plate-forme expérimentale

Avant de fabriquer la Smart surface, il nous faut déterminer le nombre de capteurs nécessaires à y inclure. Étant donné que la Smart Surface est en cours de fabrication, un Prototype de Smart Surface a été construit à l'échelle macro et mis à notre disposition au laboratoire FEMTO-ST[18]. Il nous permet de tester nos algorithmes afin de bien fixer le nombre de capteurs. La plate-forme est composée (voir figure 5.1) :

1. d'un Prototype de la Smart Surface ;
2. d'une caméra placée au-dessus.

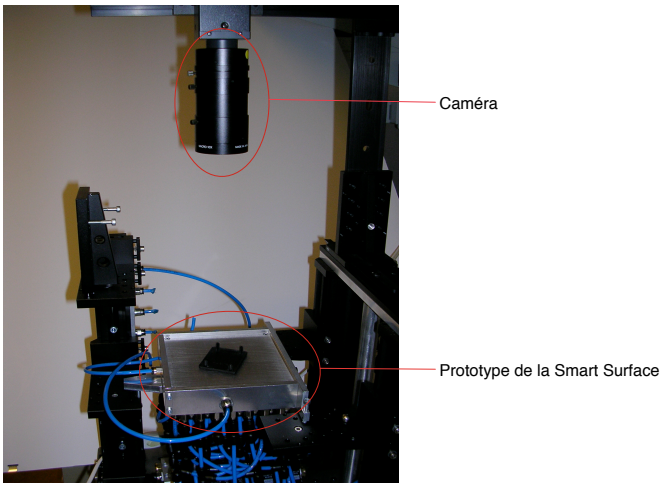


FIG. 5.1 – Vue globale de la plate-forme de la Smart Surface.

Le Prototype de la Smart Surface est une surface métallique carrée de 30 $cm \times 30$ cm qui contient une matrice de 15×15 actionneurs. La pièce qui est sur le Prototype de la Smart Surface se déplace grâce à un flux d'air qui passe à travers les actionneurs.

Grâce à la caméra positionnée au-dessus du Prototype de la Smart Surface, nous avons une liberté de discrétisation de la pièce.

La plate-forme Smart Surface nous permet de tester notre algorithme dans un cas *réel*.

5.3 Outil de sélection du nombre de capteurs

Pour bien fixer le nombre de capteurs nécessaires au fonctionnement de la Smart Surface, nous avons développé l'outil SNC (*Sensor Network Calibrator*) [70] qui permet de tester les

[18]Dans le département AS2M-Automatique et Systèmes Micro-Mécatroniques.

différentes tailles de grille de capteurs (voir figure 5.2).

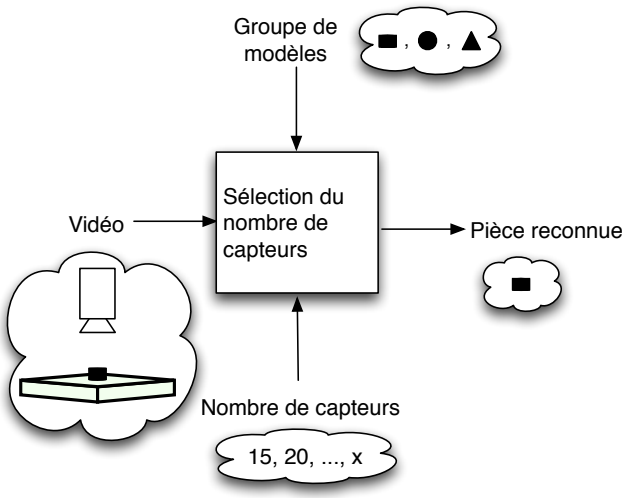


FIG. 5.2 – Une vue globale du calibrateur.

Cet outil [71], a pour but de répondre à la question suivante : pour chaque taille de grille de capteurs, quel est le résultat de différenciation obtenu ?

L'outil reçoit en entrée :

- une vidéo obtenue grâce à la caméra positionnée au dessus du Prototype Smart Surface et qui filme la surface et la pièce qui est posée dessus ;
- le groupe de pièces connues par le Prototype de la Smart Surface (PSS). Ces pièces sont appelées des *modèles de pièces* et représentent une donnée du PSS ;
- la taille de la grille de capteurs.

L'outil, va ensuite produire en sortie le résultat de différenciation obtenu pour les différentes tailles de la grille de capteurs.

La taille finale de la grille de capteurs dépendra du meilleur résultat de différenciation.

5.4 Fonctionnement de la plate-forme

Dans cette thèse, nous supposons que le fonctionnement de la Smart Surface est comme suit (voir figure 5.3) : dans la mémoire de la Smart Surface, il existe toutes les données nécessaires pour différencier un groupe de pièces. Ces données consistent en un ensemble de valeurs de tous les critères de différenciation des pièces (voir chapitre 4).

Quand une pièce est sur la Smart Surface, notre outil essaye de différencier cette pièce en calculant les différentes valeurs des critères de différenciation appliqués sur cette pièce et en comparant les valeurs obtenues au fur et à mesure avec les valeurs des critères associées aux modèles de pièces. Ce processus est réitéré tant qu'il y a une pièce sur la Smart Surface. Nous reviendrons plus en détail sur le processus de différenciation dans la section 5.8.5.

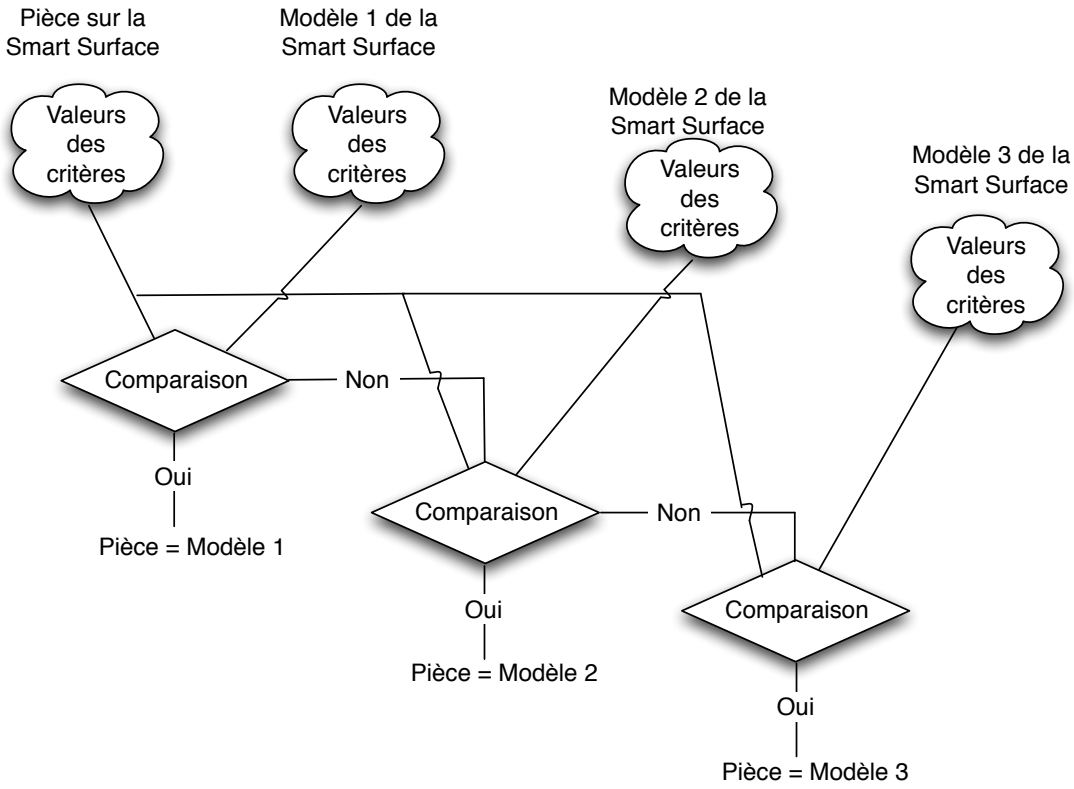


FIG. 5.3 – La structure globale de notre calibrateur.

5.5 Hypothèses de travail de l'outil SNC

Dans le chapitre 4, nous nous sommes fixés certaines hypothèses de travail (voir section 4.3), que nous allons ici assouplir :

- nous ne travaillons plus avec les représentations binaires des pièces mais directement avec les pièces ;
- nous ne travaillons plus en exhaustif mais avec un ensemble réduit de pièces (cas réel) ;
- nous ne travaillons plus en rotation à 90° mais plutôt en rotation libre ;
- il n'y a pas d'erreur de communication entre les capteurs ;
- il n'y a pas de défaillance des capteurs.

5.6 Structure globale de l'outil SNC

Notre outil SNC est composé de deux phases (figure 5.4) :

Phase de pré-traitement : cette phase qui se déroule en arrière plan (*offline*) consiste à calculer pour chaque modèle de pièce et pour chaque taille de la grille de capteurs l'ensemble des valeurs de critères associées à chaque modèle de pièce (nous reviendrons plus en détail sur ce processus dans la section 5.7). Ces résultats seront par la suite utilisés dans la phase temps réel.

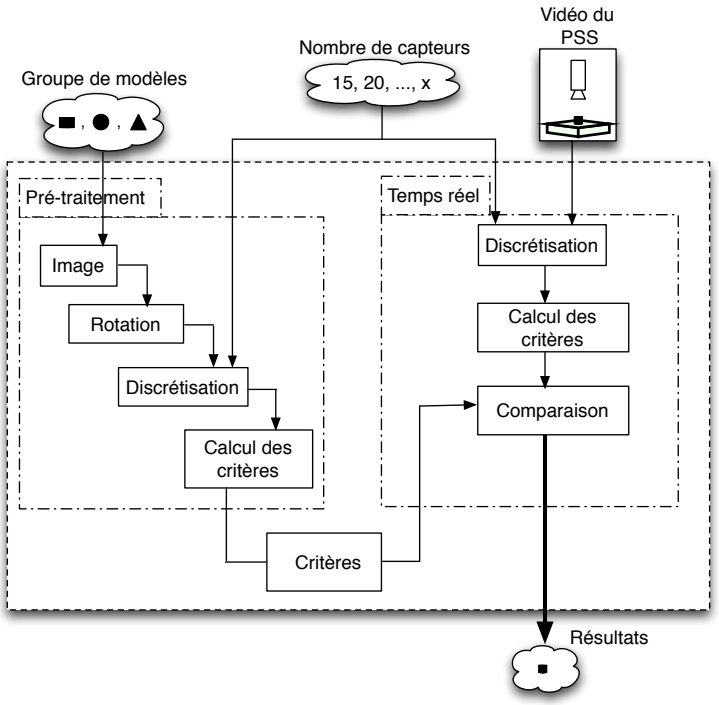


FIG. 5.4 – La structure globale de notre calibrateur.

Phase de temps réel : cette phase qui se déroule en temps réel (*online*) consiste à calculer les valeurs des critères associées à la pièce qui est sur la Smart Surface et cela pour différentes tailles de grille de capteurs. Les valeurs obtenues sont ensuite comparées avec les valeurs des critères des modèles de pièces déjà calculées dans la phase de pré-traitement (nous reviendrons plus en détail sur ce processus dans la section 5.8).

5.7 Phase de pré-traitement

Le but de cette phase consiste à calculer les différentes valeurs des critères associées à chaque modèle de pièce et cela pour chaque taille de grille de capteurs. Elle se compose de cinq sous-phases.

5.7.1 Processus de rotation

Quand une pièce est sur la Smart Surface, elle peut être dans n'importe quelle orientation. En effet, comme la pièce est maintenue en lévitation grâce à un flux d'air, l'orientation de la pièce peut changer à tout moment.

Cette partie est donc consacrée à la génération de toutes les rotations[19], avec un pas de rotation[20] de 1°, des images des **modèles** de pièces.

[19]La bibliothèque OpenCV a été utilisée pour les rotations.

[20]Pour avoir des erreurs de rotation négligeables, la résolution de l'image doit être beaucoup plus grande que

Les rotations des images sont calculées par rapport au centre des images.

La génération de toutes les rotations peut être résumée comme suit (voir figure 5.5) :

1: **for** chaque modèle de pièce M_i **do**
2: Acquisition de l'image du modèle de pièce M_i
3: **for** chaque j $\in \{1°, 2°,..., 360°\}$ **do**
4: générer $MRot_j$ la rotation de j degré de l'image M_i
5: **end for**
6: **end for**

À la fin de ce processus il va en résulter un ensemble d'images qui représentent les différentes rotations des images de chaque modèle de pièce.

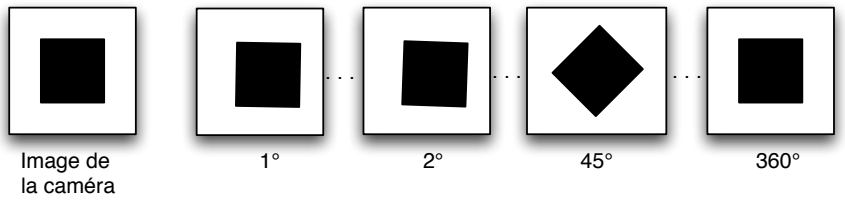


FIG. 5.5 – Toutes les rotations à 1°, d'un modèle de pièce.

5.7.2 Processus de translation

Comme la pièce est en lévitation sur la Smart Surface grâce à un flux d'air, elle subit aussi des mouvements de translation[21]. Nous allons donc aussi calculer toutes les translations des images $MRot_j$ obtenues après le processus de rotation de l'image M_i.

Le pas de translation est fixé de manière à ce que la probabilité que le capteur détecte la pièce après translation soit forte.

La translation est effectuée suivant l'axe des abscisses et l'axe des ordonnées. Le pas de translation suivant l'axe des abscisses est calculé avec l'équation suivante (voir figure 5.6(a)) :

$$Pas_{trans_X} = \frac{L}{NbCapteurs_X \times 2} \tag{5.1}$$

avec :
L : la largeur de l'image à translater.
$NbCapteurs_X$: le nombre de capteurs qui se trouvent sur une ligne de la Smart Surface.

Le pas de translation suivant l'axe des ordonnées est calculé avec l'équation suivante (voir figure 5.6(b)) :

$$Pas_{trans_Y} = \frac{H}{NbCapteurs_Y \times 2} \tag{5.2}$$

la résolution du Prototype de la Smart Surface.
[21]La bibliothèque OpenCV a été utilisée pour les translations.

avec :
H : la longueur de l'image à translater.
$NbCapteurs_Y$: le nombre de capteurs qui se trouvent sur une colonne de la Smart Surface.

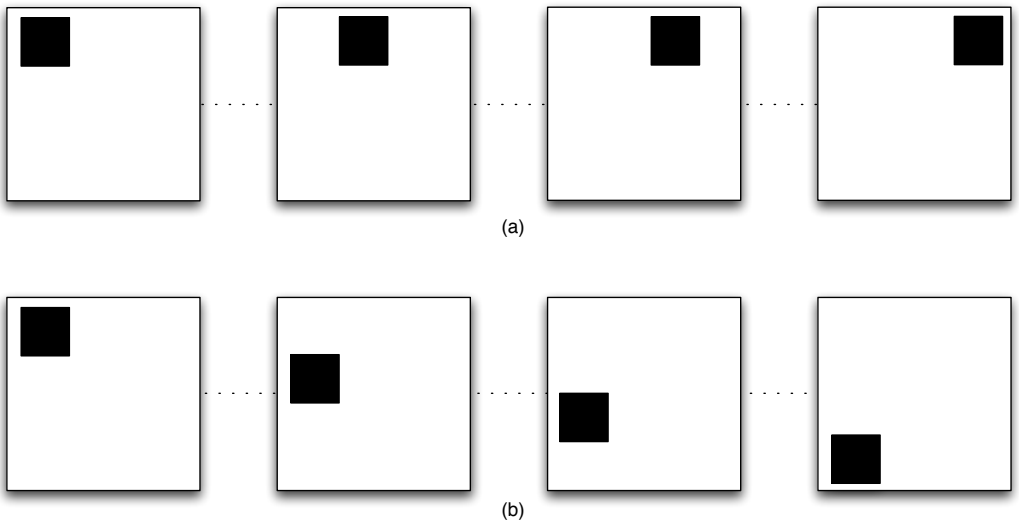


FIG. 5.6 – (a) Génération des translations suivant l'axe des abscisses. (b) Génération des translations suivant l'axe des ordonnées.

La génération de toutes les translations peut être résumée comme suit :

1: **for** chaque image $MRot_j$ obtenue après le processus de rotation de l'image M_i du modèle de pièce **do**
2: **for** chaque pas de translation Pas_{trans_X} selon l'axe des abscisses **do**
3: **for** chaque pas de translation Pas_{trans_Y} selon l'axe des ordonnées **do**
4: générer $ImTrans$ la translation de l'image $MRot_j$
5: **end for**
6: **end for**
7: **end for**

A la fin du processus de rotation et de translation, on obtient pour chaque modèle de pièce une série d'images qui correspondent à l'image du modèle de la pièce tournée avec un pas de rotation de 1°, translatée suivant l'axe des abscisses avec un pas de translation Pas_{trans_X} et translatée suivant l'axe des ordonnées avec un pas de translation Pas_{trans_Y}.

5.7.3 Processus de discrétisation

Toutes les images obtenues après l'application du processus de rotation et de translation vont être discrétisées. La discrétisation se définit par la transformation d'une image en une matrice (voir figure 5.7).

Dans notre cas, nous allons discrétiser toutes les images obtenues après l'application du processus de rotation et de translation. Étant donné que le Prototype de la Smart Surface ne

FIG. 5.7 – Principe de discrétisation d'une image.

contient pas de capteurs mais une caméra, nous allons tracer une grille (de même dimension que la grille de capteurs) sur l'image. Le centre de chaque cellule de la grille correspond à la position d'un capteur. Ensuite l'image est parcourue de gauche à droite et de haut en bas. On affecte 1 si le centre de la cellule (le capteur) est couvert par la pièce et 0 sinon.

Dans ce processus de discrétisation on remarque bien l'analogie avec la méthode Grid-based (section 2.3.1).

La discrétisation des images peut être résumée comme suit :

1: **for** chaque image **do**
2: tracer une grille de capteurs
3: récupérer la valeur de la matrice
4: **end for**

Après avoir appliqué le processus de discrétisation on obtient une matrice binaire correspondant à la représentation binaire de la pièce qui est sur la Smart Surface.

La figure 5.8 présente un exemple de discrétisation d'une image avec une grille de capteurs de taille 10×10.

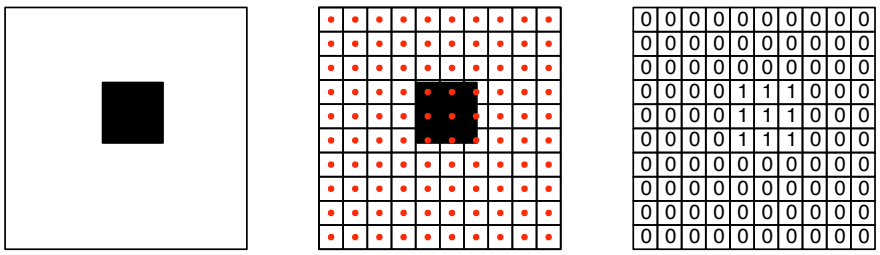


FIG. 5.8 – Exemple de discrétisation d'une image (les points représentent les positions des capteurs).

En appliquant ce processus sur l'ensemble des images, on obtient un ensemble de matrices qui correspondent à des représentations binaires des différentes images.

5.7.4 Génération des masques

Sur cet ensemble de représentations binaires, certaines sont redondantes (elles apparaissent plusieurs fois).

Cette redondance est due au pas de rotation très inférieur à la distance inter-capteurs. Cela implique qu'entre deux rotations, la différence n'est pas détectée par les capteurs (voir figure 5.9).

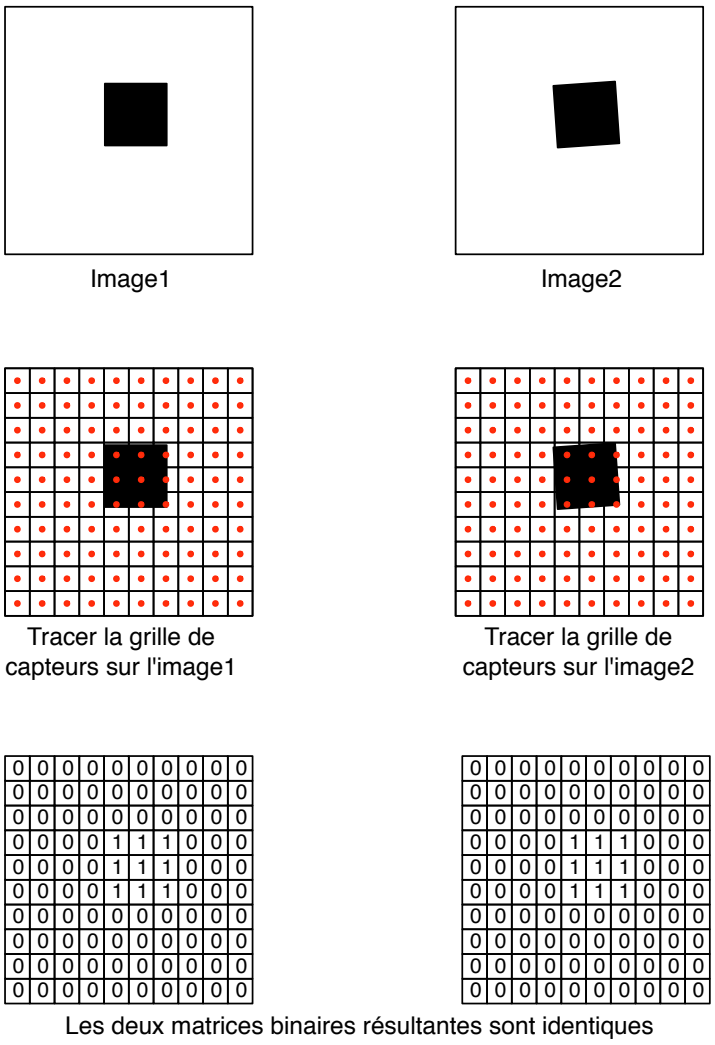


FIG. 5.9 – Exemple de pièces identiques après le processus de rotation.

Pour réduire l'ensemble des représentations binaires, on calcule les masques pour toutes les représentations binaires. Ces derniers sont comparés afin d'éliminer les masques redondants (voir figure 5.10).

La suppression des représentations redondantes peut être résumée comme suit :

1: **for** chaque représentation binaire **do**
2: calcul du masque
3: comparaison entre tous les masques

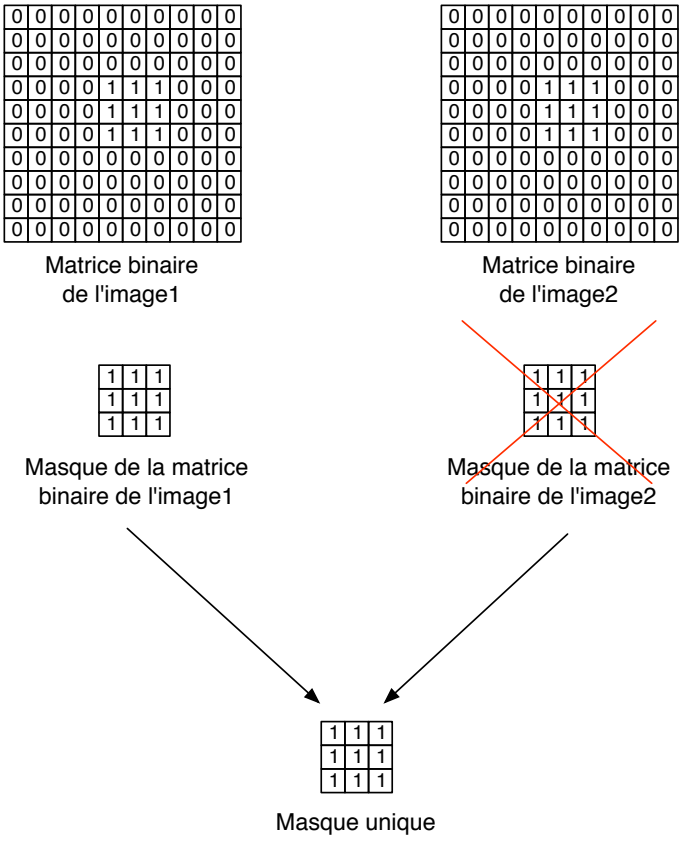


FIG. 5.10 – Exemple de suppression des représentations binaires identiques.

4: suppression des masques identiques
5: **end for**

Après ce processus de génération des masques il en résulte un ensemble de masques uniques.

5.7.5 Calcul des critères

Cette dernière étape de la phase de pré-traitement consiste à calculer les différentes valeurs de critères associées aux différents masques. Ces valeurs de critères obtenues constituent une base de données qui sera utilisée lors de la phase temps réel.

La phase de pré-traitement peut être décrite par l'algorithme suivant :

1: **for** chaque taille de grille de capteurs (n, n), avec $n \in \{15, 20, 25, 30, ..., 50\}$ **do**
2: **for** chaque modèle de pièce $M_i \in \{M_1, M_2, M_3, ..., M_{NbrModels}\}$ **do**
3: Acquérir l'image Im du modèle de la pièce sur le Prototype de la Smart Surface
4: **for** $d_{Rot} = 1°$ to $360°$ **do**
5: Générer $ImRot$ par une rotation de d_{Rot} degrés de l'image Im
6: **for** chaque pas de translation Pas_{trans_X} selon l'axe des abscisses **do**
7: **for** chaque pas de translation Pas_{trans_Y} selon l'axe des ordonnées **do**
8: Générer $ImTrans$ par une translation de l'image $ImRot$ avec Pas_{trans_X} pixels

par rapport à l'axe Ox et Pas_{trans_Y} pixels par rapport à l'axe Oy
Discrétiser l'image $ImTrans$
Générer le masque et le sauvegarder s'il est unique
Calculer et sauvegarder les valeurs de chaque critère
end for
end for
end for
end for
end for

Nous avons présenté en détail la phase de pré-traitement qui consiste en réalité à construire à partir des modèles de pièces une base de données. Cette dernière est un ensemble de valeurs des critères de différenciation que peut avoir chaque modèle de pièce selon certaines variations. Ces variations sont liées à la position des pièces, à leur orientation, à leur translation ou encore à la taille de grille de capteurs utilisée.

Cette base de données va ensuite être utilisée pour différencier une pièce qui est sur la Smart Surface. C'est l'objet de la phase temps réel.

5.8 Phase temps réel

Cette phase consiste à différencier une pièce qui est sur le Prototype de la Smart Surface. Elle répond à la question suivante : à quel modèle de pièce appartient la pièce qui est sur la Smart Surface ? Afin de déterminer la taille optimale de la grille de capteurs, cette phase est réitérée pour différentes tailles de la grille de capteurs.

Cette phase se décompose en cinq sous-phases qui s'exécutent tant qu'il existe une pièce sur la Smart Surface :

1. génération de l'arbre des valeurs de modèles de pièces ;
2. acquisition d'images ;
3. discrétisation d'images ;
4. reconstruction de l'image ;
5. différenciation.

5.8.1 Génération de l'arbre des valeurs de modèles de pièces

La différenciation s'effectue en calculant les critères de différenciation de la pièce qui est sur le Prototype de la Smart Surface. Ces valeurs seront par la suite comparées avec les valeurs des modèles de pièces calculées dans la phase de pré-traitement. Afin de faciliter cette comparaison les valeurs des critères de modèles de pièces obtenues dans la phase de pré-traitement sont classifiées dans un arbre selon les valeurs discriminantes (voir figure 5.11). Les nœuds intermédiaires de l'arbre représentent les critères, les feuilles représentent une pièce parmi les modèles de pièces et une arête peut relier soit deux critères, soit un critère avec une pièce de l'ensemble des modèles de pièces. Les arêtes qui relient deux critères sont pondérées par les différentes valeurs communes aux critères ne permettant pas de différencier les modèles de pièces. Les arêtes qui relient un

critère avec un modèle de pièce sont quant à elles pondérées par des valeurs discriminantes c'est-à-dire qui peuvent différencier un modèle de pièce.

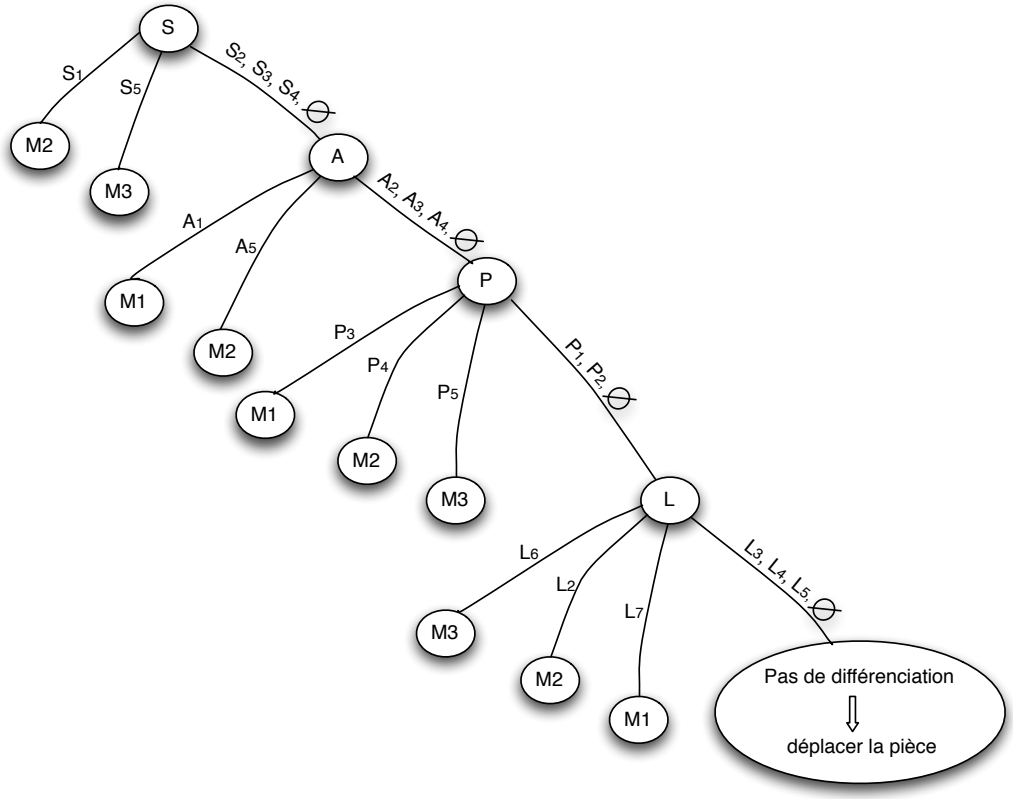


FIG. 5.11 – Exemple d'un arbre généré à partir des valeurs de critères associées aux modèles de pièces

Prenons un exemple M_1, M_2 et M_3 les modèles de pièces reconnus par la Smart Surface. Les valeurs des critères obtenues pour le modèle de pièce M_1 sont :

S : S_2, S_3, S_4.

A : A_1, A_2, A_3, A_4.

P : P_1, P_2, P_3.

L : L_3, L_4, L_5, L_7.

Les valeurs des critères obtenues pour le modèle de pièce M_2 sont :

S : S_1, S_2, S_3, S_4.

A : A_2, A_3, A_4.

P : P_1, P_2, P_4.

L : L_2, L_3, L_4, L_5.

Les valeurs des critères obtenues pour le modèle de pièce M_3 sont :

S : S_2, S_3, S_4, S_5.

A : A_2, A_3, A_4, A_5.

P : P_1, P_1, P_5.

L : L_3, L_4, L_5, L_6.

Donc pour le critère S la valeur S_1 est discriminante pour le modèle de pièce M_2 car cette valeur n'apparaît que dans les valeurs du critère S associées au modèle de pièce M_2. De même la valeur S_5 est discriminante pour le modèle de pièce M_3 car cette valeur n'apparaît que dans les valeurs du critère S associées au modèle de pièce M_3. Par contre pour les valeurs S_2, S_3, S_4 ou $\oslash$ (autre) du critère S on ne peut rien dire car ces valeurs existent dans plusieurs modèles de pièces. On calcule alors la valeur du deuxième critère, et on recommence.

5.8.2 Acquisition d'images

Grâce à une caméra positionnée au dessus du Prototype, on prend des séquences vidéo de la pièce en mouvement sur le Prototype de Smart Surface (voir figure 5.12(a)).

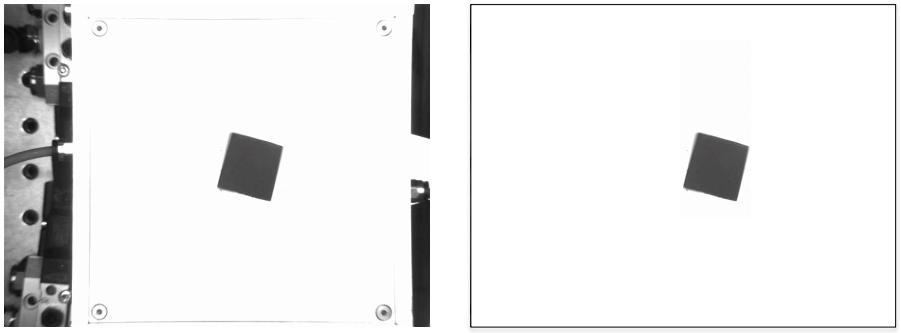


FIG. 5.12 – Image obtenue de la caméra : (a) avant le traitement, (b) après traitement.

La séquence d'images sera traitée[22] image par image. Chaque image sera binarisée (noir et blanc) afin de supprimer le bruit. Au final, on obtient une image contenant uniquement la pièce (voir figure 5.12(b)) [72].

5.8.3 Discrétisation d'images

Toutes les images (en noir et blanc) obtenues après le processus d'acquisition d'images vont être discrétisées suivant la taille de la grille de capteurs. Pour cela, on trace une grille (de mêmes dimensions que la grille de capteurs) sur l'image. Ensuite l'image est parcourue de gauche à droite et de haut en bas, on affecte 1 si le centre de la cellule est noir (couvert par la pièce) et 0 s'il est blanc (le capteur n'est pas couvert par la pièce).

Dans la figure 5.12(b) après application du processus de discrétisation avec une grille de taille 15×15 capteurs, on obtient une matrice binaire qui correspond à une représentation binaire de la pièce (voir figure 5.13).

[22] Ces traitements sont appliqués grâce à la bibliothèque OpenCV.

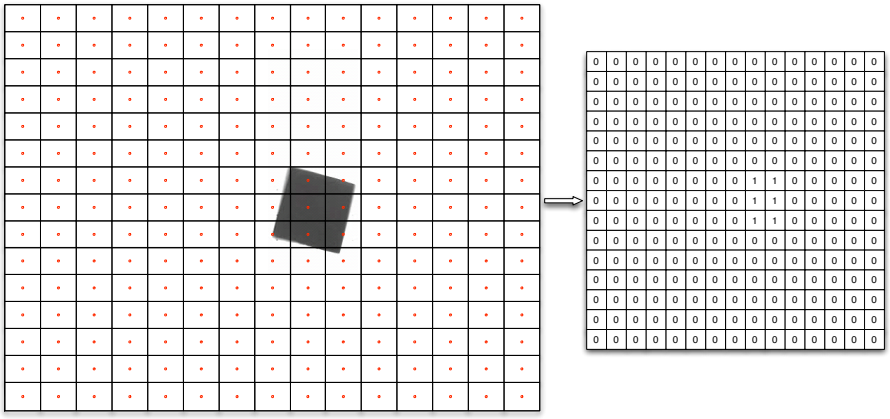


FIG. 5.13 – Discrétisation de l'image.

5.8.4 Reconstruction de l'image

La matrice binaire obtenue après la discrétisation de l'image nous permet en réalité de récupérer la vue de chaque capteur. Ce dernier connaît son état, s'il est recouvert par la pièce ou non, mais il ne connaît pas l'état de ses voisins. Les capteurs recouverts par la pièce sont appelés « *capteurs actifs* ». Chaque capteur actif va appliquer un processus itératif dans le but de reconstruire la pièce en entier. Ce processus itératif est réalisé en deux phases :

1. Phase de communication
2. Phase de calcul

Hypothèse de travail pour l'algorithme de reconstitution de la pièce

Pour appliquer notre algorithme de reconstitution d'images, nous avons fixé les hypothèses suivantes :

- chaque capteur ne connaît que son état ;
- chaque capteur communique avec ses quatre voisins (haut, bas , gauche et droit) ;
- le nombre d'itération est initialement fixé à la taille maximale du modèle de pièce `(TailleMaxPiece);`
- les capteurs s'échangent leur matrice binaire de même taille que la grille de capteurs où chaque élément de la matrice correspond à l'état du capteur ;
- il n'y a pas d'erreur de communication entre les capteurs ;
- il n'y a pas de défaillance des capteurs.

Le processus de reconstruction peut être expliqué par l'algorithme suivant :

1: **repeat**
2: Communication entre capteurs par envoi de message
3: Reconstitution de la pièce
4: **until** NbIteration> TailleMaxPiece

Phase de communication

Dans cette phase, tous les capteurs actifs vont communiquer avec leurs voisins. Cette communication s'effectue par simple envoi de message. Chaque capteur actif va envoyer un message qui consiste en une matrice binaire, de même taille que la grille de capteurs. Initialement, cette matrice contient des zéro et une seule valeur à 1. Cette valeur à 1 correspond à la vue associée au capteur qui envoie le message. Car à la première communication tous les capteurs actifs connaissent seulement leur état et ignorent totalement l'état de leurs voisins (voir figure 5.14).

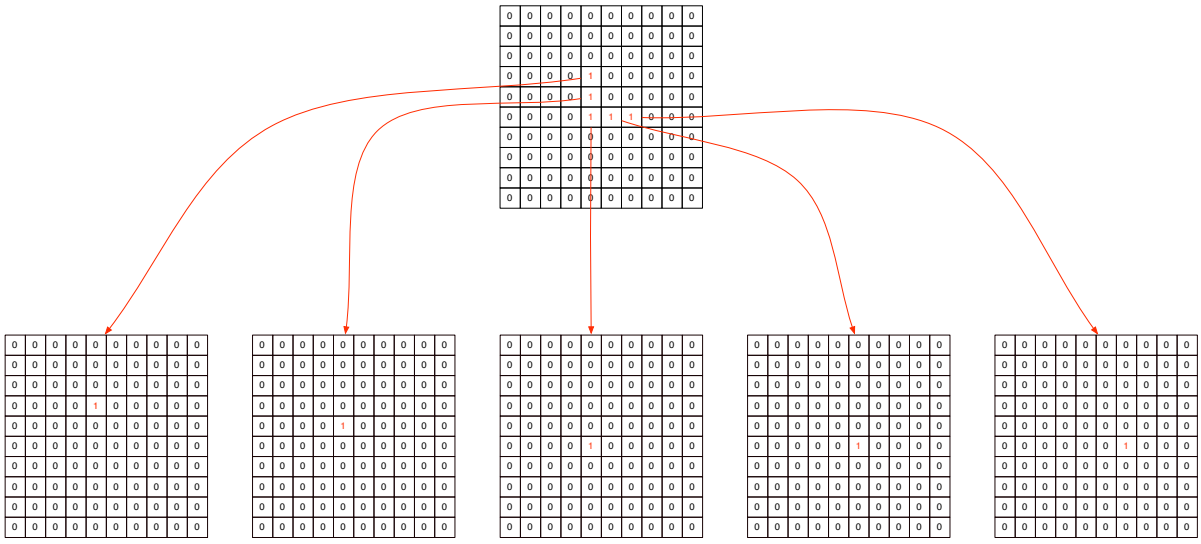


FIG. 5.14 – État des capteurs au début de la phase de communication.

Au fur et à mesure que la phase de communication est réitérée tous les capteurs actifs vont élargir leur vue grâce aux vues reçues de leurs voisins jusqu'à arriver à une vue globale de la pièce.

Phase de calcul

La phase de calcul consiste à mettre à jour la vue de chaque capteur en fonction des vues récoltées de ses voisins lors de la phase de communication. Cette mise à jour consiste à appliquer le « OU » logique entre l'ensemble des vues récoltées.

Soit M_{Cap} la matrice qui correspond à la vue du capteur Cap et M_{V_d}, M_{V_g}, M_{V_h}, M_{V_b}, $M_{V_{hg}}$, $M_{V_{bg}}$, $M_{V_{hd}}$ et $M_{V_{bd}}$ les vues correspondant respectivement aux voisins de droite, de gauche, de haut, de bas, de haut à gauche, de bas à gauche, de haut à droite et de bas à droite.

La mise à jour de la vue du capteur Cap est obtenue en appliquant l'équation 5.3 :

$$M_{Cap} = M_{V_d} \cup M_{V_g} \cup M_{V_h} \cup M_{V_b} \cup M_{V_{hg}} \cup M_{V_{bg}} \cup M_{V_{hd}} \cup M_{V_{bd}} \tag{5.3}$$

La figure 5.15 présente un exemple d'application obtenu en utilisant l'algorithme de reconstruction de la pièce.

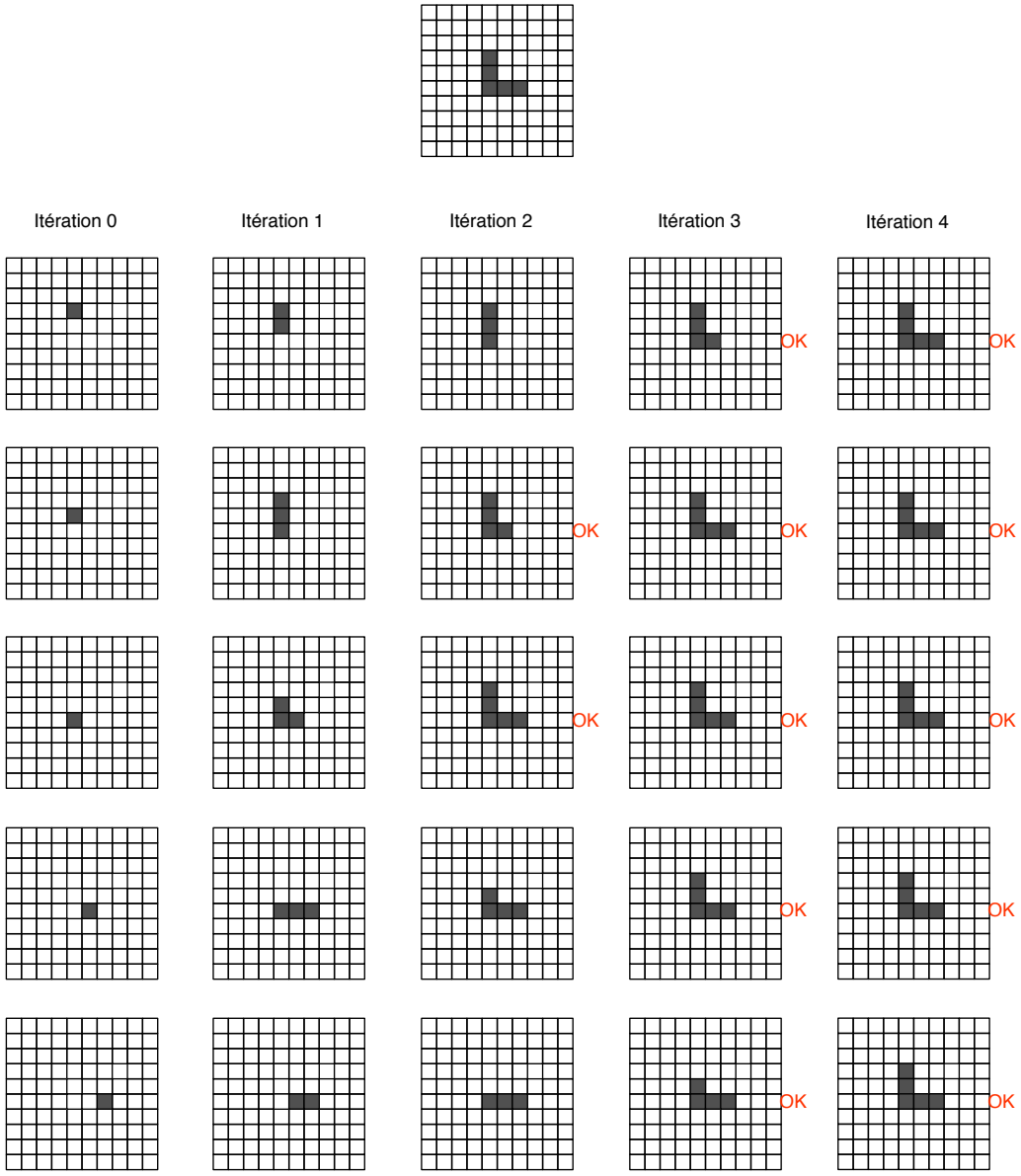


FIG. 5.15 – Exemple d'application du premier algorithme de reconstitution de la pièce.

Convergence de l'algorithme de reconstitution de la pièce

Nous avons travaillé en collaboration avec l'équipe de Didier El Baz (chargé de recherche HDR, au laboratoire du LAAS[23] à Toulouse) sur la convergence de l'algorithme de reconstitution de la pièce. Jaumouillé [73] a développé l'algorithme ROD dans lequel la taille du message dépend de l'itération à laquelle on se trouve. A la première itération, le capteur ne connaît que son état et donc transmet seulement un entier. Par la suite, la taille augmente en fonction de sa vue. L'algorithme s'arrête dès qu'un capteur ne reçoit que des valeurs nulles de ses voisins. En effet,

[23]Laboratoire d'Analyse et d'Architecture des Systèmes.

si la vue d'un capteur n'a pas changé alors il transmet une valeur nulle. Cet algorithme permet de réduire la quantité d'informations échangées entre les capteurs et converge plus rapidement. Néanmoins, il nécessite de recalculer à chaque étape la taille de la pièce (voir figure 5.16).

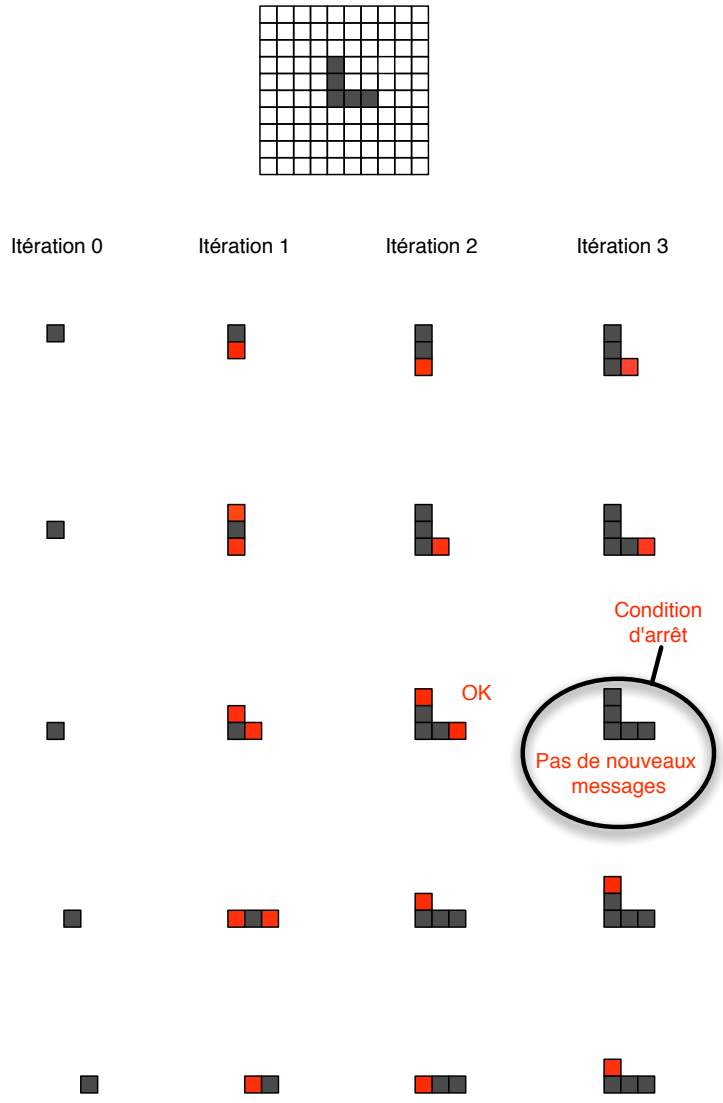


FIG. 5.16 – Exemple d'application du deuxième algorithme de reconstitution de la pièce.

5.8.5 Différenciation

La différenciation proprement dite consiste à calculer les valeurs des critères associées à la pièce qui est sur le Prototype de la Smart Surface et à comparer au fur et à mesure les résultats obtenus avec les valeurs des modèles de pièces. Pour chaque valeur de critère de la pièce calculée, l'arbre est parcouru. Si dans les arêtes de cet arbre on retrouve cette valeur dans un chemin discriminant alors la différenciation est possible, sinon on passe au critère suivant, et on recommence. Après avoir obtenu tous les résultats de différenciation, ceux-ci seront analysés

afin de déterminer la meilleure taille de la grille de capteurs. La phase temps réel peut être résumée grâce à l'algorithme ci-dessous :

1: Filmer la pièce sur le Prototype de la Smart Surface
2: Générer l'arbre résultant des valeurs de critères des modèles de pièces reconnus par le Prototype de la Smart Surface
3: **for** chaque image de la pièce sur le Prototype de la Smart Surface **do**
4: Discrétiser l'image
5: Générer le masque
6: Calculer les valeurs des critères
7: Comparer avec les valeurs des critères de chaque modèle de pièce
8: Donner le résultat de la différenciation
9: **end for**

5.9 Rapport de non différenciation

Le but de l'outil SNC est de différencier une pièce qui est sur la Smart Surface. Cela revient à calculer le taux de différenciation obtenu. Un taux de différenciation élevé implique que l'outil a bien différencié la pièce. Par contre un taux de différenciation faible signifie que l'outil n'est pas arrivé à différencier la pièce. De manière opposée, on peut dire que si le rapport de **non différenciation** est élevé cela implique que l'outil n'arrive pas à bien différencier la pièce. Par contre, un faible rapport de **non différenciation** signifie qu'il est possible de différencier la pièce. Nous allons justement nous intéresser au calcul du rapport de **non différenciation** noté R_{NoDiff}.

5.9.1 Notations

Soient : $M = \{M_1, M_2, ..., M_{nbModel}\}$: l'ensemble des images des modèles de pièces.

$n \in N$ le nombre de capteurs.

$r \in N$ le degré de rotation.

$x \in R^+$ le pas de translation suivant l'axe des abscisses.

$y \in R^+$ le pas de translation suivant l'axe des ordonnées.

$f(M_i, n, r, x, y) = \{RB_1, RB_2, ..., RB_n\}$: l'ensemble des représentations binaires de M_i.

f : $M \rightarrow f(M_i, n, r, x, y)$ une fonction de discrétisation telle que pour chaque image du modèle de pièce appartenant à M, il existe plusieurs représentations binaires qui lui sont associées (voir figure 5.17).

Il est clair que le nombre de représentations binaires augmente en fonction du nombre de capteurs (voir figure 5.18). Soient :

$C = \{C_1, C_2, C_3, ..., C_{nbCrit}\}$ l'ensemble des fonctions C_k de critères.

$C_k(f(M_i, n, r, x, y))$ l'ensemble des valeurs du critère C_k associées à chaque représentation binaire $f(M_i, n, r, x, y)$ du modèle de pièce M_i.

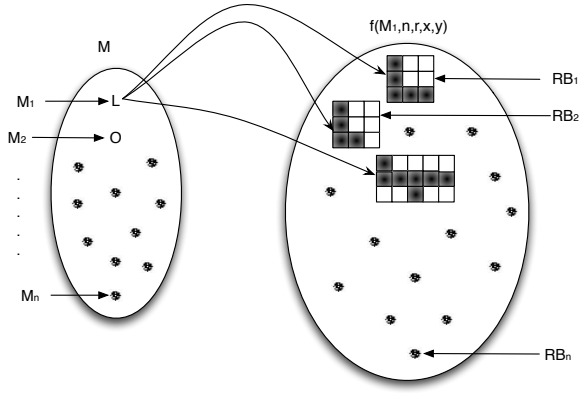


FIG. 5.17 – Relation entre les modèles de pièces et leurs représentations binaires.

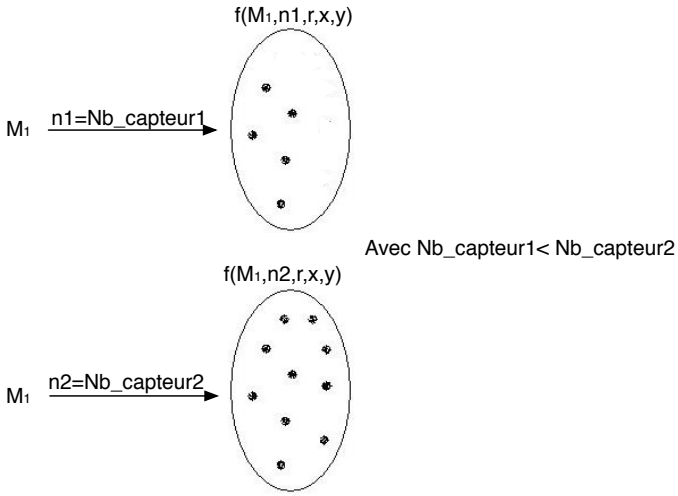


FIG. 5.18 – Relation entre les modèles de pièces et leurs représentations binaires en fonction du nombre de capteurs.

Et les fonctions C_k : $f(M_i, n, r, x, y) \rightarrow N$, k=1,...,nbCrit, telles que pour chaque représentation binaire $f(M_i, n, r, x, y)$, il existe plusieurs valeurs de critères qui lui sont associées (figure 5.19).

Le rapport de non différenciation R_{NoDiff} pour un critère est donné par l'équation suivante 5.4 :

$$R_{NoDiff} = \frac{|C_k(f(M_i, n, r, x, y))_{\forall r,x,y \in N} \cap C_k(f(M_j, n, r, x, y))_{\forall r,x,y \in N}|}{|C_k(f(M_i, n, r, x, y))_{\forall r,x,y \in N} \cup C_k(f(M_j, n, r, x, y))_{\forall r,x,y \in N}|} \tag{5.4}$$

Plus le rapport de non différenciation R_{NoDiff} augmente, plus le taux de différenciation décroît.

Le calcul du rapport de non différenciation R_{NoDiff} pour une combinaison de deux critères est donné par l'équation 5.5 :

$$R_{NoDiff} = \frac{|C_{k1}(f(M_i)) \times C_{k2}(f(M_i)) \cap C_{k1}(f(M_j)) \times C_{k2}(f(M_j))|}{|C_{k1}(f(M_i)) \times C_{k2}(f(M_i)) \cup C_{k1}(f(M_j)) \times C_{k2}(f(M_j))|} \tag{5.5}$$

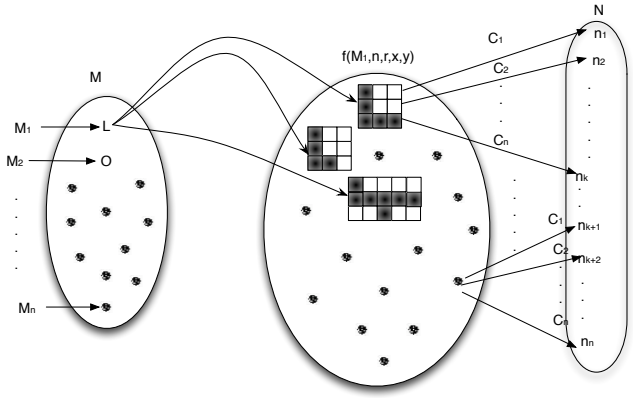


FIG. 5.19 – Relation entre les représentations binaires et leurs critères.

Avec :
$C_{k1}((f(M_i)) = C_{k1}(f(M_i, n, r, x, y))_{\forall r,x,y \in N}$
$C_{k2}((f(M_i)) = C_{k2}(f(M_i, n, r, x, y))_{\forall r,x,y \in N}$
$C_{k1}((f(M_j)) = C_{k1}(f(M_j, n, r, x, y))_{\forall r,x,y \in N}$
$C_{k2}((f(M_j)) = C_{k2}(f(M_j, n, r, x, y))_{\forall r,x,y \in N}$

Cette formule peut être généralisée dans le cas d'une combinaison de plusieurs critères avec l'équation 5.6 :

$$: R_{NoDiff} = \frac{|\cap_{k=1}^{NbCrit} (C_k(f(M_i)) \times C_k(f(M_j)))|}{|\cup_{k=1}^{NbCrit} (C_k(f(M_i)) \times C_k(f(M_j)))|} \tag{5.6}$$

Exemple : pour l'ensemble des représentations binaires de M_1, soient l'ensembles des valeurs des critères :
$C_1 = \{1, 2, 3, 4\}$.
$C_2 = \{1, 2\}$.
Donc :
$C_1(f(P_1)) \times C_2(f(P_1)) = \{(1,1), (1,2), (2,1), (2,2), (3,1), (3,2), (4,1), (4,2)\}$.
Pour l'ensemble des représentations binaires de M_2, soient l'ensembles des valeurs des critères :
$C_1 = \{1, 2, 3, 7\}$.
$C_2 = \{1, 2, 8\}$.
Donc :
$C_1(f(M_2)) \times C_2(f(M_2)) = \{(1,1), (1,2), (1,8), (2,1), (2,2), (2,8), (3,1), (3,2), (3,8), (7,1), (7,2), (7,8)\}$.
Le rapport de non différenciation pour une combinaison des deux critères est :
$R_{NoDiff} = \frac{6}{14}$.

5.10 Validation de l'approche distribuée

Afin de tester nos propositions dans un environnement réel, nous avons intégré nos outils logiciels avec ceux définis par le SP3 (contrôle distribué voir section 3.3) afin de contrôler le

Prototype de Smart Surface.

Le fonctionnement peut être résumé, selon les étapes suivantes :

- définition des modèles de pièces à différencier ;
- génération de la base de données des valeurs associées aux différents modèles de pièces ;
- utilisation de l'outil SNC pour fixer la taille de la grille de capteurs optimale à utiliser ;
- simulation des capteurs en discrétisant les images suivant la taille de la grille en respectant les contraintes temps-réel ;
- reconstitution de la pièce à partir des différentes vues des capteurs ;
- différenciation de la pièce ;
- contrôle distribué des actionneurs pour déplacer la pièce vers la bonne direction ;

Le but de cette expérimentation est de développer un programme qui sera directement exécuté sur le logiciel de la plate-forme. Cela nous permet de tester notre algorithme en temps réel, en ayant un temps de réponse inférieur à 0,02 seconde, car les images arrivent à une fréquence de 50 images par seconde. Pour ce faire, il nous a fallu adapter le code de SNC développé sous C++ pour pouvoir l'exécuter sous Simulink étant donné que ce dernier est utilisé par la caméra pour l'acquisition des images.

La figure 5.20 décrit l'architecture distribuée de notre expérimentation. Elle est composée de

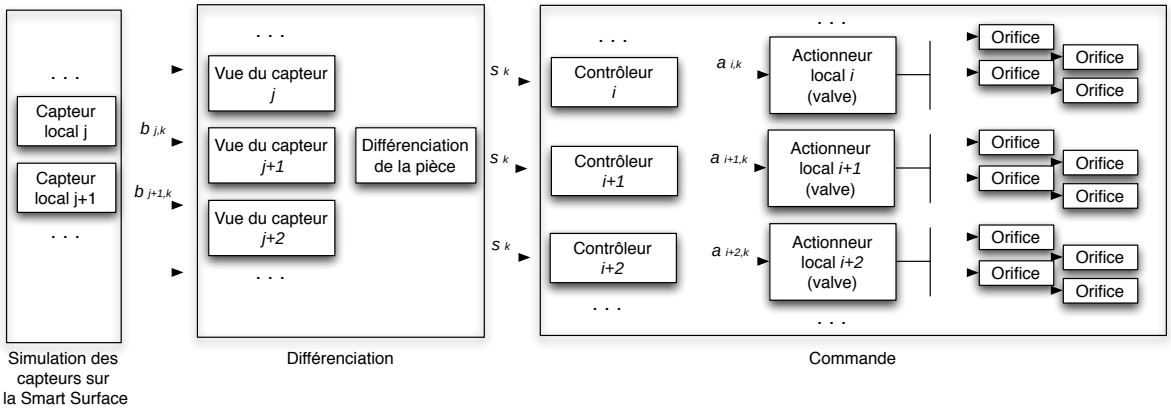


FIG. 5.20 – Architecture distribuée de notre expérimentation.

trois blocs : un bloc de simulation des capteurs sur la Smart surface, un bloc de différenciation et un bloc de commande. Le bloc de simulation permet de construire la grille de capteurs. Les vues des capteurs sont simulées grâce à une caméra. Chaque image k obtenue de la caméra est discrétisée en traçant une grille de capteurs sur l'image. Chaque capteur j va envoyer au « bloc de différenciation » son état obtenu par simulation (b_j, k) (chaque capteur j connaît son état recouvert par la pièce ou non). Dans le bloc de différenciation les capteurs vont communiquer entre eux pour reconstruire l'image de la pièce (voir section 5.8.4). Une fois la pièce différenciée l'information sera envoyée au module de commande qui va contrôler les actionneurs pour déplacer la pièce. Les blocs de différenciation et de commande sont réalisés grâce à un algorithme distribué sur un pc.

Mais avant de tester l'architecture distribuée, nous commençons d'abord par exécuter la

phase pré-traitement de l'outil SNC. Cette phase nous permettra de déterminer la taille de la grille de capteurs nécessaires pour différencier les modèles de pièces et de générer une base de données des différentes valeurs des critères associées aux modèles de pièces de la Smart Surface.

Les expérimentations sont en cours de finalisation mais les résultats obtenus jusqu'à présent nous permettent de valider le fonctionnement en temps réel de notre architecture distribuée avec un bon résultat de différenciation pour une grille de capteurs de taille 15×15. Il nous reste à la tester encore d'autres tailles de grille de capteurs et avec plus de modèles de pièces.

5.11 Conclusion

Dans ce chapitre, nous avons présenté l'outil SNC qui permet de déterminer le nombre de capteurs nécessaires pour différencier des pièces. Le nombre de capteurs est considéré comme un paramètre crucial au fonctionnement de la Smart Surface. Si ce paramètre est mal fixé cela causera des problèmes lors de la réalisation de la Smart Surface.

Conscients que les contraintes imposées dans le chapitre 4 sont trop lourdes, nous avons tenu à les alléger afin de nous approcher le mieux possible de la réalité. Par conséquent, nous ne travaillons plus sur un ensemble exhaustif de représentations binaires mais plutôt sur un ensemble réduit de modèles de pièces. De plus nous sommes dans un environnement à rotation libre.

Dans le chapitre 7 nous présenterons les résultats obtenus avec l'outil SNC. Ils montreront l'importance de l'utilisation de cet outil pour fixer le nombre de capteurs à utiliser. Le rapport de non différenciation R_{NoDiff} est comme un critère de sélection pour déterminer le nombre de capteurs nécessaires pour un bon fonctionnement de la Smart Surface. Il suffit alors de déterminer le nombre de capteurs qui donnent comme résultat un faible rapport de non différenciation R_{NoDiff}.

Troisième partie

Étude de cas

CHAPITRE 6

Étude de cas de l'outil de sélection des critères

Sommaire

6.1 Introduction

L'outil ECO (*exhaustive comparison framework*) a été réalisé dans le but de répondre à trois besoins : déterminer un ensemble de critères permettant de capter toutes les caractéristiques

de la pièce, parmi cet ensemble, ne garder que ceux qui arrivent à une différentiation totale et sélectionner parmi ces derniers ceux qui ont un coût mémoire et un temps d'exécution optimaux.

Pour déterminer si un critère arrive à une différentiation totale, nous générons toutes les pièces de taille 3×3 et 4×4, ensuite, à partir de ces pièces, nous générons tous les groupes de trois pièces de chaque ensemble. A la fin, nous obtenons deux ensembles de tous les groupes de trois pièces où chaque pièce a une taille de 3×3 et 4×4.

Cette génération exhaustive des pièces ainsi que celle des groupes va nous permettre de faire de manière objective une sélection des meilleurs critères en les comparant en terme de taux de différentiation.

6.2 Description des critères

Étant donné l'espace réduit sur la Smart Surface et la nécessité d'un temps de réponse rapide, les critères de différenciation ont été définis de manière à avoir des critères simples et faciles à implémenter. Dans ce qui suit nous allons détailler chacun de ces critères.

6.2.1 Le périmètre

Le périmètre d'une pièce est défini par le nombre de cellules à 1 qui sont à la frontière de la pièce, c'est-à-dire le nombre de cellule à 1 qui ont au moins une cellule voisine à 0. Ce critère est noté P (voir figure 6.1).

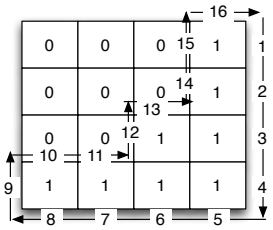


FIG. 6.1 – Une représentation binaire dont le périmètre est égal à 16.

6.2.2 La surface

La surface d'une pièce est le nombre de 1 contenus dans une représentation binaire. Ce critère est noté S (voir figure 6.2).

6.2.3 Le nombre d'angles

Ce critère consiste à parcourir la représentation binaire et compter le nombre de cellules à 1 avec au moins trois cellules voisines à 0 et formant un angle droit (voir figure 6.3). Ce critère est noté A.

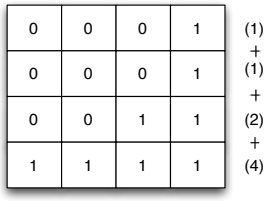


FIG. 6.2 – Une représentation binaire dont la surface est égale à 8.

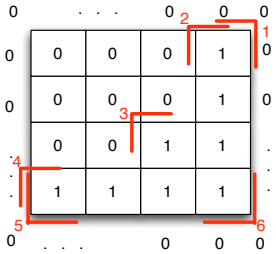


FIG. 6.3 – Une représentation binaire dont le nombre d'angles est égal à 6.

6.2.4 Distance maximale entre deux 1

Ce critère est déterminé en calculant toutes les distances de Manhattan entre les 1 de la représentation binaire de la pièce et de prendre la plus grande (voir figure 6.4). Ce critère est noté L. La distance de Manhattan est définie par l'équation 6.1 :

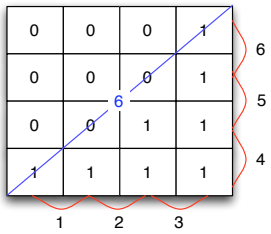


FIG. 6.4 – Une représentation binaire dont la longueur maximale est égale à 6.

$$d_M(c_1, c_2) = |c_{1_{(x)}} - c_{2_{(x)}}| + |c_{1_{(y)}} - c_{2_{(y)}}| \tag{6.1}$$

Avec :
c_1 et c_2 : deux cellules de la représentation binaire de la pièce.

6.2.5 La somme des cellules qui changent successivement

Le critère N est le nombre de cellules qui changent entre deux lignes successives et deux colonnes successives. Ce critère s'implémente facilement en parcourant la représentation binaire de la pièce et en additionnant :

1. le nombre de cellules qui changent entre deux lignes successives (voir figure 6.5(a)) ;
2. le nombre de cellules qui changent entre deux colonnes successives (voir figure 6.5(b)).

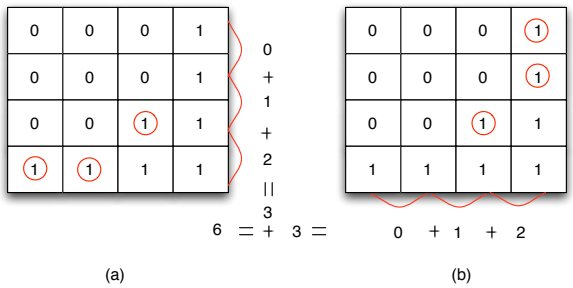


FIG. 6.5 – Une représentation binaire dont le nombre de cellules qui changent entre deux lignes (a) et deux colonnes (b) consécutives est égal à 6.

6.2.6 Distance maximale entre deux 0

La distance maximale entre deux 0 d'une pièce se définit par le calcul de toutes les distances de Manhattan entre les 0 de la représentation binaire de la pièce afin de retenir la plus grande valeur (voir figure 6.6). Ce critère est noté Z.

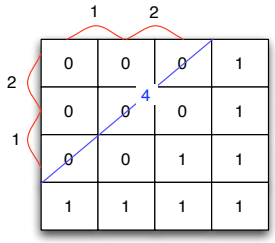


FIG. 6.6 – Une représentation binaire dont la longueur maximale entre deux 0 est égale à 4.

6.2.7 Somme des 1 des diagonales

Le critère D (voir figure 6.7), consiste à additionner le nombre de 1 qui se trouvent sur les deux diagonales. Il se définit par l'équation 6.2 :

$$D = \Sigma_{i=0}^{n-1} Mat(i,i) + \Sigma_{i=0}^{n-1} Mat(n-1-i,i) \tag{6.2}$$

avec Mat : la matrice de la représentation binaire de la pièce.

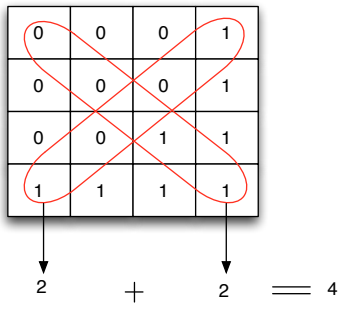


FIG. 6.7 – Une représentation binaire dont le nombre de 1 qui sont dans les deux diagonales est égal à 4.

6.2.8 Somme des distances entre les 0

La somme des distances entre les 0 d'une pièce est définit par la somme de toutes les distances de Manhattan entre les 0 de la représentation binaire de la pièce (voir figure 6.8). Elle est calculée à l'aide de l'équation 6.3. Ce critère est appelé F.

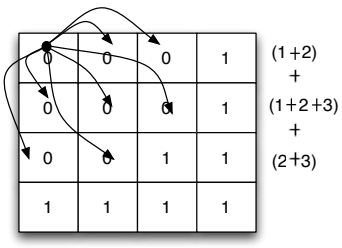


FIG. 6.8 – Une représentation binaire dont la somme des distances de Manhattan est égale à 14.

$$F = \Sigma_{c_1 \in M_{cap}} \Sigma_{c_2 \in M_{cap}(c_1 \neq c_2)} d_M(c_1, c_2)$$
$$M_{cap} = \{(x,y)/Cap_{(x,y)} = 0\} \text{ avec Cap la matrice de capteurs} \quad (6.3)$$

Avec :
c_1 et c_2 : deux cellules de la représentation binaire de la pièce.
$d_M(c_1, c_2)$: distance de Manhattan.

6.2.9 Somme des cellules qui changent

Le critère M consiste à parcourir la représentation binaire de la pièce et à calculer la somme des nombres de cellules qui changent entre :

1. la première ligne et toutes les autres lignes (voir figure 6.9(a)) ;
2. la dernière ligne et toutes les autres lignes (voir figure 6.9(b)) ;
3. la première colonne et toutes les autres colonnes (voir figure 6.9(c)) ;
4. la dernière colonne et toutes les autres colonnes (voir figure 6.9(d)).

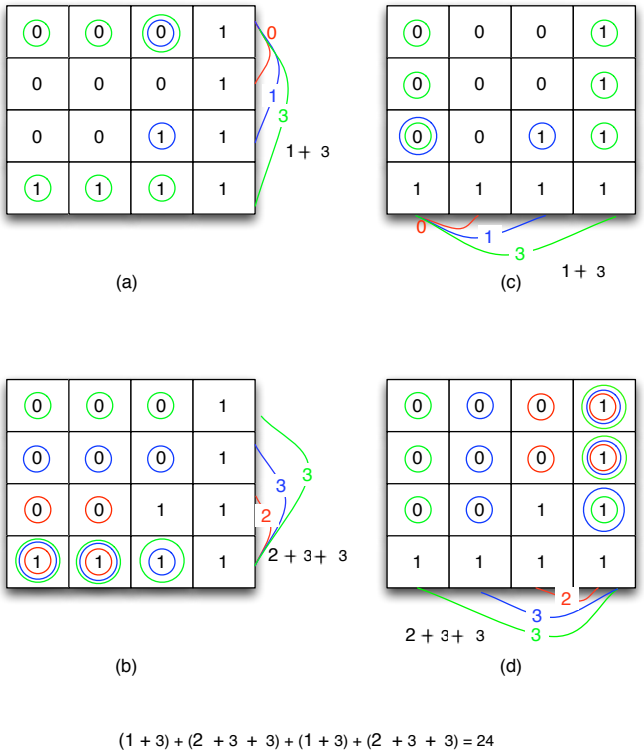


FIG. 6.9 – Une représentation binaire dont la somme des cellules qui changent est égale à 24.

6.2.10 Produit d'angles en "V"

Le produit d'angles en "V" se définit par le nombre d'angles qui forment un "V". Ce critère consiste à parcourir la représentation binaire et à compter le nombre de (voir figure 6.10) :

1. trois 1 dans la pièce qui forment un "V ";
2. trois 1 dans la pièce qui forment un ">";
3. trois 1 dans la pièce qui forment un "<" (voir figure 6.10(a)) ;
4. trois 1 dans la pièce qui forment un "Λ" (voir figure 6.10(b)).

Ce critère est noté R.

6.2.11 La somme des lignes et colonnes identiques

Ce critère consiste à parcourir la représentation binaire et à compter le couple de lignes (voir figure 6.11(a)) et colonnes (voir figure 6.11(b)) identiques. Ce critère est noté I.

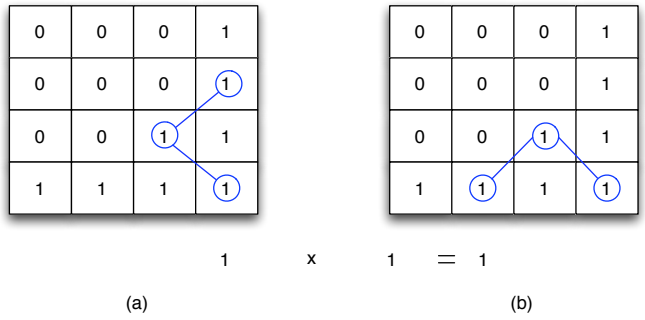


FIG. 6.10 – Une représentation binaire dont la somme des angles en “V” est égale à 1.

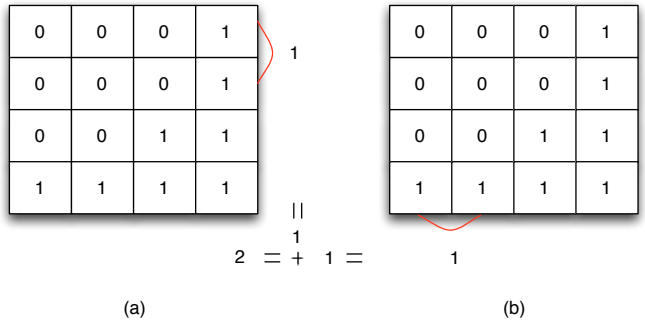


FIG. 6.11 – Une représentation binaire dont la somme des lignes (a) et des colonnes (b) identiques est égale à 2.

6.2.12 Produit des distances entre les 0

Le critère T (voir figure 6.12) consiste à calculer le produit de toutes les distances de Manhattan entre les 0 avec l'équation 6.4 :

$$\begin{gathered} T = \prod_{c_1 \in M_{cap}} \prod_{c_2 \in M_{cap}(c_1 \neq c_2)} d_M(c_1, c_2) \\ M_{cap} = \{(x,y)/Cap_{(x,y)} = 0\} \text{ avec Cap la matrice de capteurs} \end{gathered} \tag{6.4}$$

Avec :
c_1 et c_2 : deux cellules de la représentation binaire de la pièce.
$d_M(c_1, c_2)$: la distance de Manhattan.

6.2.13 Produit des distances entre les 1

Le critère Y (voir figure 6.13) consiste à calculer le produit de toutes les distances de Manhattan entre les 1 avec l'équation 6.5 :

$$\begin{gathered} Y = \prod_{c_1 \in M_{cap}} \prod_{c_2 \in M_{cap}(c_1 \neq c_2)} d_M(c_1, c_2) \\ M_{cap} = \{(x,y)/Cap_{(x,y)} = 1\} \text{ avec Cap la matrice de capteurs} \end{gathered} \tag{6.5}$$

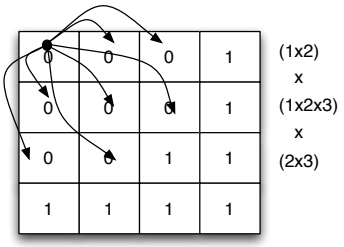


FIG. 6.12 – Une représentation binaire dont le produit de toutes les distances entre les 0 est égal à 72.

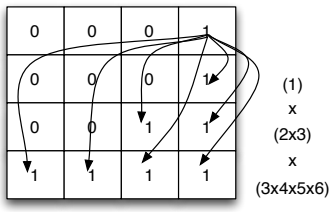


FIG. 6.13 – Une représentation binaire dont le produit de toutes les distances entre les 1 est égal à 2160.

Avec :
c_1 et c_2 : deux cellules de la représentation binaire de la pièce.
$d_M(c_1, c_2)$: la distance de Manhattan.

6.2.14 Produit des cellules qui changent successivement

Le critère E est le produit du nombre de cellules qui changent entre deux lignes et deux colonnes successives. Ce critère consiste à parcourir la représentation binaire de la pièce et à calculer le produit entre :

1. le nombre de cellules qui changent entre deux lignes successives (voir figure 6.14(a)) ;
2. le nombre de cellules qui changent entre deux colonnes successives (voir figure 6.14(b)).

6.2.15 Somme d'angles en "V" avec les deux bouts à 0

Le critère C consiste à calculer le nombre d'angles qui forment un "V". Ce critère consiste à parcourir la représentation binaire et à compter le nombre de (voir figure 6.15) :

1. 1 au coin et deux 0 au bout qui forment un "V" (voir figure 6.15(a)) ;
2. 1 au coin et deux 0 au bout qui forment un "> " (voir figure 6.15(b)) ;
3. 1 au coin et deux 0 au bout qui forment un "<";
4. 1 au coin et deux 0 au bout qui forment un "Λ".

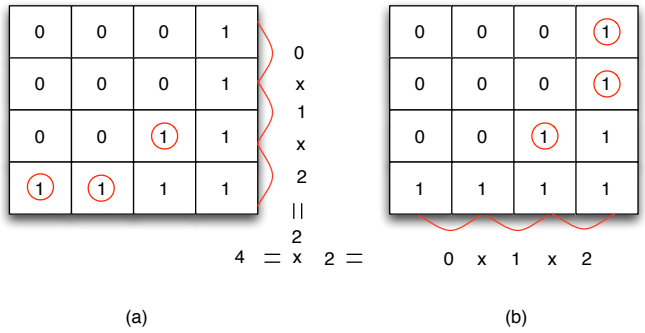


FIG. 6.14 – Une représentation binaire dont le produit des cellules qui changent successivement est égal à 4.

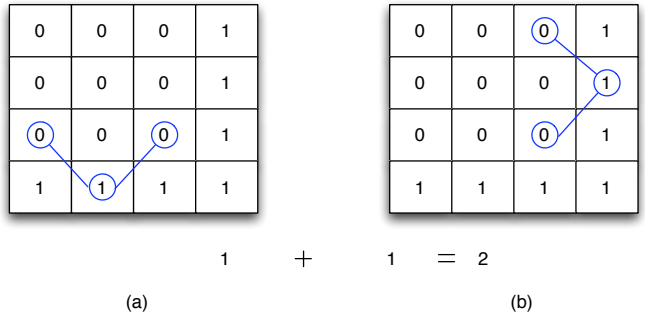


FIG. 6.15 – Une représentation binaire dont la somme d'angles en "V" avec les deux bouts à 0 est égale à 2.

6.2.16 Produit des cellules qui changent

Le produit des cellules qui changent consiste à parcourir la représentation binaire de la pièce et à calculer le produit des nombres de cellules qui changent entre :

1. la première ligne et toutes les autres lignes (voir figure 6.16(a)) ;
2. la dernière ligne et toutes les autres lignes (voir figure 6.16(b)) ;
3. la première colonne et toutes les autres colonnes (voir figure 6.16(c)) ;
4. la dernière colonne et toutes les autres colonnes (voir figure 6.16(d)).

Ce critère est noté K.

6.3 Classification des critères de différenciation

Dans la définition des critères de différenciation, nous remarquons qu'il en existe qui se basent, pour certains sur le contour et pour d'autres sur la région. Il est possible de les classer comme suit :

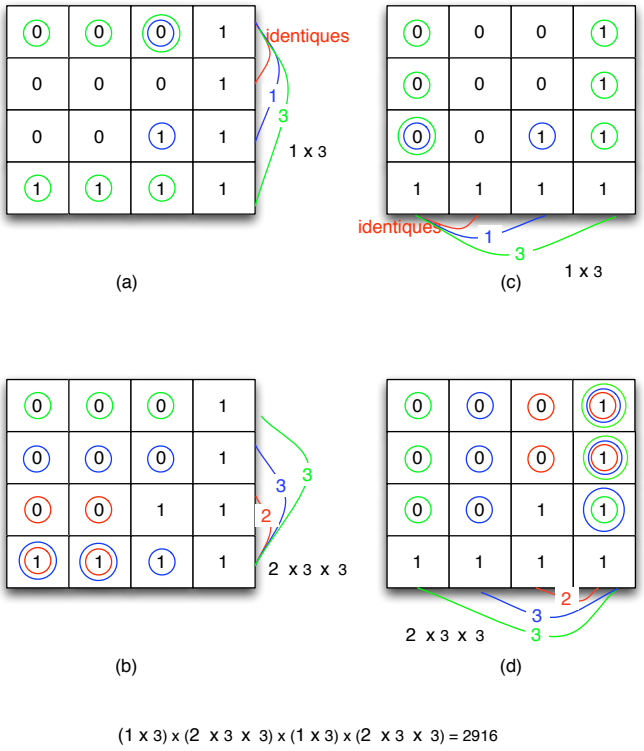


FIG. 6.16 – Une représentation binaire dont le produit des cellules qui changent est égal à 2916.

Critères basés sur le contour

- P : le périmètre.
- A : le nombre d'angles.

Critères basés sur la région

- S : la surface.
- L : la distance maximale entre deux 1.
- N : la somme des cellules qui changent successivement.
- Z : la distance maximale entre deux 0.
- D : la somme des 1 des diagonales.
- F : la somme des distances entre les 0.
- M : la somme des cellules qui changent.
- R : le produit d'angles " V ".
- I : la somme des lignes et colonnes identiques.
- T : le produit des distances entre les 0.
- Y : le produit des distances entre les 1.
- E : le produit des cellules qui changent successivement.
- C : la somme d'angles " V " avec les deux bouts à 0.
- K : le produit des cellules qui changent.

Nous remarquons qu'il existe bien une analogie avec la classification présentée au chapitre 2.

6.4 Sélection des critères de différenciation totale

Une fois tous ces critères définis, nous les avons testé sur l'ensemble des groupes de trois représentations binaires des pièces de taille 3×3 et 4×4. Nous nous sommes particulièrement intéressés aux combinaisons qui arrivent à une différenciation totale.

Les résultats montrent que les combinaisons de critères minimales qui arrivent à une différenciation totale sont :

pour les représentations binaires des pièces de taille 3×3 :

$$CC = \{\{TM\}, \{TK\}, \{YF\}, \{YM\}, \{YK\}, \{YE\}\}$$

pour les représentations binaires des pièces de taille 4×4 :

$$CC = \{\{CFIMPRZ\}, \{CFILMRZ\}, \{CDFIMRZ\}, \{ACFIPRZ\}, \{ACFINRZ\}, \{ACFIMRZ\}, \{ACFILRZ\}, \{ACDFIRZ\}, \{CFIMNRZ\}\}.$$

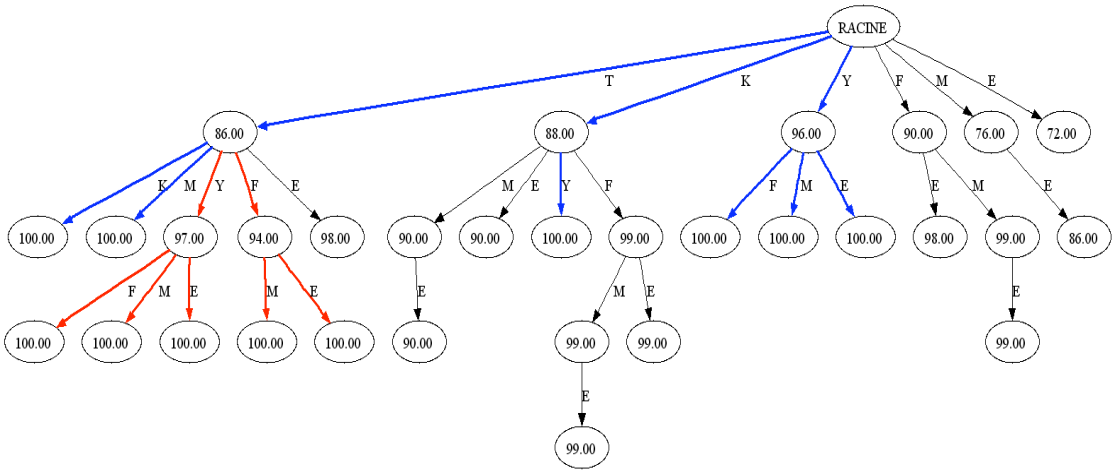


FIG. 6.17 – Arbre des combinaisons qui arrivent à 100% pour les représentations binaires de taille 3×3.

Afin de déterminer quels sont les critères ou combinaisons de critères pertinents en terme de résultat de différenciation (qui arrivent à une différenciation totale). Nous avons choisi de synthétiser ces résultats par un arbre (généré automatiquement) des combinaisons dont les arêtes représentent les critères utilisés et les nœuds les résultats de différenciation obtenus. Le plus court chemin dans l'arbre (en bleu) représente la plus courte combinaison qui arrive à 100% de différenciation. Par contre, le plus long chemin dans l'arbre (en rouge) représente la plus longue combinaison qui arrive à une différenciation totale. Dans l'exemple de la figure 6.17, le chemin en bleu $\{RACINE, T, K\}$ signifie qu' avec la combinaison $\{T, K\}$ on arrive à une différenciation totale pour l'ensemble des représentations binaires de taille 3×3 par ailleurs le chemin $\{RACINE, T, Y, F\}$ signifie qu'une plus longue combinaison de critères $\{T, Y, F\}$ est nécessaire pour arriver à une différenciation totale.

À partir de la figure 6.17, on remarque qu'il existe six chemins en bleu. Chacun d'eux est composé d'une combinaison de deux critères. Cela signifie que pour les représentations binaires

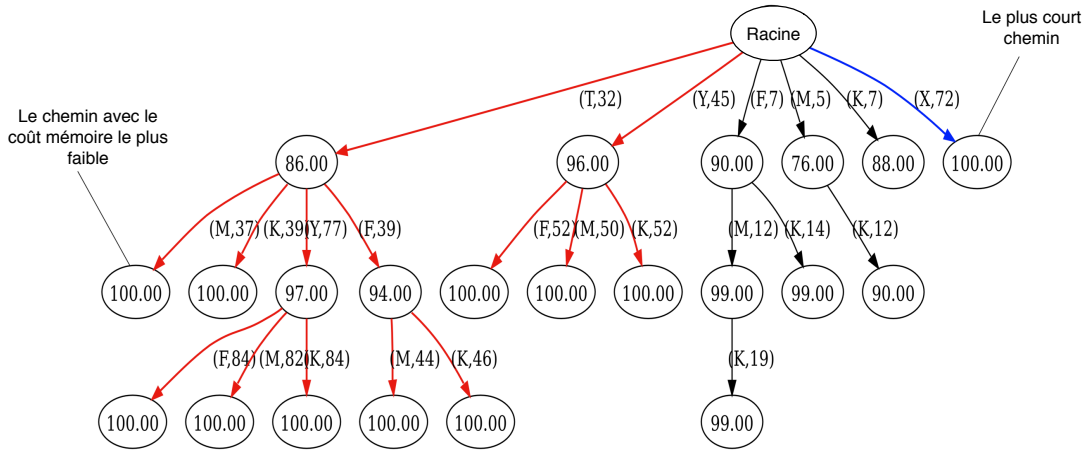


FIG. 6.18 – Coût mémoire.

des pièces de taille 3×3, il nous faut au minimum deux critères pour arriver à une différenciation totale.

Sur la page web [74] sont également présentés les différents arbres des résultats obtenus en appliquant les critères sur l'ensemble des représentations binaires des pièces de taille 4×4. On remarque qu'il existe neuf combinaisons de sept critères qui arrivent à une différenciation totale (chemin en bleu).

6.5 Le coût mémoire et le temps d'exécution des critères qui arrivent à 100%

Pour les représentations binaires des pièces de taille 3×3, il existe six combinaisons de deux critères, qui arrivent à une différenciation totale. Nous nous sommes intéressés à comparer notre méthode de différenciation à base de critères avec la méthode de Grid-based notée "X" [61]. Cette méthode consiste principalement à comparer deux images en appliquant l'opérateur XOR entre leur représentation binaire.

Appliquer le XOR sur notre ensemble des représentations binaires de pièces revient à sauvegarder toutes les représentations binaires, soit 9 bits pour une représentation binaire de pièce de taille 3×3 ainsi que ses rotations à 90° et ses miroirs. Ce qui implique un coût mémoire total de 72 bits. La figure 6.18 montre un exemple de la consommation mémoire pour tous les critères (T, Y, F, M, K, X) et leurs combinaisons. C'est le même type d'arbre qui est présenté en figure 6.17 sauf que les étiquettes des arêtes contiennent en plus du critère de différenciation le coût mémoire associé à chaque critère ou combinaison de critères.

Soient : M_{C_1} le coût mémoire associé au critère C_1 et M_{C_2} le coût mémoire associé au critère C_2. Le coût mémoire associé à la combinaison de critères C_1C_2 est obtenu par l'équation 6.6 :

$$M_{C_1C_2} = M_{C_1} + M_{C_2} \tag{6.6}$$

Par exemple sur le chemin $\{RACINE, T, M\}$, sur la première arête est associée l'étiquette

$(T, 32)$, cela signifie qu'en appliquant le critère T sur l'ensemble des représentations binaires des pièces de taille 3×3, on obtient un taux de différenciation de 86.00% avec un coût mémoire de 32 bits. Par contre à la deuxième arête est associée l'étiquette $(M, 37)$. Elle signifie qu'appliquer la combinaison de critères $\{T, M\}$ sur l'ensemble des représentations binaires des pièces de taille 3×3 permet d'obtenir un taux de différenciation de 100% avec un coût mémoire de 37 bits qui correspond à la somme du coût mémoire du critère $(T, 32)$ et le coût mémoire du critère $(M, 5)$.

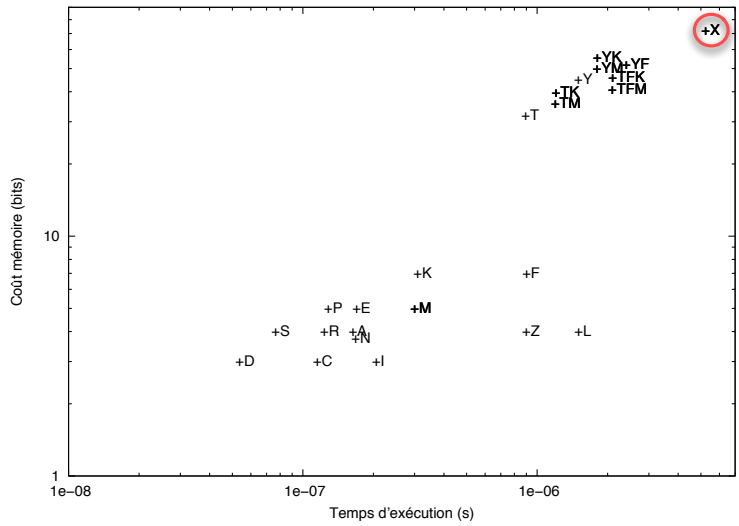


FIG. 6.19 – Coût mémoire en fonction du temps d'exécution des critères.

De la figure 6.18, nous pouvons facilement déduire que la combinaison $\{T, M\}$ arrive à une différenciation totale avec 37 bits seulement, ce qui est nettement inférieur au coût mémoire du critère X qui représente le critère XOR de la méthode de Grid-based.

Nous nous sommes aussi intéressés à comparer le coût mémoire des neuf combinaisons de sept critères qui arrivent à une différenciation totale avec le coût mémoire du critère de la méthode Grid-based qui applique le XOR sur l'ensemble des représentations binaires de taille 4×4. Cette comparaison revient à sauvegarder toute la représentation de la pièce 16 bits plus toutes ses rotations à 90° et ses miroirs ce qui fait un coût mémoire total de 128 bits.

Sur la page web [74], sont présentés des arbres de la consommation mémoire des neuf combinaisons de sept critères qui arrivent à une différenciation totale. Par exemple pour la figure 2.1 de [74], nous remarquons que pour le chemin $\{RACINE, A, C, F, I, N, R, Z\}$ on arrive à une différenciation totale avec 36 bits de consommation mémoire ce qui est nettement inférieur en comparaison avec le coût mémoire du critère X (128 bits).

Nous nous sommes aussi intéressés à comparer le critère X avec nos différents critères de différenciation du point de vue du temps d'exécution nécessaire à chacun.

La figure 6.19 présente un nuage de points du coût de la mémoire en fonction du temps d'exécution des critères. Il existe plusieurs combinaisons de critères qui arrivent à une différenciation totale avec un temps d'exécution inférieur au critère XOR (voir figure 6.20). Comme la

méthode Grid-based est basée sur le calcul du XOR entre les représentations binaires et sachant que le XOR est sensible à la rotation et au miroirs, il est nécessaire de sauvegarder pour chaque représentation binaire ses rotations et ses miroirs. Cela augmente le nombre d'opérations du XOR, contrairement à notre méthode qui consiste à calculer les valeurs du critère pour deux représentations binaires et à les comparer.

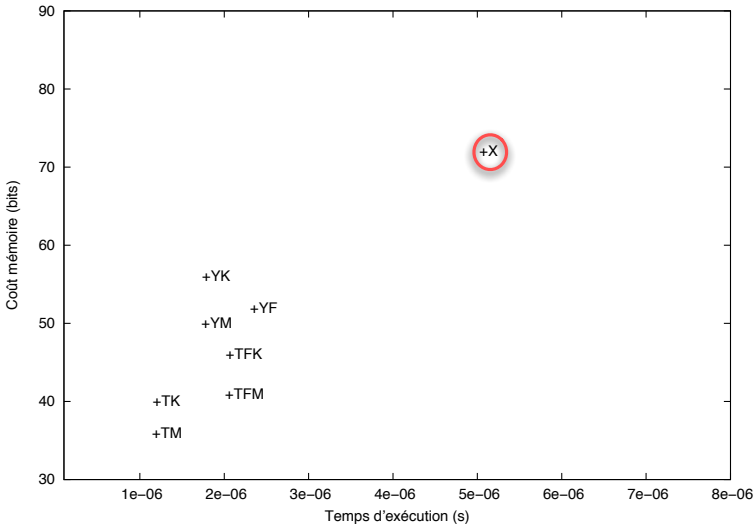


FIG. 6.20 – Coût mémoire en fonction du temps d'exécution des combinaisons de critères qui arrivent à une différenciation totale.

6.6 Conclusion

Dans ce chapitre, un ensemble de critères de différenciation ont été présentés et classés en deux catégories : les critères basés sur la région et ceux basés sur le contour.

L'application de l'outil ECO sur l'ensemble exhaustif des représentations binaires a produit des résultats qui nous ont permis de choisir les meilleurs critères de différenciation en terme de résultat de différenciation, du coût mémoire et du temps d'exécution. Nous notons, à la suite des résultats obtenus, que pour des matrices de taille 3×3, une différenciation totale peut être obtenue avec une combinaison de deux critères. Par contre, pour des matrices de taille 4×4, une combinaison de sept critères est nécessaire pour une différenciation totale.

Ensuite, nous avons effectué une étude comparative entre nos critères de différenciation et la méthode Grid-based. Notre méthode de comparaison par critères a donné de bons résultats de différenciation et s'est avérée moins coûteuse et plus rapide. Ce qui montre qu'elle peut être utilisée pour la différenciation des pièces. Mais sur un cas réel cela est-il toujours vrai ? C'est l'objet du prochain chapitre.

CHAPITRE 7

Sélection du nombre optimal de capteurs de la Smart Surface

Sommaire

7.1 Introduction

Nous allons présenter dans ce chapitre un ensemble d'expérimentations réalisées en utilisant l'outil SNC (*Sensor Network Calibrator*). Nous rappelons que cet outil nécessite la définition de modèles de pièces, par conséquent, nous avons utilisé quatre modèles de pièces classiques (voir figure 7.1(a)) afin d'effectuer des expérimentations sur des pièces réelles posées sur le Prototype de la Smart Surface.

Nous allons décrire ici chaque étape de ces expérimentations en débutant avec la phase de pré-traitement.

7.2 Phase de pré-traitement

Avant l'utilisation de notre outil sur tous les modèles de pièces, il nous faut d'abord créer une base de données pour chaque modèle de pièce. Pour cela , on positionne chaque modèle de pièce sur le Prototype de la Smart Surface et grâce à une caméra positionnée au-dessus, une acquisition d'image est effectuée. Étant donné que les modèles de pièces sont en 3 dimensions, nous avons choisi de positionner les modèles de pièces sur le Prototype de Smart Surface de façon à toujours obtenir une même image pour chaque modèle de pièce. Bien sûr si l'on choisi

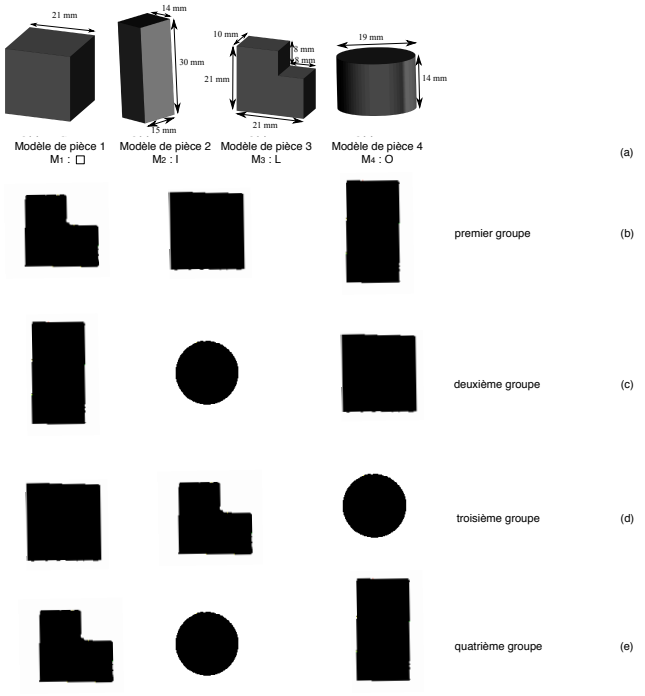


FIG. 7.1 – (a) Définition de nos modèles de pièces. (b)–(e) Tous les groupes de trois modèles de pièces.

de mettre le modèle de pièce dans n'importe quelle orientation, il nous faudra alors pour chaque modèle appliquer l'algorithme de pré-traitement pour toutes les images (facettes) du modèle de pièce. On génère alors tous les groupes de trois images de modèles de pièces((voir figure 7.1(b-e))). Puis il reste alors à appliquer l'algorithme de pré-traitement décrit dans la section 5.7 sur chaque groupe des images de modèles de pièces.

7.2.1 Hypothèse de travail de la phase de pré-traitement

L'algorithme de la phase de pré-traitement (voir section 5.7) a été appliqué comme suit :

1. acquisition d'images M_i des modèles de pièces. Les images ont une taille de 550×550 pixels ;
2. génération de toutes les rotations des images M_i avec un pas de rotation de $d_{Rot} = 1°$;
3. génération de toutes les translations suivant l'axe des abscisses avec un pas de translation $d_x = \frac{550}{NbCapteurs_x \times 2}$;
4. génération de toutes les translations suivant l'axe des ordonnées avec un pas de translation $d_y = \frac{550}{NbCapteurs_y \times 2}$;
5. discrétisation des images en utilisant les tailles de grille de capteurs suivantes : $(15, 15), (20, 20), ..., (50, 50)$;
6. sauvegarde des représentations binaires uniques de ces images ;
7. calcul des valeurs uniques de critères pour chaque taille de grille de capteurs.

Après application de l'algorithme de la phase de pré-traitement, il en résulte une base de données sur les caractéristiques des modèles de pièces. Cette base consiste en un ensemble de valeurs associées aux différents critères de chaque modèle de pièce.

La figure 7.2 représente un exemple des résultats obtenus en appliquant la phase de pré-traitement sur les modèles de pièces : $M_2 = I$ (une pièce représentant la lettre I), $M_3 = L$ (une pièce représentant la lettre L) et $M_4 = O$ (une pièce représentant la lettre O). Ces résultats concernent le groupe de la figure 7.1(e) avec les critères de différenciation : D (la somme des 1 se trouvant sur les deux diagonales), C (la somme du nombre d'angles de forme V) et L (la distance (longueur) maximale de la pièce). Pour chaque modèle de pièce est également présenté un tableau récapitulatif contenant les différents résultats de critères obtenus.

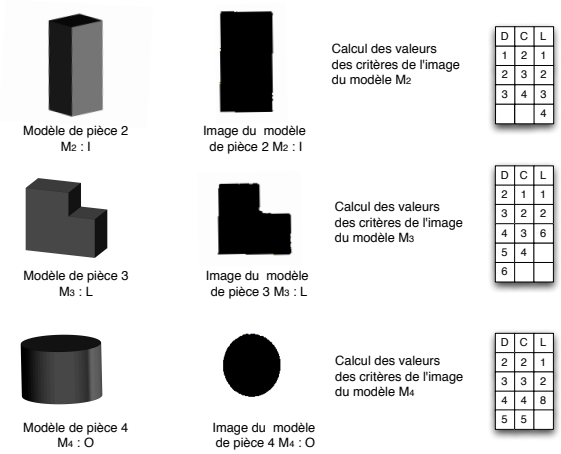


FIG. 7.2 – Exemple des valeurs des critères obtenues pour un groupe de modèles de pièces.

Les résultats obtenus dans la phase de pré-traitement vont être utilisés dans la phase temps réel pour différencier les pièces sur la Smart Surface.

7.3 Phase temps réel

La phase temps réel consiste à différencier la pièce qui est sur la Smart Surface, en calculant les critères associés à la pièce et en comparant les résultats obtenus avec les valeurs des critères des modèles de pièces calculées dans la phase de pré-traitement.

7.3.1 Construction de l'arbre

Avant de commencer la phase de différenciation, nous allons construire l'arbre des résultats obtenus lors de la phase de pré-traitement.

La figure 7.3 représente un arbre généré avec les résultats de la figure 7.2. Cet arbre permet d'avoir une meilleure représentation des résultats des critères obtenus.

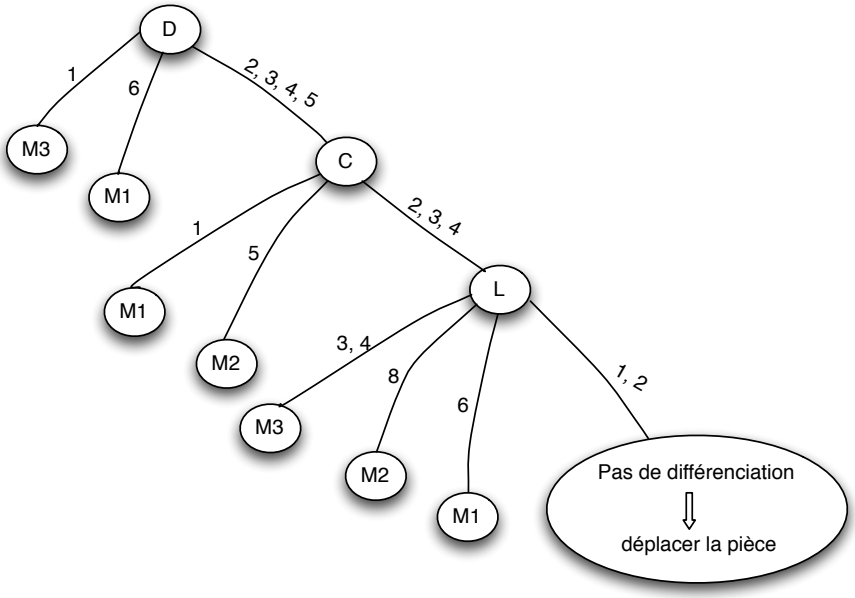


FIG. 7.3 – Exemple d'arbre généré à partir des valeurs de critères en fonction de la combinaison de critères et du groupe de modèles de pièces.

7.3.2 Hypothèse de travail de la phase temps réel

Pour chaque groupe de modèles de pièces, la taille de la grille de capteurs qui permet de différencier les modèles de pièces est déterminée. Cela se fait en calculant le taux de différenciation de chaque pièce sur la Smart Surface avec chaque modèle de pièce, c'est-à-dire :

1: **for** chaque taille de la grille de capteurs **do**
2: **for** chaque $M_i \subset$ modèles de pièces du groupe **do**
3: calculer le taux de différenciation pour M_i
4: **end for**
5: **end for**

Taille de la grille

L'algorithme de la phase de pré-traitement a été appliqué sur le groupe de modèles de pièces {L,O,I} (voir figure 7.1(e)).

La figure 7.4 représente le résultat du taux de différenciation des modèles de pièces L, I et O dans le groupe {L,O,I}. La taille de la grille de capteurs doit être déterminée de façon à arriver au meilleur taux de différenciation pour tous les modèles de pièces du groupe. Pour cela, la moyenne[24] du taux de différenciation de nos modèles de pièces a été calculée. La figure 7.4 montre le taux de différenciation du groupe {L,O,I,} et sa moyenne. Elle montre que le plus grand taux de différenciation est obtenu avec une grille de 35×35 capteurs.

[24]Nous considérons que les modèles de pièces ont la même probabilité (33%) d'être sur la surface, sinon un poids doit être utilisé pour calculer la moyenne.

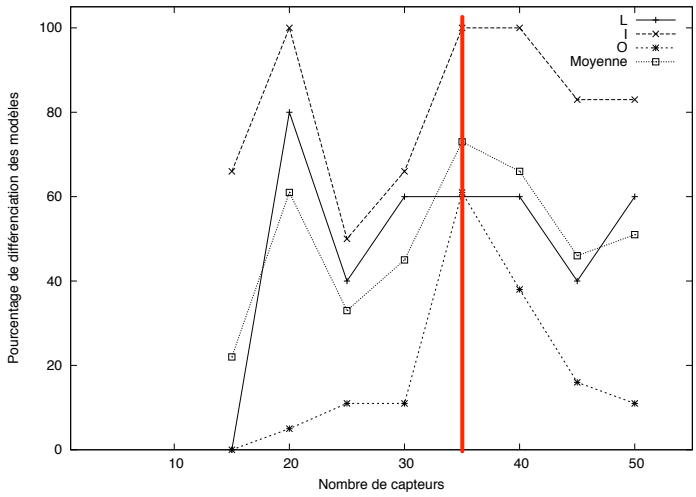


FIG. 7.4 – Pourcentage de différenciation des modèles de pièces {L,O,I} et leur moyenne de différenciation.

Seuil de différenciation

Avec la taille de grille de capteurs (35×35), nous avons obtenu les résultats suivants :
- pour le L, le taux de différenciation est 60% ;
- pour le O, le plus grand taux de différenciation est 61.11% ;
- pour le I, le plus grand taux de différenciation est 100%.

Logiquement nous pensions que le taux de différenciation augmenterait en fonction du nombre de capteurs, mais la courbe de la figure 7.4 montre que le taux augmente et décroît. Nous pensons que ceci s'explique par l'augmentation de l'information des capteurs, qui implique l'augmentation des représentations binaires.

Si l'intersection entre les valeurs de critères des différents modèles de pièces augmente plus vite que l'union alors le $Rapport_{NonDiff}$ augmente ce qui implique que le taux de différenciation décroît. Soit :

M_1 et M_2 les modèles de pièces (voir figure 7.1(a)).

$n = \{15, 20, 25, 30, ..., 60\}$ le nombre de capteurs ;

$r = 1°$ le pas de rotation des images ;

$x = Pas_{trans_X} = \frac{550}{NbCapteurs_X \times 2}$ le pas de translation suivant l'axe des abscisses ;

$y = Pas_{trans_Y} = \frac{550}{NbCapteurs_Y \times 2}$ le pas de translation suivant l'axe des ordonnées ;

$f(M_1, n, r, x, y), f(M_2, n, r, x, y)$: l'ensemble des représentations binaires associées respectivement aux modèles de pièces M_1 et M_2 ;

$C_1(f(M_1, n, r, x, y)), C_1(f(M_1, n, r, x, y))$: l'ensemble des valeurs du critère $C_1 = \{c\}$ (voir section 6.2.15) pour toutes les représentations binaires $f(M_1, n, r, x, y)$ et $f(M_2, n, r, x, y)$ associées respectivement aux modèles de pièces M_1 et M_2.

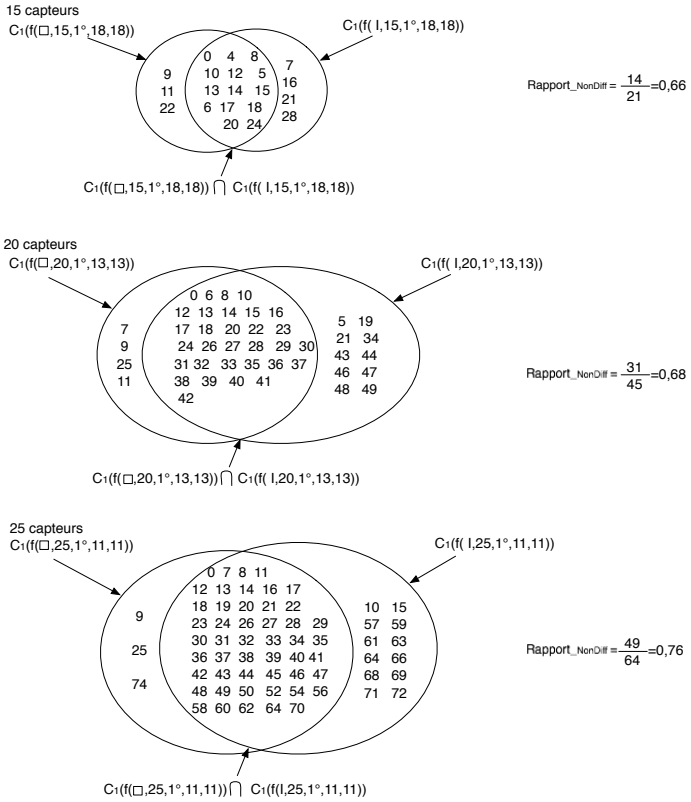


FIG. 7.5 – Exemple des valeurs du critère {c} associées aux modèles de pièces M_1 et M_2 en fonction du nombre de capteurs.

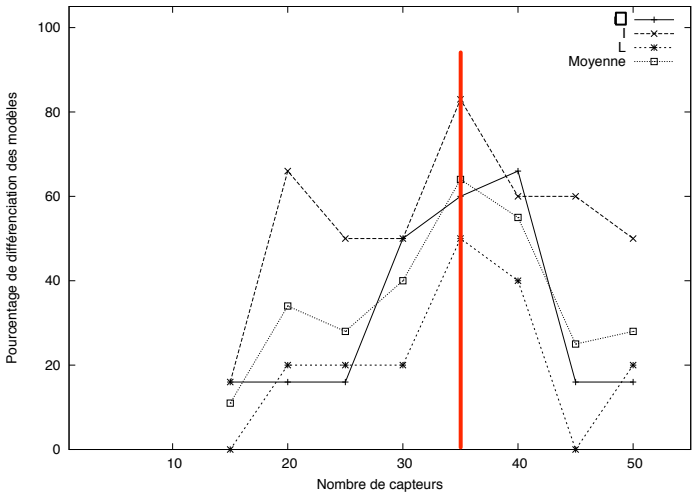


FIG. 7.6 – Pourcentage de différenciation des modèles de pièces L, □, I et leur moyenne de différenciation.

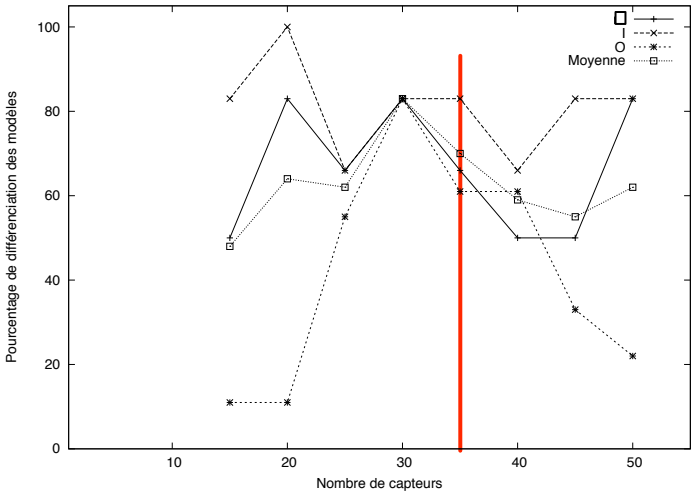


FIG. 7.7 – Pourcentage de différenciation des modèles de pièces I, O, □ et leur moyenne de différenciation.

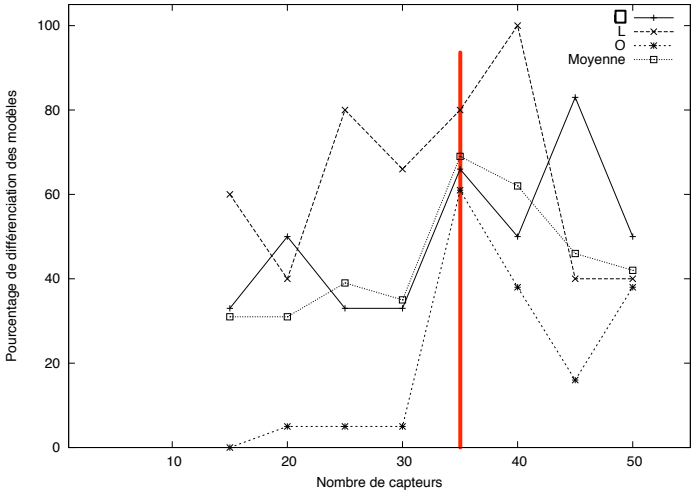


FIG. 7.8 – Pourcentage de différenciation des modèles de pièces □, L, O et leur moyenne de différenciation.

La figure 7.5 présente un exemple des résultats obtenus en appliquant le critères {c} sur les modèle $M_1 = \square$ et $M_2 = I$ et cela pour :

NbCapteurs=15, r= 1°, $x = \frac{550}{15\times2} = 18$ et $y = \frac{550}{15\times2} = 18$

NbCapteurs=20, r= 1°, $x = \frac{550}{20\times2} = 13$ et $y = \frac{550}{20\times2} = 13$

NbCapteurs=25, r= 1°, $x = \frac{550}{25\times2} = 11$ et $y = \frac{550}{25\times2} = 11$

Les résulats obtenus montrent que plus le nombre de capteurs augmente plus l'intersection $(C_1(f(M_1, n, r, x, y)) \cap C_1(f(M_2, n, r, x, y)))$ entre les valeurs des critères augmente. Cette intersection augmente plus rapidement que l'union ce qui implique l'augmentation du $Rapport_{NonDiff}$.

Les figures 7.6, 7.7 et 7.8 montrent le taux de différenciation de chaque modèle de pièce parmi chaque groupe de modèles de pièces : {L, □, I}, {I, O, □} et {□, L, O} (voir respectivement les figures 7.1(b), 7.1(c) et 7.1(d)) ainsi que leur moyenne de taux de différenciation pour chaque groupe.

En nous basant sur les valeurs des figures 7.6, 7.7 et 7.8, nous nous sommes rendus compte qu'en moyenne un bon taux de différenciation est atteint avec une grille de capteurs de taille $(35, 35)$, car il permet d'obtenir la plus grande moyenne de taux de différenciation pour l'ensemble des quatre modèles de pièces définis (voir figure 7.1(a)).

7.4 Conclusion

L'outil SNC nous a permis de trouver une taille optimale de la grille de capteurs qui est de l'ordre de 35×35. Avec cette taille, nous avons obtenu le taux de différenciation le plus élevé pour certains modèles de pièces.

L'augmentation de la taille ne permet pas, contrairement à ce qu'on pourrait croire, d'augmenter le taux de différenciation, bien au contraire ce taux devient instable. Cette variation est expliquée par l'augmentation rapide de l'intersection des valeurs des critères par rapport à leurs unions.

Ce résultat est très important et montre la nécessité d'avoir réalisé cet outil et ces expérimentations.

Nous sommes conscients que 35×35 capteurs est un paramètre propre aux modèles de pièces que nous avons utilisés. Pour valider ce nombre il nous faut alors réaliser des expériences sur d'autres types de modèles de pièces.

Conclusion

Nous arrivons au terme de cette thèse. Il nous faut fixer les éléments que nous avons apportés pour le développement et la conception d'une Smart Surface qui est un réseau de micro-modules. Chaque micro-module est composé d'un micro-capteur, d'un micro-actionneur et d'une unité de traitement. Nous nous sommes particulièrement concentrés sur la gestion et l'organisation d'information dans un réseau de micro-modules. Il nous a fallu répondre à quelques questions primordiales quant au fonctionnement optimal de la Smart Surface.

Est-il possible de différencier les pièces sur la Smart Surface ? Arrive-t-on à une différenciation totale d'un ensemble de pièces générées exhaustivement ? Combien de capteurs faut-il embarquer sur la Smart Surface pour arriver à différencier les pièces et obtenir son fonctionnement optimal ?

Pour mener à bien ces recherches et répondre le plus efficacement possible à ces questions, nous avons organisé notre travail comme suit :

1. La première étape a consisté à trouver le premier paramètre de la Smart Surface. Ce dernier avait pour but de trouver les meilleurs critères pouvant différencier toutes les pièces d'une certaine taille. Pour cela, nous avons développé l'outil ECO (*Exhaustive Comparison Framework*) qui s'est avéré crucial pour déterminer les critères de différenciation. Il nous a permis de définir les meilleurs critères de différenciation pour un ensemble de pièces de taille 3×3 et 4×4 capteurs.
2. La deuxième étape a consisté à déterminer le nombre de capteurs nécessaires au bon fonctionnement de la Smart Surface. Dans ce but, nous avons élaboré l'outil SNC (*Sensor Network Calibrator*) qui nous a permis de constater que, contrairement à ce que nous pensions, l'ajout d'information n'améliore pas forcement le résultat de différenciation. De là, s'est déduit le calcul d'un rapport de non différenciation R_NoDiff qui nous permet de déterminer le nombre de capteurs optimal.

Tout au long de cette thèse, nous avons travaillé en collaboration avec des équipes venant de domaines scientifiques différents et intervenants dans le projet Smart Surface. Cette collaboration a nécessité de faire face à plusieurs difficultés telles que la diversité des langages, les connaissances de chaque équipe et les verrous spécifiques dans chaque domaine. Il en résulte une collaboration avec des compromis, car ce qui est une condition pour un domaine peut être considérée comme un inconvénient pour un autre domaine. Par exemple, pour l'informatique, une des problématiques

majeure est d'essayer de réduire, d'optimiser au maximum le temps de calcul, ce qui nous a amené à réfléchir sur l'optimisation du temps de réponse de la différenciation, particulièrement pour la convergence de l'algorithme de reconstitution de la pièce. Bien sûr cela entraîne un temps de réponse moyen plus court mais différent d'une pièce à une autre. Pour l'automatique, ceci ne présente aucun avantage. Il est, en effet, préférable d'avoir un fonctionnement cyclique, c'est-à-dire un temps de réponse identique à chaque fois, d'une pièce à une autre, même s'il n'est pas optimal.

Il est également intéressant de constater que le temps de développement nécessaire varie considérablement d'un domaine à un autre. Le temps de développement en informatique est beaucoup plus court en comparaison avec le temps de développement nécessaire dans le domaine de l'automatique. Et le temps de développement nécessaire dans le domaine de l'automatique reste plus court que le temps de développement dans le domaine de fabrication technologique qui nécessite beaucoup plus de moyen pour développer, concevoir ou réaliser techniquement un composant.

Cette collaboration a nécessité de fréquentes réunions pour éviter de déterminer des conditions qui pouraient être des contraintes pour d'autres équipes. Lors de l'élaboration de cette thèse, j'ai travaillé en proche collaboration avec l'équipe d'automatique du laboratoire de FEMTO-ST (*Département AS2M-Automatique et Systèmes Micro-Mécatronique*). Cela m'a permis, d'une part, de tester nos algorithmes directement sur la plate-forme de la Smart Surface développée au sein de ce laboratoire, et d'autre part de fusionner notre travail sur la différenciation avec leur travail sur la commande distribuée des actionneurs.

De nos jours, tous les projets de recherche ont tendance à s'orienter de plus en plus vers un partenariat multidisciplinaires, où des équipes de différents domaines inter-agissent pour aboutir au final à un composant fonctionnel.

Les perspectives à court terme de ce travail de thèse seront d'essayer de réduire encore plus la taille de la combinaison de critères de différenciation, de continuer à travailler sur la convergence de l'algorithme de reconstruction de la pièce et aussi de gérer les erreurs de capteurs. Il nous faudrait par la suite tester notre outil SNC sur d'autres modèles afin de valider le nombre de capteurs à intègrer sur la Smart Surface. Puis, utiliser l'outil SNC pour trouver s'il existe une relation entre la taille de la pièce et le nombre de capteurs à positionner sur la Smart Surface. Si cette relation existe, il faudra la formaliser et trouver le moyen de l'exprimer par une formule simple.

Actuellement, la différenciation est réalisée après la phase de recontruction de l'image. Il serait intéressant d'utiliser des méthodes statistiques afin de permettre de différencier la pièce pendant sa phase de reconstitution et cela en utilisant les vues intermédiaires des capteurs sur les modèles.

Les perspectives à long terme seraient de proposer la Smart Surface pour le tri de micro-pièces très sensibles dans le monde de l'industrie par exemple dans une chaîne de montage pour l'horlogerie ou toute autre chaîne de montage qui nécessite la manipulation de micro-pièces.

Ces dernières années, on retrouve beaucoup d'applications de micro- et nano-manipulation dédiées à la biologie (ADN, cellules, gouttes) [75]. Ces nouveaux systèmes sont basés sur des phénomènes physico-chimique comme la diélectrophorèse [76], l'électrowetting ou l'électrowetting-

on-dielectric (EWOD) [77, 78], etc. Ces nouveaux domaines sont en pleine évolution. La Smart Surface pourrait alors être proposée pour manipuler des éléments biologiques pour ces domaines.

En nous inspirant des travaux de [79] nous pouvons aussi utiliser notre réseau de capteurs comme une caméra très rapide où chaque capteur pouvant communiquer avec ses voisins est considéré comme un pixel.

Publications

Revues internationales avec comité de lecture

– Kahina Boutoustous, Eugen Dedu and Julien Bourgeois. An Exhaustive Comparison Framework for Distributed Shape Differentiation in a MEMS Sensor Actuator Array. In International Journal of Signal and Imaging Systems Engineering (IJSISE), soumis en 2009

– Kahina Boutoustous, Eugen Dedu and Julien Bourgeois. A framework to Calibrate a MEMS Sensor Network. In Journal of Ubiquitous Computing and Intelligence (JUCI), soumis en 2008.

Revues nationales avec comité de lecture

– Kahina Boutoustous, Eugen Dedu and Julien Bourgeois. Aide au paramétrage d'une grille de capteurs/actionneurs MEMS. In Technique et Science Informatiques (TSI), soumis en 2009.

Conférences internationales avec comité de lecture et actes

– Kahina Boutoustous, Eugen Dedu and Julien Bourgeois. A framework to Calibrate a MEMS Sensor Network. LNCS. In International Conference on Ubiquitous Intelligence and Computing (UIC), pages 136 - 149, Brisbane, Australia, 2009.

– Kahina Boutoustous, Eugen Dedu and Julien Bourgeois. An Exhaustive Comparison Framework for Distributed Shape Differentiation in a MEMS Sensor Actuator Array. IEEE Computer Society Press. In International Symposium on Parallel and Distributed Computing (ISPDC), pages 429 - 433, Kraków, Poland, 2008.

Conférences nationales avec comité de lecture et actes

– Kahina Boutoustous, Eugen Dedu and Julien Bourgeois. Un calibrateur pour un réseau de capteurs MEMS. In Journées doctorales en informatique et réseaux (JDIR), pages 122-127, Belfort, France, février 2009.

Conférences internationales sans comité de lecture et actes

– Kahina Boutoustous, Eugen Dedu and Julien Bourgeois. Distributed information management. In Connecting Women and Completing the Circuit of International Research Collaboration (WIRES). Barcelona, Spain, juin 2009.

BIBLIOGRAPHIE

[1] Karl-Friedrich Böhringer, Bruce Randall Donald, and Noel C. MacDonald. Upper and lower bounds for programmable vector fields with applications to MEMS and vibratory plate parts feeders. In *International Workshop on Algorithmic Foundations of Robotics*, pages 255–276, 1996.

[2] Karl-Friedrich Böhringer, Bruce Randall Donald, Lydia E. Kavraki, and Florent Lamiraux. Part orientation with one or two stable equilibria using programmable force fields. *Transactions On Robotics And Automation*, 16, April 2000.

[3] Lennart Löfdahl and Mohamed Gad el Hak. MEMS applications in turbulence and flow control. *Progress in Aerospace Sciences*, 35(2) :101– 203, 1999.

[4] Manabu Ataka, Akito Omodaka, Naohiro Takeshima, and Hiroyuki Fujita. Fabrication and operation of polyimide bimorph actuators for a ciliary motion system. *Microelectromechanical Systems*, 2 :146–150, August 2002.

[5] John W. Suh, R. Bruce Darling, Karl-Friedrich Böhringer, Bruce Randall Donald, Henry Baltes, and Gregory T. A. Kovacs. CMOS integrated ciliary actuator array for general-purpose micromanipulation tool for small objects. *IEEE Journal of Microelectromechanical Systems*, 8 :483–496, 1999.

[6] Karl-Friedrich Böhringer, Bruce Randall Donald, and Noel C. MacDonald. Single-crystal silicon actuator arrays for micro manipulation tasks. In *IEEE Workshop on Micro Electro Mechanical Systems MEMS*, pages 7–12, 1996.

[7] Jonathan Luntz, William Messner, and Howie Choset. Virtual vehicle : Parcel manipulation and dynamics with a distributed actuator array. In *International Symposium on Intelligent Systems and Advanced Maufacturing Sensors and Controls for Advanced Manufacturing*, pages 1541–1546, 1997.

[8] Tetsuhiko Iizuka and Hiroyuki Fujita. Precise positioning of a micro conveyor based on superconducting magnetic levitation. *International Symposium on Micromechatronics and Human Science*, pages 131–135, June 1997.

[9] Haruo Nakazawa, Yasumasa Watanabe, Osamu Morita, Masaharu Edo, and Eiichi Yonezawa. The two-dimensional micro conveyer : principles and fabrication process of the

actuator. *International Conference on Solid-State Sensors and Actuators*, 1 :33–36, June 1997.

[10] Andrew Berlin, David Biegelsen, Patrick Cheung, Markus Fromherz, and David Goldberg. Objects using large-area arrays of MEMS-like distributed manipulators. *Micromechatronics*, 2000.

[11] Kristofer S.J. Pister, Ronald Fearing, and Roger T. Howe. A planar air levitated electrostatic actuator system. In *IEEE Micro Electro Mechanical Systems. An Investigation of Micro Structures, Sensors, Actuators, Machines and Robots*, pages 61–71, 1990.

[12] Satoshi Konishi and Hiroyuki Fujita. A conveyance system using air flow based on the concept of distributed micro motion systems. *Journal of microelectromechanical systems*, 3 :54–58, June 1994.

[13] Babu M. Mehtre, Mohan S. Kankanhalli, and Wing F. Lee. Shape measures for content based image retrieval : a comparaison. *Information Processing and Management*, 33(3) :319–337, May 1997.

[14] Yi Tao and William I. Grosky. Object-based image retrieval using point feature maps, 1999.

[15] Maytham Safar, Cyrus Shahabi, and Xiaoming Sun. Image retrieval by shape : a comparative study. In *IEEE International Conference on Multimedia and Exposition*, pages 141–144, 1999.

[16] Dengsheng Zhang and Guojun Lu. Content-based shape retrieval using different shape descriptors : a comparative study. In *International Conference on Multimedia and Expo*, pages 1139–1142, Los Alamitos, CA, USA, 2001.

[17] Dengsheng Zhang and Guojun Lu. Study and evaluation of different Fourier methods for image retrieval. *Image and Vision Computing*, 23(1) :33–49, 2005.

[18] Diane Lingrand. *Introduction au Traitement d'Images*. Vuibert, Paris, France, 2nd edition, February 2008.

[19] Stéphane Bres, Jean-Michel Jolion, and Frank Lebourgeois. *Traitement et analyse des images numériques*. Hermès, 2003.

[20] Atul Sajjanhar and Guojun Lu. A Grid-based shape indexing and retrieval method. *Australian Computer Journal*, 29(4) :131–140, 1997.

[21] Steve Bourgeois, Sylvie Naudet, and Michel Dhome. Descripteur de contours pour la reconnaissance d'objets. *Congrès des jeunes chercheurs en vision par ordinateur*, May 2005.

[22] Dan S. Reznik. *The Universal Planar Manipulator*. PhD thesis, University of California at Berkeley, 2000.

[23] Karl-Friedrich Böhringer, Vivek Bhatt, Bruce Randall Donald, and Kenneth Y. Goldberg. Algorithms for sensorless manipulation using a vibrating surface. *Algorithmica*, 26(3-4) :389–429, 2000.

[24] Karl-Friedrich Böhringer, Bruce Randall Donald, and Noel C. MacDonald. Programmable force fields for distributed manipulation, with applications to MEMS actuator arrays and vibratory parts feeders. *The International Journal of Robotics Research*, 18 :168–200, 1999.

[25] Kenneth Y. Goldberg. Orienting polygonal parts without sensors. *Algorithmica*, 10(2-4) :210–225, 1993.

[26] Karl-Friedrich Böhringer. *Programmable Force Fields For Distributed Manipulation And Their Implementation Using Micro-Fabricated Actuator Arrays.* PhD thesis, The Faculty of the Graduate School of Cornell University, 1997.

[27] Ronald S. Fearing. Survey of sticking effects for micro parts handling. In *International Conference on Intelligent Robots and Systems*, volume 2, pages 212–217, August 1995.

[28] Y. Zhou and B. J. Nelson. Adhesion force modeling and measurement for micromanipulation. In *International Conference on Society of Photo-Optical Instrumentation Engineers.*, volume 3519, pages 169–180, October 1998.

[29] C. Girardand, D V. Labeke, and J. M. Vigoureux. Van der waals force between a spherical tip and a solid surface. *Physical Review*, 40(18), December 1989.

[30] Yassine Haddab. *Conception et réalisation d'un système de micromanipulation contrôlé en effort et en position pour la manipulation d'objets de taille micrométrique.* PhD thesis, l'Université de Franche-Comté, 2000.

[31] Stéphane Senturia. *Microsystem design.* Kluwer Academic Publishers, November 2000.

[32] Andrew A. Berlin and Kaigham J. Gabriel. Distributed MEMS : new challenges for computation. *Computational Science and Engineering, IEEE*, 4 :12–16, January 1997.

[33] Peter F. Van Kessel, Robert E. Meier, and Michael R. Douglass. A MEMS-based projection display. In *Proceedings of the IEEE*, volume 86, pages 1687–1704, August 1998.

[34] Douglas R. Sparks, Shih-Chia Chang, and David S. Eddy. Application of MEMS technology in automotive sensors and actuators. In *International Symposium on Micromechatronics and Human Science*, pages 1747–1755, 1998.

[35] Hiroyuki Fujita. Sensors and actuators. *Journal of microelectromechanical systems*, 56 :105–111, August 1996.

[36] Y. Fukuta, Y.-A. Chapuis, Y. Mita, and H. Fujita. Design, fabrication and control of MEMS-based actuator arrays for air-flow distributed micromanipulation. *IEEE Journal of Micro-Electro-Mechanical Systems*, 15(4) :912–926, August 2006.

[37] J. W. Suh, R. B. Darling, K-F. Böhringer, B. R. Donald, H. Baltes, and G. T. Kovacs. Fully programmable MEMS ciliary actuator arrays for micromanipulation tasks.

[38] John W. Suh, Steven F. Glander, Robert B. Darling, Christopher W. Storment, and Gregory T. A. Kovacs. Combined organic thermal and electrostatic omnidirectional ciliary microactuator array for object positioning and inspection. In *Solid State Sensor and Actuator Workshop*, volume 58, pages 51–60, 1996.

[39] Karl-Friedrich Böhringer, Bruce Randall Donald, and Noel C. MacDonald. What programmable vector fields can (and cannot) do : force field algorithms for MEMS and vibratory plate parts feeders. In *International Conference on Robotics and Automation*, pages 822–829, 1996.

[40] Jonathan E. Luntz and William Messner. A distributed control system for flexible materials handling. *Control Systems Magazine, IEEE*, 17(1) :22–28, February 1997.

[41] David Biegelsen, Andrew Berlin, Patrick Cheung, Markus Fromherts, David Goldberg, Warren Jackson, Bryan Preas, James Reich, and Lars Swartz. Air jet paper mover. In *International Symposium on Micromachining and Microfabrication*, 2000.

[42] Hiroyuki Fujita. Group work of microactuators. In *International Advanced Robot Program Workshop on Micromachine Technologies and Systems*, pages 24–31, Tokyo, Japan, October 1993.

[43] Thorbjörn Ebefors and Göran Stemme. *Microrobotics*, chapter 28. The MEMS Handbook, CRC Press LLC edition, 2002.

[44] Thorbjörn Ebefors. *Polyimide V-Groove Joints For Three-Dimensional Silicon Transducers*. PhD thesis, Instrumentation Laboratory Department Of Signals, Sensors and Systems (S3) Royal Institute Of Technology, 2000.

[45] Chang Liu, Thomas Tsao, Yu-Chong Tai, Wenheng Liu, Peter H. Will, and Chih-Ming Ho. A micromachined permalloy magnetic actuator array for micro robotics assembly systems. In *8th International Conference on Solid-State Sensors and Actuators*, 1995.

[46] T. Hirata, T. Akashi, A. Bertholds, H. P. Gruber, A. Schmid, M.-A. Gratillat, O. Guenat, and N. F. de Rooij. A novel pneumatic actuator system realised by microelectro-discharge machining. In *11th Annual International Workshop on Micro Electro Mechanical Systems*, pages 160–165, 1998.

[47] Anil K. Jain, Robert P.W. Duin, and Jianchang Mao. Statistical pattern recognition : a review. *IEEE Transactions On Pattern Analysis Ans Machine Intelligence*, 22(1), January 2000.

[48] Dengsheng Zhang and Guojun Lu. Review of shape representation and description techniques. *Pattern Recognition*, 37(1) :1–19, January 2004.

[49] S.A. Tabbone, L. Wendling, and J.P. Salmon. A new shape descriptor defined on the radon transform. *Computer Vision Image Understanding*, 102(1) :42–51, April 2006.

[50] Jean-Pierre Salmon, Laurent Wendling, and Salvatore Tabbone. Automatical definition of measures from the combination of shape descriptors. In *8th International Conference on Document Analysis and Recognition*, volume 2, pages 986–990, Seoul/Korea, August 2005.

[51] Andre Folkers and Hanan Samet. Content-based image retrieval using Fourier descriptors on a logo database. In *International Conference on Pattern Recognition*, volume 3, pages 521–524, 2002.

[52] Hao Zhang and Eugene Fiume. Shape matching of 3-D contours using normalized Fourier descriptors. *International Conference on Shape Modeling and Applications*, pages 261–268, 2002.

[53] Sven Loncaric. A survey of shape analysis techniques. *Pattern Recognition*, 31(8) :983–1001, 1998.

[54] Dengsheng Zhang and Guojun Lu. Shape-based image retrieval using generic Fourier descriptor. *Signal processing Image communication*, 17(10) :825–848, November 2002.

[55] Rocio Diaz de Leon and Luis Enrique Sucar. Human silhouette recognition with Fourier descriptors. In *International Conference on Pattern Recognition*, volume 3, pages 709–712, 2000.

[56] Stephane Derrode, Mohamed Daoudi, and Faouzi Ghorbel. Invariant content-based image retrieval using a complete set of Fourier-Mellin descriptors. *International Conference on Multimedia Computing and Systems*, 2 :877–881, 1999.

[57] Per-Erik Forssén and Anders Moe. Contour descriptors for view-based object recognition. Technical report, Linköping University, Sweden, September 2005.

[58] Junding Sun and Xiaosheng Wu. Shape retrieval based on the relativity of Chain codes. In *Multimedia Content Analysis and Mining*, pages 76–84, 2007.

[59] Bribiesca Ernesto and Guzman Adolfo. Shape description and shape similarity measurement for two-dimensional regions. In *International Conference on Pattern Recognition*, pages 608–612. Paris, France, 1978.

[60] Whoi-Yul Kim and Yong-Sung Kim. A region-based shape descriptor using Zernike moments. *Signal Processing : Image Communication*, 16(1-2) :95–102, September 2000.

[61] Cyrus Shahabi and Maytham Safar. An experimental study of alternative shape-based image retrieval techniques. *Multimedia Tools and Applications*, 32(1) :29–48, 2007.

[62] Richard J. Prokop and Anthony P. Reeves. A survey of moment-based techniques for unoccluded object representation and recognition. *Graphical Models and Image Processing*, 54(5) :438–460, September 1992.

[63] Muharrem Mercimek, Kayhan Gulez, and Tarik Veli Mumcu. Real object recognition using moment invariants. *Academy Proceedings in Engineering Sciences*, 30(6) :765–775, 2005.

[64] Chaur-Chin Chen. Improved moment invariants for shape discrimination. *Pattern Recognition*, 26(5) :683–686, May 1993.

[65] A. Ashbrook and N.A. Thacker. Tutorial : Algorithms for 2-dimensional object recognition. Technical report, Imaging Science and Biomedical Engineering Division, Medical School, University of Manchester, Manchester, 1998.

[66] Ting Xia, Hongqing Zhu, Huazhong Shu, Pascal Haigron, and Limin Luo. Image description with generalized pseudo-Zernike moments. *Journal of the Optical Society of America*, 24(1) :50–59, January 2007.

[67] Lingfei Zhou. *Modélisation VHDL-AMS multi-domaines de structures intelligentes, autonomes et distribuées à base de MEMS*. PhD thesis, Université Louis Pasteur-Strasbourg, 2007.

[68] Laëtitia Matignon. *Synthèse d'agents adaptatifs et coopératifs par apprentissage par renforcement*. PhD thesis, l'Université de Franche-Comté, 2008.

[69] Kahina Boutoustous, Eugen Dedu, and Julien Bourgeois. An exhaustive comparison framework for distributed shape differentiation in a MEMS sensor actuator array. In *International Symposium on Parallel and Distributed Computing*, 7, pages 429–433, Kraków, Poland, July 2008.

[70] Kahina Boutoustous, Eugen Dedu, and Julien Bourgeois. A framework to calibrate a MEMS sensor network. In *6-th International Conference on Ubiquitous Intelligence and Computing*, LNCS, pages 136–149, Brisbane, Australia, July 2009.

[71] Kahina Boutoustous, Eugen Dedu, and Julien Bourgeois. Un calibrateur pour un réseau de capteurs MEMS. In Alexandre Caminada, editor, *Journées doctorales en informatique et réseaux*, 10, pages 122–127, Belfort, France, February 2009.

[72] Alamin Mansouri, Alexandra Lathuilière, Franck S. Marzani, Yvon Voisin, and Pierre Gouton. Toward a 3d multispectral scanner : An application to multimedia. *IEEE MultiMedia*, 14(1) :40–47, 2007.

[73] Didier El Baz, Vincent Boyer, and Rodolphe Jaumouillé. Smart Surafce. `http://www.laas.fr/SMART-SURFACE/`.

[74] Kahina Boutoustous. Résultat eco. `http://lifc.univ-fcomte.fr/page_personnelle/recherche/70`.

[75] Hyejin Moon, Sung Kwon Cho, Robin L. Garrell, and Chang-Jin Kim. Low voltage electrowetting-on-dielectric. *Journal of applied physics*, 92(7) :4080–4087, 2002.

[76] Karl-Friedrich Böhringer. Modeling and controlling parallel tasks in droplet-based microfluidic systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 25(2) :334–344, February 2006.

[77] Michael G. Pollack, Richard B. Fair, and Alexander D. Shenderov. Electrowetting-based actuation of liquid droplets for microfluidic applications. *Applied Physics Letters*, 77(11) :1725–1726, 2000.

[78] Sung Kwon Cho, Hyejin Moon, and Chang-Jin Kim. Creating, transporting, cutting, and merging liquid droplets by electrowetting-based actuation for digital microfluidic circuits. *Journal of Microelectromechanical Systems*, 12(1) :70–80, February 2003.

[79] Sándor P. Fekete, Dietmar Fey, Marcus Komann, Alexander Kröller, Marc Reichenbach, and Christiane Schmidt. Distributed vision with smart pixels. In *Proceedings of the 25th annual symposium on Computational geometry*, pages 257–266, New York, NY, USA, 2009. Association for Computing Machinery.

Résumé

Ces dernières années, de nombreux projets se sont intéressés aux systèmes MEMS permettant le convoyage de micro-pièces. Pourtant, la majorité de ces projets ont porté sur des manipulations sans retour d'information ce qui ne permet pas le tri des micro-pièces. D'autres projets qui prennent en compte ces retours d'information sont tributaires d'une unité de calcul externe, typiquement un PC. Cette dépendance limite l'autonomie du système de micro-convoyage et l'empêche de s'intégrer dans un système embarqué. Le projet Smart Surface est un partenariat de recherche qui s'est créé afin de réfléchir et de mettre en œuvre des solutions matérielles et logicielles pour les micro-systèmes distribués qui seraient applicables, par exemple, au projet de nanodrone volant de type libellule de la société Silmach (Besançon).

Ce projet réunit 5 laboratoires de recherche (FEMTO-ST, INESS, LAAS, LIFC et LIMMS) et 22 chercheurs. Son objectif est de créer une surface composée de plusieurs centaines de micro-capteurs/micro-actionneurs disposés en matrice et qui sont capables de réaliser collectivement un tri et un déplacement sans source de calcul externe.

Notre Smart Surface est un ensemble de micro-modules intelligents, où chaque micro-module est composé d'un micro-actionneur, d'un micro-capteur et d'une unité de traitement. Cette thèse intervient plus précisément dans la gestion de la coopération entre ces nombreux micro-modules pour permettre un traitement et une gestion distribuée des informations.

L'objectif de mon travail de thèse est de proposer des algorithmes permettant une différentiation optimisée des micro-pièces à partir des informations fournies par les capteurs embarqués. Notre approche consiste à proposer un environnement de test permettant d'évaluer ces algorithmes. Cet environnement permet d'évaluer automatiquement la pertinence de différents critères de différentiation par le biais de la construction automatique de graphes. Cet outil automatique appelé ECO (*Exhaustive COmparison Framework*) est la brique de base qui permet d'évaluer des algorithmes de différenciation et d'identifier les algorithmes de différenciation les plus pertinents en terme d'utilisation de mémoire et de temps processeur.

Nous avons par la suite généralisé les conditions posées en début de thèse et nous sommes maintenant capables de prendre des images réelles de la Smart Surface et de calculer les meilleurs critères pour un jeu de pièces donné. Ce travail s'effectue en collaboration proche avec le Département Automatique et Systèmes Micro-Mécatroniques (AS2M) de Femto-ST.

Mots-clés: Grille de capteurs, Différenciation de formes, Calcul distribué, MEMS.

Abstract

In recent years, many projects have focused on systems for conveying MEMS micro-parts. Yet most of these projects have focused on manipulations without feedback, this not allowing the sorting of micro-parts. Other projects that do take into account the feedback are dependent on an external computing unit, usually a PC. This dependence limits the autonomy of the micro-conveyor system and prevents it from being integrated into an embedded system. The Smart Surface project is a research partnership created in order to reflect and implement hardware and software solutions for micro-distributed systems, meant to be used, for example, in the project of the flying nanodrone of dragonfly type, developed at the Silmach company in Besançon. This project involves 5 research laboratories (FEMTO-ST, Inessa, LAAS, and LIMM LIFC) and 22 researchers. The goal is to create a surface composed of hundreds of micro-sensors/micro-actuators arranged in a matrix and that are able to collectively achieve sorting and source-displacement without external design.

Our Smart Surface is a set of intelligent micro-modules, where each micro-module is composed of a micro-actuator, a micro-sensor and a processing unit. This thesis is particularly involved in the management of cooperation between the many micro-modules in order to enable handling and distributed management of information.

The objective of my thesis is to propose algorithms for an optimized differentiation of micro-parts starting from the information provided by on-board sensors. Our approach is to propose a testing environment that will allow the evaluation of these algorithms. This environment allows to automatically evaluate the relevance of various differentiation criteria through the automatic construction of graphs. This automatic tool called ECO (*Exhaustive Comparison Framework*) is the basis for evaluating differentiation algorithms, and for identifying the most relevant differentiation algorithms in terms of memory usage and CPU time.

We have subsequently generalized the conditions imposed at the beginning of the thesis and we are now able to take actual pictures of the Smart Surface and calculate the best criteria for a given set of parts. This work is done in close collaboration with the Department of Automatics and Micro-Mechatronic Systems (AS2M) of Femto-ST.

Keywords: Sensor grid, Shape differentiation, Distributed computing, MEMS.