De l'appariement a l'indexation des images

Patrick Gros

Mémoire présenté par

Patrick GROS

pour obtenir le diplôme

d' **HABILITATION À DIRIGER DES RECHERCHES**

de l'**INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE**

(Arrêté ministériel du 23 novembre 1988)

Spécialité : **Informatique et mathématiques appliquées**

DE L'APPARIEMENT À L'INDEXATION DES IMAGES

Date de soutenance : vendredi 18 décembre 1998

Composition du jury :

Président :	**Claude PUECH**
Rapporteurs :	**Martial HÉBERT**
	Jan-Olof EKLUNDH
	Roger MOHR
Examinateurs :	**Patrick BOUTHEMY**
	Radu HORAUD

Mémoire préparé au sein du laboratoire GRAVIR – IMAG

Sommaire

Nos publications se distinguent, il est vrai,
du simple catalogue de foire.
Mais ce qui les en distingue fait justement
que personne ne les lit plus.
Lichtenberg, troisième cahier, 1775-1779.

Remerciements

Une phrase d'esprit donne toujours
l'impression que ce qui suit l'est aussi.
Bart Lamiroy

Diriger des recherches, c'est faire partager ses objectifs à plusieurs personnes, pour démultiplier sa force de travail et l'ampleur des résultats que l'on obtient. Pour savoir si l'on en est capable, il faut essayer.

Je voudrais donc tout d'abord remercier ceux qui furent les cobayes de cet essai. Dans l'ordre d'apparition, Gudrun Söcher, Marie Legendre et Marie-Hélène Malissen, Edmond Boyer, Sébastien Gelgon, Olivier Bournez, Olivier Michel, Marie-Claude Pellegrini, Sylvaine Picard, Rémi Delon, Olivier Chapelle, Franck Taverne, Yves Dufournaud, Yann Hervouet, Nagib Aouini et Jean-Yves Lahitte, Nicoleta Druga et enfin et surtout Bart Lamiroy, qui m'a le plus supporté et a aussi le plus apporté au travail présenté ici.

Le travail présenté ici est donc une œuvre collective, et le « nous » sera donc de rigueur pour faire droit à cette pluralité.

Merci ensuite à tous ceux qui m'ont entouré, au sein de l'équipe movi bien sûr, mais aussi plus largement. Je voudrais en particulier citer Roger Mohr, Radu Horaud, Françoise Veillon, Long Quan et Danièle Herzog, mais aussi les organismes qui ont mis leurs moyens à disposition, le cnrs et l'inria.

Merci enfin à ceux qui prennent le temps d'évaluer ce travail, Martial Hébert et Jan-Olof Eklundh comme rapporteurs et grands voyageurs, Claude Puech et Patrick Bouthemy comme examinateurs.

Introduction

Je l'ai faite un peu longue, et je m'en aperçois.
On va s'imaginer que c'est une préface.
Moi qui n'en lis jamais ! - ni vous non plus, je crois.
Alfred de Musset.

Généralités

La vision artificielle a pour but de permettre à des systèmes automatiques, informatiques ou robotiques, d'acquérir et de traiter des informations visuelles, en s'affranchissant au maximum des capacités visuelles d'un opérateur humain. En cela, elle diffère de l'imagerie, qui utilise aussi des images, mais qui laisse les opérations de traitement de haut niveau et d'interprétation à un opérateur humain.

Le processus mis en place pour utiliser automatiquement les images peut, dans un premier temps, se couper en deux parties distinctes : l'acquisition des images et le traitement de celles-ci.

L'acquisition des images est un domaine fort ancien. On peut considérer que la peinture en est un lointain ancêtre. Plus proches de nous, la photographie, le cinéma et la vidéo sont, entre autres, des systèmes d'acquisition. Le but est de transformer un signal physique fugitif, composé d'ondes électromagnétiques, en un signal d'une autre nature, électronique, magnétique, chimique ou informatique. Les domaines concernés par une telle opération sont nombreux : physique, instrumentation, électronique, automatique, traitement du signal, micro-informatique...

Le traitement des images consiste à transformer le signal obtenu pour en tirer des informations de niveaux d'abstraction ou de symbolisme croissants. Les informations peuvent être des mesures, des formes, ou une interprétation symbolique complète de la scène observée.

La vision par ordinateur s'inscrit dans ce deuxième grand domaine. On considère des images, représentées de manière interne à l'ordinateur par des tableaux d'entiers appelés pixels, chaque entier codant l'intensité lumineuse perçue en un point. Cette information est le résultat d'un long traitement optique et électronique dont le but est de transformer le signal optique initial en ce tableau. Ce traitement n'est pas sans effets secondaires, et l'information obtenue est très dégradée [Bey92] d'une part, et très peu structurée d'autre part.

Deux difficultés majeures apparaissent immédiatement : tout d'abord, toute mesure précise à partir du tableau que nous appellerons désormais image est délicate ; elle nécessite un calcul d'incertitude rigoureux et des méthodes robustes pour obtenir un résultat utilisable. D'autre part, la traduction des données sous une forme plus intelligible va nécessiter un travail de structuration très important.

Cela dit, l'œil humain sait, non sans mal, utiliser de telles images. Celles-ci contiennent donc nécessairement une information utile : le problème consiste à trouver les

traitements qui peuvent l'extraire.

Ce problème n'est point trivial. L'homme dispose, en effet, d'un système de vision perfectionné et performant, tant par son capteur stéréoscopique autofocus à ouverture variable que par les capacités de traitement cérébral qui lui permettent de passer d'un traitement de bas niveau à une interprétation symbolique. Par contre, l'homme ne possède pas l'expertise de sa propre vision : il est incapable d'expliquer ce qu'il fait pour reconnaître et interpréter ce qu'il voit et, en particulier, il ne sait donner aucune méthode exploitable directement comme algorithme de reconnaissance pour un calcul sur un ordinateur.

C'est l'un des buts de la psychologie cognitive de rechercher et d'expliciter les mécanismes de la vision humaine ou animale. La vision par ordinateur, quant à elle, ne vise pas directement une reproduction sur machine des mécanismes de la vision humaine. Elle vise plutôt à trouver des algorithmes permettant d'arriver à des résultats proches. À l'échelle de l'histoire de la science, c'est un domaine neuf, tirant ses outils tant de l'informatique et de l'intelligence artificielle que du traitement du signal, des mathématiques et de la géométrie ou de la physique.

Où en sommes-nous ?

La vision par ordinateur est un sujet de recherche actif depuis 40 ans environ. Bien que cela soit un peu arbitraire et artificiel, on peut séparer les travaux faits dans le domaine en deux grandes classes. D'un côté, les techniques quantitatives, de l'autre les qualitatives. Le but des premières est d'extraire principalement une information numérique des images. On trouve donc dans cette catégorie tous les travaux faits sur la géométrie, la reconstruction de scènes, l'étude du mouvement ou le suivi de trajectoires. Du côté qualitatif, on trouve des algorithmes visant plus à l'interprétation des images. Ainsi en est-il des algorithmes d'appariement, de reconnaissance, de catégorisation...

Entre les deux, on trouve les algorithmes d'extraction de primitives (points, contours, courbes...) des images dont les résultats sont à la fois qualitatifs et quantitatifs : ces algorithmes ont à la fois pour but de détecter des primitives et de les localiser.

Les succès de la vision ont été divers suivant les domaines. Bien qu'apparues récemment, les techniques géométriques ont connu un grand développement et ouvrent d'intéressants champs applicatifs, comme la métrologie. Par contre, d'autres domaines, comme celui de la reconnaissance automatique, bien que plus anciens, ne connaissent que des progrès plus lents. Quelle en est la cause?

Les approches géométriques sont basées sur l'emploi de primitives simples, principalement des points, des droites et parfois des coniques, toutes primitives dont on suppose généralement la mise en correspondance effectuée. On utilise, d'autre part, la connaissance d'un modèle décrivant l'opération de projection réalisée par les caméras. Le reste n'est que calculs, géométrie et précision. Beaucoup des difficultés rencontrées par ces méthodes sont dues à leurs données d'entrée : l'extraction des primitives et leur mise en correspondance sont causes de nombreuses erreurs ou aberrations dont souffrent les techniques géométriques. Si certaines de ces techniques sont prévues pour être robustes à ces problèmes, les échecs observés ne peuvent toutefois pas être mis au débit de ces méthodes géométriques dont il faut reconnaître le brillant succès.

À l'inverse, les méthodes d'extraction, celles d'appariement ou de reconnaissance partent souvent de l'image brute. On dispose alors d'une information en grande quantité, mais dont la nature est très différente de celle de que l'on voudrait pouvoir obtenir. On veut parler de droites approchant les contours d'un objet, voire d'objets, alors qu'on ne dispose que d'un quadrillage régulier de pixels ne contenant chacun qu'une à trois valeurs (suivant que l'image est en niveaux de gris ou en couleur). Il y a donc un véritable saut qualitatif à franchir dans le type des données : c'est là la vraie difficulté, difficulté que n'ont pas à affronter les techniques géométriques et qui expliquent pour

une grande part la différence de succès entre ces deux domaines.

Il est pourtant possible de contourner le problème. Il ne faut pas poser le problème de reconnaissance en termes de la reconnaissance d'un objet connu par son nom dans une image décrite par ses pixels. Il faut réduire au maximum le saut à faire entre les deux types de représentation, essayer de travailler avec des primitives dans les images, mais aussi de décrire les objets à reconnaître en fonction de leurs primitives. C'est cette idée qui structure le travail présenté dans ce mémoire.

Présentation du travail

Le but du travail que nous avons entrepris consiste à développer des méthodes permettant de résoudre les problèmes qualitatifs de la vision : sont concernés l'appariement et la catégorisation d'images, la modélisation et la reconnaissance d'objets. Les applications de telles techniques sont très nombreuses, à la fois comme briques de base pour d'autres techniques, par exemple les techniques géométriques, mais aussi directement.

De nombreux utilisateurs de grandes bases d'images aimeraient pouvoir faire des recherches par l'exemple : telle image ou tel fragment d'image est-il dans cette base? Quelle est l'image de la base la plus semblable à une image donnée? On retrouve de telles questions aussi bien dans les agences de photos, dont le stock comprend généralement entre 500 000 et 4 000 000 d'images, qu'à l'INA qui gère 400 000 heures d'archives de télévision ou chez des fabricants de matériels confectionnant des bases de photos de leurs produits ou aux douanes pour la vérification du copyright sur les motifs de tissu.

Les techniques développées dans le domaine de la vision pour répondre à ces besoins n'ont pas abouti pendant longtemps. De nombreuses techniques étaient basées sur le paradigme de prédiction – vérification : le processus de reconnaissance mélangeait la reconnaissance à un autre algorithme de détection ou de reconstruction qui était utilisé comme vérification. On procédait ainsi : on tentait une première reconnaissance dans l'image, reconnaissance qui était le plus souvent une mise en correspondance entre une primitive de l'image et une primitive du modèle de l'objet dont on voulait tester la présence. Cette première supposition permettait de faire de nouvelles propositions de mise en correspondance. La cohérence de ces propositions avec les premières mises en correspondance déjà effectuées était vérifiée en utilisant le processus annexe. Si cette cohérence était vérifiée, on retenait les propositions faites et on recommençait. Sinon, on cherchait à faire d'autres propositions et on recommençait.

Le principal défaut avec une telle méthode est son caractère séquentiel. Il n'est guère possible de tester la présence de plusieurs objets dans une image qu'en testant successivement chacun des modèles de ces objets. Dès qu'il y a trop de modèles à tester, le temps de calcul devient rédhibitoire.

Pour contourner ce problème, plusieurs principes ont été retenus comme bases de notre propre travail. Dans tout système de reconnaissance, le système doit avoir une connaissance préalable des objets à reconnaître. Cette information que le système doit posséder sera une information visuelle issues d'images – exemples. Selon les cas, il pourra s'agir d'une information brute, une image et les primitives qui en sont extraites, ou d'une information issue d'un apprentissage, c'est à dire d'un processus condensant une information redondante, par exemple un modèle obtenu à partir de plusieurs images.

C'est le processus de reconnaissance qui doit contraindre la manière dont est construit le modèle. Ce dernier ne doit pas être élaboré de manière indépendante, ce qui serait par exemple le cas si l'on utilisait systématiquement des modèles CAO : l'information disponible sur un objet peut être de nature très variable, structure géométrique, mécanique, électrique, thermique... alors que la reconnaissance nécessite une

information de type visuel. Il faut même que cette information visuelle soit la plus proche possible de celle qui sera utilisée dans l'image à reconnaître pour assurer le meilleur succès.

Autre principe retenu, la séparation entre la reconnaissance et les processus utilisant celle-ci doit permettre de factoriser cette reconnaissance. Le but premier de la reconnaissance est de réduire au maximum le nombre des objets dont la présence est possible dans une image : si on réduit ce nombre de un million à cent, il sera possible de visualiser ces cent possibilités ou de lancer un algorithme plus lent sur ces cent objets pour vérification. On peut considérer que l'objectif de reconnaissance aura été atteint. Il est donc nécessaire de pouvoir envisager et éliminer au plus vite de nombreuses possibilités sans les passer toutes en revue les unes après les autres. En un mot, il est impératif de pouvoir traiter les modèles de manière sublinéaire.

Enfin, nous avons choisi comme processus de base dans tout le travail la mise en correspondance des images. Ce processus consiste à chercher les éléments communs de deux images, communs au sens où ils représentent un même objet ou une même partie de la scène. C'est un processus de base difficile, car il ne dispose au départ d'aucune information a priori sur le contenu des images disponibles. En particulier, il n'est pas assuré que les deux images que l'on cherche à apparier représentent la même scène.

Basés sur ces principes, nos travaux ont portés sur :

1° l'appariement des images lui-même, pour lequel plusieurs types d'images sont envisagés ;

2° la modélisation des objets, par la détection et la modélisation des aspects principaux d'un objet ;

3° la catégorisation d'images pour modéliser des concepts plus vagues que des objets singuliers ;

4° l'indexation des images ou des modèles d'objets pour la reconnaissance.

Plan du mémoire

Le présent mémoire est organisé en 5 chapitres. Le premier chapitre fait un rapide état de l'art des techniques d'appariement et d'indexation d'images. Il situe alors notre contribution par rapport à l'état de l'art et fait état des travaux que j'ai dirigé. Les chapitres suivant fournissent un exposé technique et synthétique de nos contributions : le deuxième concerne l'appariement des images, que celles-ci soient structurées, texturées ou colorées, le troisième est consacré au problème de modélisation et de catégorisation, et la quatrième a pour sujet les problèmes d'indexation et de reconnaissance. Le dernier chapitre présente deux applications de nos travaux et décrit les perspectives ouvertes par notre travail. Diverses publications sont ajoutées en annexe et reprennent certains des points présentés.

Chapitre 1

État de l'art et contribution

Everything that can be invented has been invented.
Charles H. Duell,
Commissioner, U.S. Office of Patents, 1899.

POUR COMMENCER, ce chapitre fournit un rapide état de l'art des techniques d'appariement et d'indexation d'images. Cet état de l'art ne vise pas d'abord l'exhaustivité, mais sert à situer nos travaux par rapport à l'existant. La deuxième partie du chapitre est donc consacrée à une présentation générale de notre contribution, en détaillant en particulier les travaux que j'ai encadré.

1.1 État de l'art

1.1.1 Appariement d'images

L'appariement des images est un des difficiles problèmes de base de la vision. Il apparaît dès que l'on veut utiliser plusieurs images. Prenons, par exemple, le cas de la reconstruction : on dispose de deux images d'un objet tridimensionnel et on veut retrouver la géométrie de cet objet. On doit résoudre deux problèmes :

1. étant donné un point de l'objet, il faut retrouver sa projection dans chacune des deux images : c'est le problème de l'appariement ;
2. une fois les deux projections connues, il faut calculer la position du point de l'objet correspondant : c'est la reconstruction proprement dite.

Le premier problème se pose souvent légèrement différemment : on dispose d'un point p' dans une des deux images, qui est la projection d'un point P de la scène observée. On cherche alors la projection p'' de P dans l'autre image. Les points p' et p'' sont correspondants l'un de l'autre et on parle d'appariement ou de mise en correspondance pour désigner ce processus.

De nombreux autres processus nécessitent de pouvoir apparier des images et, en particulier, ceux de modélisation et reconnaissance présentés plus loin.

La difficulté vient du fait que l'on cherche p'' à partir de p' sans connaître P. Les deux images peuvent avoir été prises sous des points de vues différents, les conditions d'éclairage ont pu varier, sans compter le cas des objets déformables ou articulés que l'on ne considérera pas dans la suite où c'est l'objet lui même qui s'est modifié. Si le type de transformation entre les deux images est souvent connu ou prévisible, les paramètres précis de ces transformations sont eux inconnus.

Il faut ajouter à cela le fait que P et son voisinage ne peuvent être reconstitués uniquement à partir de p'. Tout cela montre que la recherche de p'' uniquement à partir de p' va être pour le moins difficile.

Il existe plusieurs solutions alternatives pour échapper à l'appariement. La première est la poursuite de primitives dans des séquences d'images (*tracking* en anglais). On considère alors des séquences d'images rapprochées. L'appariement entre les deux premières images est basé sur le fait que le mouvement apparent entre ces images est faible. De ce premier appariement, on tire un modèle du mouvement apparent des pixels, modèle que l'on utilise et met à jour avec chaque nouvelle image pour prédire la position des points dont on suit la trace dans les images. Dans l'image, on utilise des mesures de corrélation pour retrouver les points et affiner le modèle de mouvement.

D'autres auteurs proposent de résoudre certains problèmes comme la reconnaissance ou l'indexation sans appariement explicite: ils utilisent alors des mesures de ressemblance globale, basées sur les contours [HKR93] ou des histogrammes [SC96].

De nombreuses méthodes ont été proposées par le passé dans la littérature pour résoudre le problème de l'appariement. On peut tout d'abord distinguer celles destinées aux images en niveaux de gris (ou de couleur mais bien plus rarement) et celles qui utilisent des primitives extraites des images. Dans les premières, on peut chercher à apparier soit tous les points, soit seulement quelques-uns, choisis par exemple à l'aide d'un extracteur de points ou répartis de manière uniforme dans l'image. À chacun de ces points est associée une valeur, dite niveau de gris, qui code l'intensité lumineuse reçue. En reprenant la classification proposée dans [Fau93], on peut trier les méthodes en trois groupes.

Les méthodes de corrélation : il s'agit de mesurer par une corrélation la ressemblance entre deux sous-images centrées sur les points dont on veut savoir s'ils sont en correspondance. Pour cela, on considère une sous-image (c'est à dire une partie) de la première image et on recherche la sous-image la plus ressemblante dans la deuxième image. Pour limiter la taille de l'espace de recherche, on utilise soit des contraintes liées à la prise de vue, recherche uniquement le long des droites épipolaires par exemple, soit des limitations d'ordre technique et algorithmique, en n'utilisant par exemple que des sous-images rectangulaires de côtés parallèles aux axes. De telles méthodes sont utilisées depuis longtemps en photogrammétrie [KMM77, FP86] et en vision par ordinateur [Gen80]. Des améliorations sont régulièrement proposées [Kas83, Nis84, Gru85, Kas88, Ana89, Fua90].

Les méthodes de relaxation : dans ces méthodes, on commence par proposer quelques appariements. On utilise alors les contraintes déduites de ces premiers appariements pour en faire d'autres. Puis, on réitère le processus. Des algorithmes basés sur ce principe ont été proposés dans la littérature [MP76, MP79], avec des améliorations ultérieures [Gri81, Gri85, Pol85, PMF85].

Les méthodes de programmation dynamique : dans ce cas, on pose le problème de la mise en correspondance sous la forme de minimisation d'une fonction de nombreuses variables discrètes. On trouve des exemples de tels algorithmes dans [BB81, OK85].

Lorsqu'on dispose de données de plus haut niveau dans les images, telles des points de contour ou des segments de droite — on parlera dans la suite d'images structurées pour désigner ce cas — le problème est légèrement différent. Tout d'abord, les données sont moins nombreuses et plus riches d'information, elles possèdent en particulier des caractéristiques géométriques qui peuvent être utilisées. Ensuite, ces primitives sont généralement plus fiables. Par contre, il n'est pas possible d'obtenir d'appariement ou de reconstruction dense. Un autre problème est la pertinence des données: des formes

non polyédriques représentées par des segments ne peuvent, en général, être appariées, car le découpage des segments est très instable.

Diverses méthodes adaptées à ces données ont été proposées dans la littérature.

Méthode par prédiction et vérification : la démarche est similaire à celle des méthodes par relaxation. On fait des hypothèses d'appariement, puis on en déduit des contraintes qui permettent de vérifier les hypothèses faites et de proposer de nouveaux appariements. On trouve une telle méthode dans [MN85, AF87].

Méthode de recherche d'isomorphismes de sous-graphes :
en considérant les segments présents dans une image comme les arcs ou les sommets d'un graphe, on peut alors utiliser les méthodes de recherche d'isomorphismes de sous-graphes. Ces méthodes nécessitent d'avoir des graphes peu bruités et n'utilisent pas les informations géométriques disponibles. De plus, elles ont très souvent une complexité élevée. On trouvera des exemples dans [Sko88, Hér91].

Méthodes de calculs d'invariants : si on considère que les images ont été formées par projection perspective, on peut calculer des invariants projectifs pour caractériser des configurations de points et de droites et en déduire leur appariement [Wol90, Rot93]. D'autres invariants ont été suggérés pour divers autres types de configurations [FMZ90, FMZR91, MZ92, MZ93]. Mais ces méthodes ne marchent pour l'instant que pour des objets plans ou de révolution autour d'un axe ou des scènes simples.

Utilisation de quasi-invariants : l'exigence d'une invariance stricte à un groupe de transformations peut se révéler trop restrictive ou oblige à utiliser des quantités peu stables. D'où l'idée introduite par Binford [BL93] d'utiliser plutôt des quasi-invariants. Ceux-ci sont des quantités qui doivent être égales à leur équivalent dans la scène pour un point de vue au moins et être constantes au premier ordre lors d'une perturbation infinitésimale des paramètres de la projection autour de ce point de vue. On parle d'invariants forts lorsque la quantité ne varie pas au second ordre.

Binford montre que l'angle de deux segments et le rapport entre les longueurs de ces segments sont des quasi-invariants. Ces deux quantités sont largement utilisées dans notre travail. Il montre aussi que les coordonnées affines définies dans un plan de la scène fournissent autant de quasi-invariants forts.

On peut voir les quasi-invariants comme une version pragmatique des invariants. Les invariants impliquent une invariance, y compris pour les transformations les plus proches des cas singuliers, cas que l'on doit souvent éliminer en pratique car on ne dispose pas de données assez précises pour les traiter. C'est, par exemple, le cas des plans vus presque par la tranche où tous les points se retrouvent sur une droite dans l'image.

Par comparaison, les quasi-invariants ne visent qu'à une invariance locale, restreinte à des transformations « raisonnables ». Cela permet d'obtenir des formules de calcul bien plus simples, passage du birapport au rapport de longueur par exemple, et ainsi à des résultats bien plus stables. Cela en fait donc des candidats de choix pour les algorithmes d'appariement.

L'effet de pic : les idées développées par Ben-arie [BA90b] et ceux qui se sont inspiré de ses travaux constituent une introduction pragmatique et expérimentale de la notion de quasi-invariants. Ben-arie a étudié la distribution de l'angle et du rapport de longueur entre les projections de deux segments vus de tous les points d'une sphère de vue. Il arrive à la conclusion que «la densité de probabilité du rapport entre l'angle projeté et l'angle original a un pic pointu correspondant

au rapport unité, c'est à dire au point où l'angle projeté a la même valeur que l'angle original. Un phénomène de pic similaire apparaît avec la courbe de densité de probabilité du rapport des distances ; le pic correspond au point où le rapport des distances dans l'image est égal au rapport des distances correspondantes dans la scène... la probabilité qu'un angle projeté ait une valeur entre la moitié et le double de celle de l'angle original est supérieure à 84 % ! La même plage de proportion concernant les rapports de distance donne une probabilité supérieure à 86 % !»

Les statistiques ainsi établies sont ensuite utilisées pour calculer une distance pour la comparaison des angles et des rapports de longueurs. BEN-ARIE base son système de reconnaissance sur ces distances et sur une méthode de relaxation. D'autres systèmes basés sur cet effet de pic ont été proposés dans [Ols95, SP95], mais en utilisant le paradigme de prédiction – vérification où la vérification est faite par un calcul de pose.

Il est intéressant de voir comment deux approches indépendantes et fort différentes arrivent aux mêmes conclusions sur les mêmes mesures, mais via des argumentations différentes. L'étude de BEN-ARIE permet de jauger le comportement global des quasi-invariants tandis que les quasi-invariants de BINFORD montrent comment on peut calculer les quantités sujettes à l'effet de pic.

Le hachage géométrique : les techniques de hachage géométriques, introduites par LAMDAN et WOLFSON [LW88, LSW90, Wol90], ont, à l'origine, un double objectif d'appariement et d'indexation. Dans chaque image à apparier, ils considèrent successivement tous les triplets de points comme des repères affines, et calculent les coordonnées affines de tous les autres points dans ces repères. Ils essayent alors de retrouver les points de coordonnées identiques dans les deux images.

À l'usage, la complexité s'avère assez importante et réduit donc la possibilité d'utiliser la méthode de manière directe pour l'indexation d'images. En ce qui concerne l'appariement, cette complexité peut être grandement améliorée en prenant en compte la structure de l'image. Si les points utilisés proviennent de l'approximation polygonale des contours d'un objet, ces contours fournissent une information supplémentaire qui peut justement permettre de réduire la complexité de l'appariement. Si les points sont extraits d'images en niveaux de gris, ces niveaux peuvent être utilisés pour rendre les points plus informatifs et, là aussi, réduire la complexité.

Utilisation de la distance de Hausdorf : mention doit être faite des travaux de HUTTENLOCHER [HR93, HKR93, HJ95, HLO96]. Ce dernier utilise les contours extraits des images et cherche à superposer au mieux les contours des deux images sans chercher d'appariements point à point entre les points de contours. Pour cela, il utilise une mesure globale de recouvrement entre les contours, la distance de HAUSDORF. Pour trouver la meilleure superposition, il applique à l'une des images une transformation et réalise une minimisation sur les coefficients de cette transformation. Ne cherchant pas d'appariements point à point, cette méthode évite de nombreuses erreurs et se montre robuste. Par contre, elle ne fournit pas les appariements entre points, indispensables à tant d'algorithmes. Cela limite grandement son domaine d'application.

Un dernier cas, qui a été très étudié, est celui de l'appariement stéréoscopique. On désigne par ce terme le problème de l'appariement lorsqu'on connaît la géométrie épipolaire entre les images. Cela exige, bien entendu, une étape préalable d'étalonnage. Mais cet étalonnage est incomplet et est assez facile à réaliser. Par exemple, si les deux caméras utilisées sont rigidement fixées l'une par rapport à l'autre, on peut, en observant une mire, calculer la matrice fondamentale à partir d'appariements faits à la

main ou à partir d'un modèle tridimensionnel de la mire utilisée. La matrice calculée ne varie pas lorsque le système de caméras est déplacé. Une limite de cette approche est qu'elle nécessite un premier appariement entre deux images qu'il peut être fastidieux de réaliser à la main.

Les méthodes utilisées dans ce cas sont le plus souvent basées sur des corrélations locales, la connaissance de la géométrie épipolaire fournissant de nombreuses contraintes supplémentaires. Aux contraintes géométriques exactes (un point doit se situer sur la droite épipolaire définie par son correspondant), il est ajouté une série de contraintes heuristiques : la plupart des objets ne sont pas transparents, ils sont vus sous des angles proches par les deux caméras...

1.1.2 Modélisation et catégorisation

Pour re-connaître un objet, il faut le connaître auparavant. La question est de savoir de quel type de connaissance on va avoir besoin pour la reconnaissance, et comment on peut stocker et utiliser cette information. Dans l'état actuel de la vision, il n'est pas envisageable de partir d'une connaissance autre que graphique des objets. Le passage entre description textuelle et description visuelle est encore bien trop grand pour qu'on puisse le franchir pour des objets généraux à l'heure actuelle. Mais, même dans la catégorie des descriptions visuelles, il faut se restreindre.

De nombreux travaux ont été faits pour reconnaître des objets dont on connaissait le modèle CAO [Goa86, Bha87, Ike87, SLH87, SB91a, ZSB91, FJ91]. Dès que les objets ne sont pas très simples, ces méthodes sont mises en échec. La cause en est multiple [CM91, Bow91]. Tout d'abord, l'information codée dans ces modèles est principalement d'ordre géométrique et non visuelle. En particulier, les caractéristiques les plus discriminantes visuellement ne sont pas forcément les plus importantes géométriquement. Ensuite, ces modèles ne savent pas gérer la notion de détail : toute l'information est traitée au même niveau alors que visuellement, certains détails n'apparaissent qu'à certaines échelles. D'autre part, l'effet global de nombreux détails invisibles individuellement est difficilement prévisible.

La démarche inverse, construire des modèles CAO à partir d'images, n'est guère plus facile à cause de cela.

En réaction, la constitution de modèles autres que CAO repose sur une apprentissage réalisé à partir de données réelles. Dans ce cadre très général, le terme apprentissage signifie principalement que l'on utilise des données redondantes pour ne retenir que la partie invariable de ces données, ou modéliser cette variabilité.

Les modèles construits peuvent être 3D, mais on se rapproche alors des techniques de reconstruction de modèles CAO, ou 2D, une des manières courantes de coder la variabilité étant l'utilisation d'images propres [TP91, MN93].

La catégorisation consiste à regrouper des images semblables dans un sens plus général. Il ne s'agit plus en effet de construire un modèle précis d'un objet ou d'un aspect d'un objet, mais de retrouver dans un ensemble quelles sont les images qui représentent une même classe d'objets ou de scènes. Il y a alors deux problèmes successifs à résoudre.

Il faut tout d'abord choisir une série de descripteurs des images et définir une distance entre images basée sur ces descripteurs. Les histogrammes de couleurs avec une distance issue du χ^2 est une solution courante [SB91b, SS94], qui peut être améliorée pour être plus invariante aux changements d'illumination [FF95, Nag95, NB93, SH96]. Mais les possibilités sont innombrables : LAN propose de calculer en de nombreux points un vecteur de plus de 8 000 composantes [LM97].

Le second problème est celui du regroupement effectif des images en fonction de leur distance mutuelle. On retrouve ici le traditionnel problème du regroupement ou

clustering en anglais pour lequel l'analyse de données fournit de nombreuses techniques [LMP95, Rip96]. Quelques unes de ces techniques sont présentées plus en détail au troisième chapitre.

1.1.3 Indexation d' images par le contenu

Si l'appariement des images s'intéresse prioritairement au cas où l'on dispose de deux images à comparer, l'indexation élargit ce cadre au cas où l'on dispose d'un grand nombre d'images ou de modèles qui forment ce que l'on appellera une *base d'images*. On suppose que l'on dispose, en outre, d'une autre image, extérieure à la base, que l'on souhaite confronter à la base. Par convention, nous l'appellerons l'*image inconnue*. Cette dénomination se comprend si l'on identifie la base avec ce que l'ordinateur «connaît», et cette nouvelle image avec quelque chose qui ne sera «connu» que lorsqu'il aura été rapproché ou mis en lien avec un élément de la base.

Cette situation générale peut se retrouver sous des formes proches dans divers problèmes pratiques.

- On peut tout d'abord chercher si l'image inconnue n'est pas une des images de la base, ou une partie ou un sur–ensemble d'une des images de la base. Cela peut servir à une agence de photographie à retrouver parmi les images publiées dans la presse celles qui viennent de sa propre base et vérifier ainsi que les droits afférents ont été payés. Ou cela peut servir à une personne voulant publier une image à retrouver son origine en la confrontant à diverses bases pour se mettre en contact avec le possesseur des droits. Autre cas concret, cela aurait pu servir pour rechercher si, par hasard, la photographie qui a été publiée comme étant celle de la voiture endommagée de la princesse de Galles ne sortait pas en fait d'une base d'images. On parlera alors de recherche de l'image inconnue dans la base.

- Au lieu de faire une recherche exacte, on peut chercher l'image ou les images les plus proches de l'image inconnue. Même si la scène représentée sur l'image inconnue est aussi représentée sur des images de la base, il est fréquent que les angles de vues ou les conditions de la prise de vue aient variés. Il convient alors de savoir s'affranchir de ces transformations, de la même manière que cela devait être fait lors d'un appariement. Dans ce cas, on parlera de reconnaissance de l'image inconnue.

- On peut enfin chercher à classer l'image inconnue dans la base, en fonction de groupes préexistants. Il s'agit alors d'un problème de catégorisation.

Ces problèmes de recherche, reconnaissance ou catégorisation se retrouvent dans de nombreux domaines d'application : données biomédicales, agences de photographie, images satellitaires et photo-interprétation, gestion de catalogues de photographies de produits dans des entreprises, gestion d'archives photographiques ou cinématographiques, structuration de vidéos...

D'un point de vue technique, ces problèmes peuvent être séparés en deux : il faut d'une part disposer d'une technique de comparaison d'images, et d'autre part d'un système de gestion de la base.

Pour le premier point, il faut, bien entendu, que la technique puisse s'adapter au cas d'un grand nombre d'images. Cela exclut la plupart des algorithmes de prédiction – vérification qui ne pourrait fonctionner que de manière linéaire, c'est à dire en essayant toutes les images de la base les unes après les autres [BJ85, CSF86]. Dès que la base est grande, le temps de réponse devient rédhibitoire, sauf si des mesures spécifiques ont été prises pour faire face à ce problème [AF86, BH86].

Le deuxième point a pour but de rendre cette comparaison entre une image et un ensemble d'images possible dans la pratique. L'idée de base est qu'il faut qu'une requête puisse être traitée en un temps acceptable. Pour atteindre un tel objectif, on va précalculer un maximum d'éléments relatifs aux images de la base, éléments choisis en fonction de la méthode de comparaison choisie, puis indexer ces informations dans la base, pour que la phase de confrontation de l'image inconnue avec la base soit la plus brève possible. La phase d'indexation est donc cruciale pour la rapidité de la réponse à une requête.

De manière pratique, il faut donc traduire la méthode de comparaison en terme de comparaison d'indices visuels, indices que l'on va indexer dans une base de données [Rot95].

Point de vue indices, les premiers systèmes réalisés et destinés à des bases de plus de 100 images utilisent principalement des descripteurs globaux des images : couleur et texture. Ces mêmes descripteurs peuvent être calculés de manière plus locale, en utilisant un prédécoupage a priori des images en quelques régions, ce qui permet d'introduire des notions de composition spatiales entre les couleur ou les textures. Quelques systèmes proposent des indices de plus haut niveau, tels des formes, mais on ne voit pas bien comment ces formes ont pu être extraites autrement que manuellement.

Divers systèmes reposant sur ces principes, tels QBIC d'IBM ou Virage, ainsi que divers autres projets en cours de développement dans des universités sont recensés sur le site web `http://206.169.1.82/vim/`. Divers articles sur ces systèmes sont aussi disponibles [NBE+93, FBF+94].

Une fois les index choisis, reste à les stocker effectivement en mémoire et à pouvoir les retrouver de manière efficace. Cette question du stockage n'est certainement pas neuve en informatique. Les bases de données sont un des sujets de recherche les plus anciens dans la communauté informatique. Nous ne ferons pas d'état de l'art, mais nous nous contenterons d'observer que le domaine d'excellence de ces bases est celui des données exactes.

La question du stockage des index visuels présente, par contre, des aspects nouveaux, en ce sens que ses contraintes particulières en font une question loin d'être résolue. Pourquoi ? Les index que l'on souhaite stocker sont souvent des vecteurs de grande dimension que l'on doit comparer en tolérant une certaine variabilité. Cela revient donc à chercher dans une base tous les vecteurs dont la distance à un vecteur donné est inférieure à un seuil donné.

Le caractère variable des données rend inutilisable la plupart des techniques d'indexation des systèmes de gestion de base de données (SGBD) disponibles. Quelques algorithmes sont disponibles pour gérer les informations spatiales dans les systèmes d'information géographiques (SIG), mais leur complexité est le plus souvent fortement exponentielle avec la dimension des données. L'utilisation de ces algorithmes pour des dimensions supérieures à trois ou quatre est donc vouée à l'échec le plus souvent.

1.2 Notre contribution

1.2.1 Point de départ

Notre travail de recherche en vision a commencé en 1990. À cette date, extraire les contours d'une image 256×256 prenait 30 secondes de calcul sur une station de travail. Toute autre utilisation directe du signal en niveau de gris était donc souvent considérée comme trop coûteuse. L'alternative la plus employée consistait à utiliser les contours qui offrent une très forte diminution du volume des données, tout en gardant l'information nécessaire pour étudier tant la géométrie de la scène représentée que son contenu et en assurant une invariance aux changements d'illumination. Une telle

hypothèse s'est souvent révélée réaliste dans le cas de scène comportant principalement des objets polyédriques, même si elle n'a jamais été la panacée.

Toujours en 1990, la cadre privilégié d'application des techniques de vision, dans l'équipe MOVI du moins, était la robotique et l'aide à la planification et au contrôle des robots. Le scénario classique consistait à aider un bras muni d'une pince à saisir un objet en détectant les obstacles à éviter, en reconnaissant l'objet à saisir et en participant au contrôle de la saisie.

Nos travaux ont eu pour but, quasiment depuis le début, de réaliser un système de reconnaissance d'objets. Au début, le cadre d'hypothèses dans lequel on se plaçait était le suivant :

1° les scènes contiennent principalement des objets polyédriques,

2° les primitives de base à utiliser sont les segments approchant les contours,

3° les objets peuvent s'occulter partiellement les uns les autres.

Ces hypothèses se sont ensuite élargies au cas des images texturées, pour lesquelles l'emploi des segments n'est plus pertinent.

1.2.2 Principes

Pour résoudre ce problème, plusieurs principes ont été retenus. Tout d'abord, le fait de construire les modèles devant servir à la reconnaissance à partir de données perçues, et non de données a priori. Ceci permet de se soustraire aux difficultés que rencontrent les techniques basées sur l'utilisation de modèles CAO.

Ensuite, nous avons choisi de ne garder que des modèles 2D et ainsi de séparer complètement la reconnaissance de l'utilisation de cette reconnaissance, que ce soit pour la saisie, la reconstruction ou le calcul de pose. L'idée sous-jacente est que la reconnaissance est d'autant plus facile et d'autant plus rapide que les termes à comparer sont de types les plus semblables. Cela devait donc permettre d'apparier ou reconnaître des objets dont on ne connaît pas la description 3D, et de ne se baser que sur la façon dont il apparaissent dans les images.

Troisième principe, le fait d'utiliser, à la base du système, un algorithme géométrique d'appariement entre images. Comme déjà dit plus haut, l'appariement est un des traitements de base qui permet de comparer le contenu même des images, et il peut donc servir de brique de base à tout système voulant traiter des images par leur contenu.

Cet algorithme, tout d'abord purement géométrique, a été complété, indépendamment de nos travaux par SCHMID, pour prendre en compte les aspects photométriques dans les images en niveaux de gris, et nous avons réalisé la généralisation à la couleur.

Dernier principe, le fait de pouvoir transformer l'algorithme d'appariement en algorithme d'indexation n'obligeant pas à considérer tous les modèles ou images de la base séquentiellement.

1.2.3 Étapes de réalisation

L'algorithme d'appariement

La première étape de notre travail (qui a correspondu à ma thèse) a été le développement d'un algorithme d'appariement basé sur la recherche d'une approximation euclidienne du mouvement apparent entre les deux images à apparier [GHM91, Gro91, Gro92, Gro93b]. Cet algorithme nous a amené à utiliser des invariants euclidiens plans très simples, des angles et des rapports de distance, dont la robustesse et les possibilités ont surpris.

Une première extension a été réalisée avec Edmond BOYER pendant son stage de DEA. Nous avons adapté l'algorithme pour pouvoir utiliser des invariants affines ou projectifs, et à ajouter une phase de vérification des appariements par estimation robuste de la géométrie épipolaire et approximation robuste du mouvement apparent par une homographie. Divers algorithmes de complètement des appariements ont aussi été développés [Gro93a].

L'utilisation de ces invariants plus riche a correspondu dans le temps aux débuts de l'étude de la géométrie projective pour la vision, mais les invariants affines n'ont fourni que des résultats un peu moins bons que ceux obtenus avec les invariants euclidiens, et ceux donnés par les invariants projectifs furent bien moins bons.

L'explication fut fournie peu après par différents travaux, dont ceux de BINFORD [BL93], de BEN-ARIE [BA90b, BA90a], de ÅSTRÖM [ÅM92, Åst93] et de MORIN [Mor93]. Ces travaux montraient que l'invariance n'est pas forcément la notion utile en pratique. Cette invariance est beaucoup trop stricte et exige de prendre en considération des situations trop proches de cas singuliers. Pour y faire face, les invariants ont des expressions plus complexes, qui les rendent instables et très sensibles au bruit présent dans les images. D'autre part, la comparaison de tels invariants est délicate.

À l'inverse, les invariants euclidiens plans que nous utilisons se sont révélés être des quasi-invariants selon la définition de BINFORD, sujets à l'effet de pic de BEN-ARIE. Les invariants affines sont eux aussi des quasi-invariants si les configurations 3D correspondantes sont planes, ce qui ne peut pas être vérifié a priori.

Modélisation des aspects d'un objet

Dès qu'un premier algorithme d'appariement a été disponible, nous avons commencé à travailler sur la modélisation des aspects d'objets pour la reconnaissance. Il s'agit de calculer un modèle des principaux aspects visuels d'un objet à partir d'un ensemble d'images de cet objet, ces modèles devant permettre de remplacer les modèles CAO couramment employés.

Avec trois stagiaires, Gudrun SOCHER, Marie LEGENDRE et Marie-Hélène MALISSEN, nous avons travaillé en 1992 sur la mise au point d'une distance entre images et sur l'utilisation de cette distance pour regrouper des images. L'année suivante, avec l'aide d'Olivier BOURNEZ, l'adaptation de l'algorithme d'appariement au cas de n images a été mise au point, ce qui a permis de calculer les premiers modèles [Gro95]. Ces deux étapes mises bout à bout fournissaient une première version de l'algorithme de modélisation, qui a été employée dans le cadre du projet européen SECOND. Il s'agissait d'apparier une image connue et une nouvelle image d'un objet pour faire le lien entre cette image et un modèle CAO 3D de l'objet, calculer ainsi sa pose et pouvoir le saisir à l'aide de la pince d'un robot.

La limitation principale de l'expérimentation menée est qu'il fallait savoir à l'avance avec qu'elle image devait être appariée avec la nouvelle image. Il aurait été bien plus pratique de pouvoir laisser le système choisir parmi un ensemble qu'elle était l'image la plus adéquate. Ceci est d'autant plus nécessaire dans le cadre de la saisie, que l'objet à saisir peut bouger ou être bousculé et présenter à la caméra un aspect différent de celui attendu.

C'est par ce biais que nous avons commencé à nous intéresser au problème de la confrontation d'une image inconnue à une ensemble d'images, que l'on peut appeler une base d'images. Il s'agissait donc bien de faire de la reconnaissance d'objet, et non de la recherche d'images globalement semblables, recherche qui n'a guère d'intérêt dans le cadre de la saisie en robotique.

Mon séjour à CMU a été l'occasion d'un travail plus particulièrement axé sur la catégorisation des images en environnement urbain [TGHI97]. Il s'agissait au départ

de savoir reconnaître visuellement le type d'environnement dans lequel évolue un véhicule automatique, pour pouvoir adapter le style de conduite à cet environnement. Nous avons donc proposé une méthode de catégorisation qui permette d'apprendre des classes d'environnements. À cette occasion, nous avons développé un test d'arrêt pour les méthodes de regroupement, test qui a aussi été utilisé pour la modélisation des aspects d'un objet.

Indexation et reconnaissance d'images

Le travail sur l'indexation a tout d'abord été abordé lors du stage de DEA d'Olivier MICHEL en 1993, puis, le sujet étant mieux délimité, de la thèse de Bart LAMIROY de 1994 à 1998 [Lam98]. La première partie de cette thèse a consisté à factoriser l'appariement en mettant au point une technique d'indexation des images de la base, ce qui a débouché sur un système complet et fonctionnel.

L'intérêt pour le sujet de l'indexation et de la recherche d'images dans de grandes bases de données à l'échelle mondiale date de cette époque. Parallèlement à nos travaux, Cordelia SCHMID a fait sa thèse sur les invariants pour images texturées, apportant des possibilités nouvelles, tout en restant dans le cadre de la reconnaissance d'objet. Beaucoup d'autres équipes universitaires, mais aussi industrielles ont commencé à développer des systèmes de recherche d'images.

Cet enthousiasme collectif a aussi correspondu aux premières années où Internet a commencé à être massivement utilisé par des non – informaticiens et par des industriels. Ces personnes ont exprimées de nouveaux besoins, en particulier pour la gestion de leurs données images ou multimédias qu'ils commençaient à posséder en quantités importantes. Le basculement au numérique de nombreuses techniques s'est fait sentir : cédéroms et disques compacts, vidéodisques, images satellitaires, sites web, télévision numérique, appareils photos numériques...

Or, toute cette information ne peut être vraiment utile que si on peut y accéder facilement, et certains se sont plus à souligner la différence entre les autoroutes de l'information annoncés, et le labyrinthe que l'on devait affronter dans la pratique, par exemple sur le web.

Cette mini-révolution a amené plusieurs changements majeurs dans la façon dont se pose le problème de la reconnaissance. Tout d'abord, on trouve des besoins de recherche généraux, pour lesquels des techniques simples, telles celles mises au point sur de nombreux systèmes à base d'histogrammes globaux, sont suffisantes. Mais le fait de pouvoir faire de la reconnaissance exacte a aussi été fortement exprimé.

D'autre part, la quantité des images et données en jeu impose l'emploi de base de données pour les stocker. Malheureusement, les techniques actuelles d'indexation dans les SGBD actuels sont mal adaptées aux descripteurs d images que nous utilisons.

Un troisième facteur important est la variété des applications envisagées, et par suite, celle des types d'images employées : images aériennes, images de télévision, photographies, photo-montages... On y trouve toutes les résolutions, toutes les qualités, tous les sujets.

Enfin, l'utilisateur non – informaticien a besoin d'interfaces conviviales permettant d'exprimer simplement des requêtes complexes, offrant la possibilités de reformuler ou d'affiner itérativement ses requêtes, tout en obtenant des temps de réponse courts, ce qui pose la question de la complexité des techniques utilisées.

Tout cela fournit un ensemble de problèmes auquel on ne peut échapper si l'on veut rester crédible face à nos partenaires industriels. Nous avons donc commencé à les explorer.

Pour pouvoir traiter des images de types plus divers, nous avons mis au points de nouveaux descripteurs d'images. L'extension au cas des images en couleur les invariants

de texture de SCHMID a été faite lors du stage d'été de Rémi DELON [GMD+97]. Bart LAMIROY a proposé diverses façons de faire collaborer les méthodes d'appariement pour traiter de nouvelles images difficiles. Enfin, le travail de thèse de Sylvaine PICARD [MGL+97] est l'occasion de tester une classe d'invariants robustes assez différents, basés sur les codes ordinaux de ZABIH [ZW94].

Nous avons aussi commencé à travailler sur l'utilisation de SGBD pour stocker nos descripteurs. Cela implique la définition d'un schéma de base de données et la mise au point d'une technique d'indexation rapide. L'imprécision des descripteurs utilisés en fait un problème difficile pour lequel les SGBD actuels ne fournissent pas de réponse standard. Le stage de fin d'études de Yann HERVOUET a eu pour objet une première exploration de ce sujet. ce premier travail va être repris cette année dans le cadre d'une collaboration avec une équipe de recherche en base de données.

En complément, Bart LAMIROY [LG97] a démontré divers résultats de complexité sur l'indexation de descripteurs imprécis pour les techniques procédant par découpage de l'espace. Ces résultats soulignent en particulier la difficulté de la manipulation de descripteurs de grande dimension. Un des défis non résolus est de savoir si des méthodes par découpage des données offrent une meilleure complexité.

Deux applications ont été plus particulièrement développées pour démontrer l'intérêt de nos techniques. D'une part un système d'indexation d'images fixes, dont l'interface a été développée en collaboration avec Nagib AOUINI et Jean-Yves LAHITTE lors de leur stage de fin d'études, et d'autre part, la structuration des vidéos fait l'objet d'un important partenariat avec Alcatel auquel participent plusieurs membres de l'équipe : Roger MOHR, Patrick GROS, Cordelia SCHMID, Riad HAMMOUD, Pascal BERTOLINO et plusieurs stagiaires [GMGB97]. Ces applications, ainsi que les perspectives ouvertes par notre travail sont détaillés dans le dernier chapitre.

Chapitre 2

Appariement d'images

Le moins qu'on puisse demander à une sculpture,
c'est qu'elle ne bouge pas.
Salvadore Dali.

Ce chapitre et les deux suivants ont pour but d'exposer les principaux algorithmes développés et résultats obtenus en matière d'appariement, de modélisation et d'indexation. Ils ont donc une visée plus technique que le chapitre précédent qui replace ces travaux dans un cadre plus vaste, ou que le dernier chapitre qui a pour but d'exposer applications et perspectives de notre travail.

Le présent chapitre est divisé en trois sections. La première décrit l'algorithme développé dans la cas des images structurées, fournit divers résultats expérimentaux et présente une extension de l'algorithme à l'appariement de plus de deux images. La deuxième section présente brièvement les travaux de Schmid sur l'appariement des images texturées en niveaux de gris et la troisième nos travaux sur l'extension de ces résultats aux images colorées.

2.1 Appariement d'images structurées

Cette section comporte quatre parties : la première décrit l'algorithme de base, la deuxième étudie la complexité de l'algorithme et fournit de premiers résultats, la troisième indique comment améliorer ces premiers résultats, et la dernière montre comment passer d'appariements de deux images à un appariement global entre plusieurs images.

2.1.1 Algorithme d'appariement de deux images

Dans toute cette partie, on se place dans la cas où l'on veut apparier deux images contenant des segments de droites reliés entre eux par leurs extrémités. Ces images sont le plus généralement obtenues à partir d'images en niveaux de gris par extraction de contours [Can86, Der87], puis par approximation polygonale de ces contours. Les segments obtenus sont alors reliés par leurs extrémités et organisés en un graphe d'adjacence [HSV90]. La Fig. 2.1, page 27, montre de telles images. Le problème consiste à trouver les correspondances entre les segments des deux images et entre leurs points extrémités.

Si on munit chaque image d'un repère dont l'origine est par exemple le coin inférieur gauche de l'image, on peut immédiatement s'apercevoir que les primitives en

correspondance n'ont pas les mêmes coordonnées dans les deux images. Cette différence est appelée mouvement apparent (même s'il n'y a aucune relation temporelle entre les images).

Ce mouvement apparent n'est pas une transformation géométrique classique. En effet, deux points de la scène peuvent donner des projections confondues dans une des deux images et des projections séparées dans l'autre. Aucune application ou transformation géométrique classique ne peut donc, dans le cas général, décrire ce mouvement de manière exacte et il ne peut donc y avoir d'invariant pour ce mouvement.

Par contre, on peut trouver des quasi-invariants, tels les angles et les rapports de longueurs, qui s'avèrent être des invariants pour la transformation plane, euclidienne ou affine qui approche le mouvement apparent (d'où une appellation d'invariants ou de quasi-invariants suivant la façon de considérer ces quantités). Ce ne sont, par contre, pas des invariants pour la projection perspective.

L'algorithme que nous proposons est basé sur le calcul d'une approximation du mouvement apparent par une transformation géométrique plane simple, soit une similitude, soit une transformation affine (nommée affinité dans toute la suite).

Cette méthode présente plusieurs caractéristiques intéressantes pour traiter des images réelles :

1° elle ne suppose pas connus les paramètres des caméras utilisées ou la géométrie épipolaire, ce qui aurait, dans le cas contraire, permis d'utiliser la méthode par prédiction et vérification ; le mouvement apparent entre les images n'est pas supposé infiniment petit, ce qui aurait permis l'emploi des techniques de poursuite d'indices (*tracking* en anglais) ou de corrélation ;

2° les objets de la scène peuvent avoir des mouvements différents ;

3° l'utilisation de quantités numériques géométriques associées à des configurations de primitives permet d'avoir une complexité très faible par rapport à celle des algorithmes de recherche d'isomorphismes ;

4° les invariants locaux utilisés offrent une grande robustesse au bruit affectant les images.

2.1.1.1 L'algorithme

Les étapes de cet algorithme sont les suivantes.

Choix du type d'approximation du mouvement apparent. On commence par choisir un des deux types d'approximation du mouvement apparent : similitude ou affinité. Remarquons que, dans le cas des affinités, on sort du cadre des quasi-invariants : les invariants affines du plan calculés à partir de quatre points ne sont, en effet, des quasi-invariants que si les quatre points correspondant dans la scène sont coplanaires. Tant que la géométrie épipolaire n'est pas connue ou qu'une information de coplanarité n'est pas disponible, il n'est pas possible de vérifier la coplanarité des points. Une moins grande robustesse au changement de point de vue est donc prévisible. En contre partie, cette approximation affine offre plus de souplesse de déformation, c'est à dire de degrés de liberté.

Étape 1 : calcul des invariants. On calcule, dans chacune des deux images, les invariants locaux de certaines configurations de primitives pour le type de transformation choisi. Pour limiter la complexité de l'énumération de ces configurations, on se limite aux configurations connexes de deux segments en V et à celles de 3 segments en Z et Y. Dans le cas des similitudes, on prend alors comme invariants les angles formés entre les segments et les rapports entre les longueurs de ces derniers. Dans le cas

des affinités, on prend, pour le configurations en Y, les coordonnées barycentriques du point central par rapport aux trois extrémités dans branches. Pour les configurations en Z, on construit le point d'intersection du segment central et de la droite reliant les deux points extrêmes de la configuration, puis on retient comme invariant les rapports définis par ce point sur les deux segments dont il est l'intersection.

Étape 2 : appariement des invariants. On apparie ces invariants entre les deux images en fonction de leur valeur et de seuils qui servent à prendre en compte le bruit présent dans les images et sa répercussion sur les valeurs calculées. On préfère des seuils à des techniques floues pour limiter le nombre de paires d'invariants considérés dans la suite. À cette étape, il n'est pas rare qu'une configuration soit appariée à une ou deux dizaines d'autres.

D'un point de vue pratique, cette comparaison peut être faite avec une complexité $\Theta(n \log n)$ et non pas quadratique.

Étape 3 : calcul des transformations. De tous les appariements réalisés, seule une petite partie est correcte. Pour la séparer des mauvais appariements, on se sert du fait que les appariements corrects doivent définir le même mouvement apparent et donc des approximations proches de ce mouvement.

Chaque fois que deux configurations de primitives ont leurs invariants appariés, on calcule donc la transformation (similitude ou affinité selon le cas) qui amène la première configuration sur la deuxième. ce calcul est effectué aux moindres carrés.

Étapes 4 et 5 : filtrage des similitudes. Ces transformations sont représentées dans leur espace de paramètres $\mathbb{R}^d$, avec $d = 4$ pour les similitudes (2 pour la translation, 1 pour la rotation et 1 pour le facteur d'échelle) et $d = 6$ pour les affinités (4 pour la partie vectorielle et 2 pour la translation).

Les appariements incorrects ne sont pas corrélés entre eux : les similitudes correspondantes sont donc représentées par des points qui sont répartis dans une grande partie de l'espace $\mathbb{R}^4$. À l'opposé, les appariements corrects définissent tous de bonnes approximations du mouvement apparent, et donnent donc des similitudes qui sont regroupées dans une petite région de l'espace. C'est ce point d'accumulation qu'on va donc chercher.

Pour cela, on définit une distance entre similitudes :

$$\mathrm{d}\left((t_x,t_y,\theta,k),(t'_x,t'_y,\theta',k')\right) = \frac{(t'_x-t_x)^2}{\lambda_x^2} + \frac{(t'_y-t_y)^2}{\lambda_y^2} + 2\left(k^2+k'^2-2kk'\cos(\theta'-\theta)\right)$$

où λ_x et λ_y sont les tailles des images selon x et y respectivement, et entre affinités :

$$\mathrm{d}\left(\begin{pmatrix} a_1 & a_2 & t_x \\ a_3 & a_4 & t_y \\ 0 & 0 & 1 \end{pmatrix}, \begin{pmatrix} a'_1 & a'_2 & t'_x \\ a'_3 & a'_4 & t'_y \\ 0 & 0 & 1 \end{pmatrix}\right) = \sum_{i=1}^{4}(a_i-a'_i)^2 + \frac{(tx-t'x)^2}{\lambda_x^2} + \frac{(t_y-t'_y)^2}{\lambda_y^2}$$

On calcule alors pour chaque transformation, la somme des inverses des distances de cette transformation à ces transformations voisines. La transformation obtenant le score le plus élevé est retenue comme point d'accumulation maximal.

On ne garde alors que les appariements correspondant à des transformations qui appartiennent à cette accumulation maximale.

Étape 6 : appariement des primitives. Il suffit alors de passer des appariements d'invariants et donc de configurations de primitives à ceux des primitives individuelles.

En cas d'ambigüité, on ne garde que les appariements les plus probables : si une primitive a est appariée cinq fois à une primitive b et une seule fois à une primitive c, on ne garde que l'appariement (a, b).

2.1.1.2 Comparaison avec le hachage géométrique et l'effet de pic.

Le hachage géométrique est aussi basé sur l'utilisation d'invariants locaux : les points extraits des deux images sont organisés en configurations auxquelles sont associés des invariants. De là, la ressemblance avec la méthode exposée ici.

La principale différence vient de l'utilisation de l'information topologique. Le hachage géométrique utilise en effet tous les n-uplets de points possibles : cela conduit d'une part à une très forte combinatoire, et fait reposer les résultats sur la validité globale dans l'image de l'approximation du mouvement apparent.

Avec l'effet de pic, la principale différence est l'absence, dans la méthode présentée ici, de tout modèle ou connaissance 3D.

2.1.2 Évaluation et résultats

2.1.2.1 Complexité

L'opération la plus consommatrice de temps dans l'algorithme est la comparaison de n-uplets à un seuil ε près. On prend cette opération de comparaison entre deux vecteurs comme opération élémentaire pour l'évaluation de la complexité. Si m est le nombre de n-uplets, la complexité de l'opération est $O(m \log m)$ quand $\varepsilon = 0$ et $O(m^2)$ quand $\varepsilon \to \infty$. Notons $O(f_\varepsilon(m))$ cette complexité dans le cas général.

Pour évaluer la complexité de l'algorithme, on peut faire l'hypothèse qu'il y a approximativement le même nombre de segments dans les deux images. Notons s ce nombre. Comme le cas de jonctions entre 4 segments est plutôt rare, on peut considérer que la complexité en temps du calcul des invariants est $O(n)$. La comparaison des invariants des deux images a donc pour complexité $O(f_{\varepsilon_1}(s))$.

À l'issue des comparaisons d'invariants, m transformations sont calculées. Au maximum, m vaut $O(f_{\varepsilon_1}(s))$. La comparaison de ces transformations peut être réalisée en $O(f_{\varepsilon_2}(m))$. Les dernières étapes de l'algorithme ont des complexités plus faibles.

Au total, la complexité globale de l'algorithme est donc $O(f_{\varepsilon_2}(f_{\varepsilon_1}(s)))$, avec :

$$O(n(\log s)^2) \leqslant O\left(f_{\varepsilon_2}(f_{\varepsilon_1}(s))\right) \leqslant O(s^4)$$

Par comparaison, le hachage géométrique nécessite la comparaison de tous les quadruplets de points : sa complexité est donc $O(n^4)$. Il faut toutefois remarquer que le hachage géométrique est plus un algorithme d'indexation que d'appariement. En matière d'indexation un appariement complet n'est pas toujours nécessaire, ce qui permet de réduire la complexité.

Le tableau 2.1 présente quelques résultats d'évaluation expérimentale de la complexité. Pour chacune des expériences d'appariement, les colonnes fournissent les nombres de segments dans les images (1), extrémités dans les images (2), invariants dans chaque image (3), comparaisons faites entre invariants (4), transformations calculées (5), distances calculées entre transformations (6), transformations retenues (7), segments appariés (8) et extrémités appariées (9). La colonne T indique le type d'approximation du mouvement apparent : S pour similitude, A pour affinité. Il apparaît clairement que la complexité reste de loin très inférieure à $O(s^4)$!

TAB. 2.1: *Évaluation expérimentale de la complexité de l'algorithme d'appariement.*

	T	1	2	3	4	5	6	7	8	9
Image 1	S	33	29	53						
Image 2	S	34	27	62	402	91	403	12	18	17
Image 3	S	136	142	173						
Image 4	S	102	106	131	2648	487	1223	13	21	25
Image 5	S	306	266	485						
Image 6	S	312	276	491	26241	3960	17988	33	62	78
Image 1	A	33	29	101						
Image 2	A	34	27	128	5119	1292	9894	30	21	20
Image 3	A	136	142	264						
Image 4	A	102	106	199	16677	2998	8790	26	22	22
Image 5	A	306	266	906						
Image 6	A	312	276	920	31276	5625	20499	5	15	20

2.1.2.2 Résultats expérimentaux

Ce paragraphe a pour but d'illustrer l'algorithme décrit précédemment par des résultats obtenus avec des images réelles, et, par là, de valider les hypothèses et approximations faites.

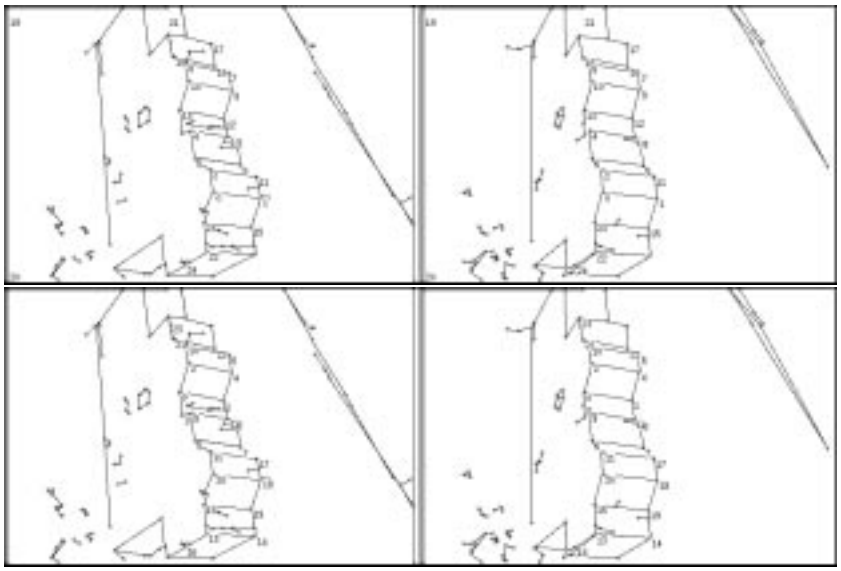

FIG. 2.1: *Premiers exemples d'appariements, en haut avec les similitudes, en bas avec les affinités.*

La FIG. 2.1 montre les résultats de l'appariement de deux images. Pour les deux images du haut de la figure, on a utilisé l'approximation par les similitudes, et les extrémités de segments appariées portent les mêmes numéros. Pour les deux images du bas (qui sont les mêmes qu'en haut), on a utilisé l'approximation par les affinités. Il n'y a pas de correspondances entre les numéros des images du haut et ceux des images du bas.

La première image contient 133 points, la deuxième 106. Sur ce nombre, on peut, à vue, en compter une quarantaine qui sont caractéristiques de l'objet, les autres étant dus au bruit, au fond ou à la texture (le décompte exact de ces points est

délicat). Avec les similitudes, on a obtenu 25 appariements et 19 avec les affinités. Tous ces appariements sont corrects, bien qu'avec les similitudes, on ait aussi apparié deux points du bord de l'image. Par contre, certains points de l'objet ne sont pas appariés. Ces lacunes sont liées à des segments fortement bruités (par exemple, coupés en deux par une jonction dans une des images) ou à des zones où l'approximation du mouvement apparent par une similitude ou même une affinité est trop grossière (non conservation des angles due à la projection perspective par exemple).

Ces résultats montrent toutefois clairement que les quasi-invariants utilisés sont robustes, alors même que le mouvement apparent global est visiblement assez loin d'une similitude ou même d'une affinité.

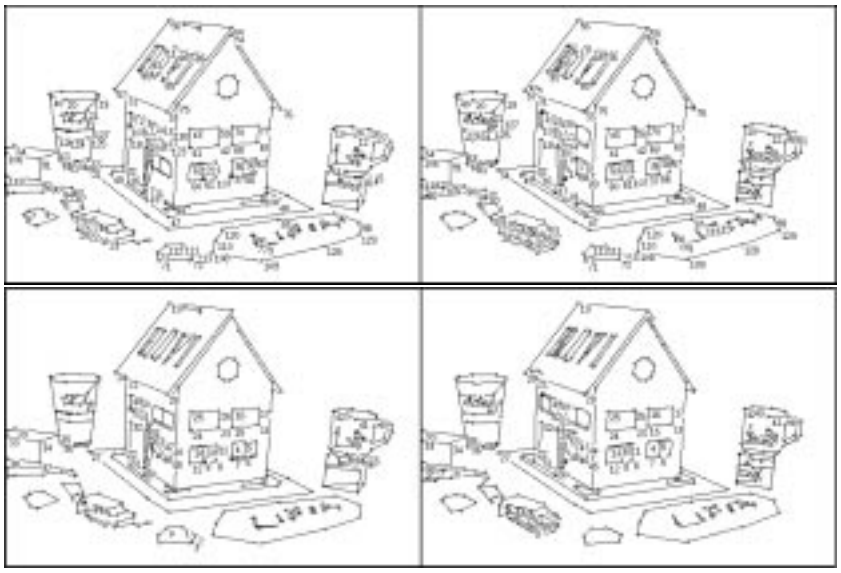

FIG. 2.2: *Autres exemples d'appariements, en haut avec les similitudes, en bas avec les affinités.*

La FIG. 2.2 montre d'autres résultats sur le même principe, avec deux autres images. Ces images contiennent des objets non polyédriques pour lesquels l'approximation des contours par des segments est peu robuste et fort bruitée. La première image contient 322 points, la deuxième 352. On peut estimer que 80 sont des caractéristiques stables de la maison. Avec les similitudes, on a apparié 131 points ; il y a une quinzaine d'erreurs grossières et une quinzaine d'erreurs liées aux fenêtres : à ces endroits, les rectangles forment des motifs répétitifs et les erreurs sont fréquentes. Avec les affinités, on trouve 58 appariements dont 8 erreurs grossières et 4 liées aux fenêtres.

2.1.3 Amélioration de l'appariement de deux images

L'algorithme décrit dans la partie précédente permet, sans aucune information préalable, de faire un premier appariement entre deux images. Les résultats obtenus peuvent contenir quelques appariements incorrects et beaucoup de lacunes : l'approximation du mouvement apparent par une similitude ou une affinité est, en effet, trop restrictive pour certaines parties des images. Mais il permet de passer d'un état où on ne disposait d'aucune information d'appariement à un autre où une telle information est disponible, même si elle n'est pas de la meilleure qualité : cela représente un saut très important.

Pour améliorer ce premier appariement, nous proposons la procédure suivante.

Approximation projective du mouvement apparent. Puisque l'utilisation d'une similitude ou d'une affinité pour approcher le mouvement apparent apparaît parfois trop limitative, une première idée est d'utiliser une transformation qui a plus de paramètres. Dans la hiérarchie de KLEIN [Kle74], la première classe qui englobe celle des affinités est formée des transformations projectives ou homographies. Lorsque la scène observée est plane, le mouvement apparent est d'ailleurs exactement une homographie.

Pour définir une homographie entre deux images, il est nécessaire de disposer d'au moins quatre appariements de points. L'algorithme de la section précédente en fournit généralement bien plus. L'homographie est calculée par une méthode robuste (moindre médiane des carrés), puis une fois les appariements aberrants ôtés, par une méthode non linéaire (LEVENBERG – MARQUARDT).

Calcul de la géométrie épipolaire. Ensuite, on calcule la géométrie épipolaire. Elle ne fournit pas une correspondance point à point, mais seulement point à droite. Elle a, par contre, l'intérêt d'être exacte pour des scènes non planes. Le calcul et l'utilisation de ces deux transformations complémentaires, font l'objet de cette section.

Comme précédemment, on utilise d'abord une méthode robuste de calcul, suivie par une estimation non linéaire. Pour la partie robuste, plusieurs méthodes linéaires sont disponibles. Il y a d'abord la méthode par moindres carrés simple, mais qui présente un mauvais conditionnement numérique conduisant souvent à de mauvais résultats. HARTLEY [Har95] a montré comment corriger ce conditionnement par un simple changement de coordonnées dans les images. De plus, pour obtenir une matrice de rang 2, il préconise de décomposer la matrice obtenue en valeurs singulières, d'annuler la plus petite de ces valeurs, puis de recomposer la matrice.

Une autre possibilité consiste à utiliser la méthode proposée par BOUFAMA [BM95], qui est basée sur une paramétrisation différente de la matrice fondamentale. L'avantage de sa méthode est, d'une part, de fournir de manière intrinsèque une matrice fondamentale de rang 2 et, d'autre part, de fournir des faisceaux d'épipolaires cohérents, même dans le cas de scènes planes où les épipoles ne peuvent être déterminés : on est ainsi assuré que le correspondant de chaque point est toujours sur la droite épipolaire qui lui correspond. Même si ces droites ne sont pas les véritables épipolaires, la contrainte qu'elles fournissent est équivalente du point de vue de l'appariement.

Corrections et complètement Les outils décrits dans les deux paragraphes précédents permettent d'améliorer de manière significative les appariements trouvés par l'algorithme présenté dans la première partie du chapitre.

- Le calcul robuste de la géométrie épipolaire et de l'homographie permettent de détecter les erreurs grossières d'appariement. L'erreur moyenne trouvée lors du calcul de l'homographie permet de savoir si la scène est presque plane ou non. Dans le premier cas, on utilisera plutôt la méthode de BOUFAMA pour le calcul de la géométrie épipolaire. Dans le deuxième, on accordera moins d'importance à l'homographie qui ne peut approcher le mouvement apparent que de manière imprécise.

- On peut aussi déduire de nouveaux appariements de ceux déjà réalisés. Si un segment s_1, d'extrémités a_1 et a_2, a une seule extrémité appariée, par exemple a_1 à un sommet b_1 de l'autre image, on peut essayer de trouver un point b_2 de l'autre image tel qu'il existe un segment entre b_1 et b_2, et que les points a_2 et b_2 soient compatibles pour la géométrie épipolaire et l'homographie.

- D'une manière plus générale, on peut chercher parmi les points non appariés des deux images tous les couples (a, b) qui vérifient les deux contraintes fournies par la géométrie épipolaire et l'homographie. Cette dernière méthode qui n'assure

aucune cohérence topologique des résultats trouvés vis à vis de la structure des images doit être maniée avec précaution.

2.1.3.1 Résultats expérimentaux

Premiers résultats. La phase de détection des erreurs grossières appliquée aux images des figures FIG. 2.1 et 2.2 permettent dans un premier temps de supprimer toutes les erreurs d'appariement. Après cette phase, il reste respectivement 22 et 86 appariements de points dans les exemples présentés.

La FIG. 2.3 montre les résultats obtenus après la phase de complètement des appariements. On a obtenu respectivement 35 et 237 appariements.

Il est, bien sûr, possible que quelques erreurs soient de nouveau présentes. Mais celles-ci ne peuvent être trop grossières, car elles doivent respecter la géométrie épipolaire et l'homographie.

D'un autre côté, certains points ne sont pas appariés. Cela vient du fait que l'homographie a été calculée à partir d'un nombre restreint de points et qu'elle ne donne donc une bonne approximation du mouvement apparent que dans une partie seulement de l'image. Dans le reste, elle empêche les nouveaux appariements.

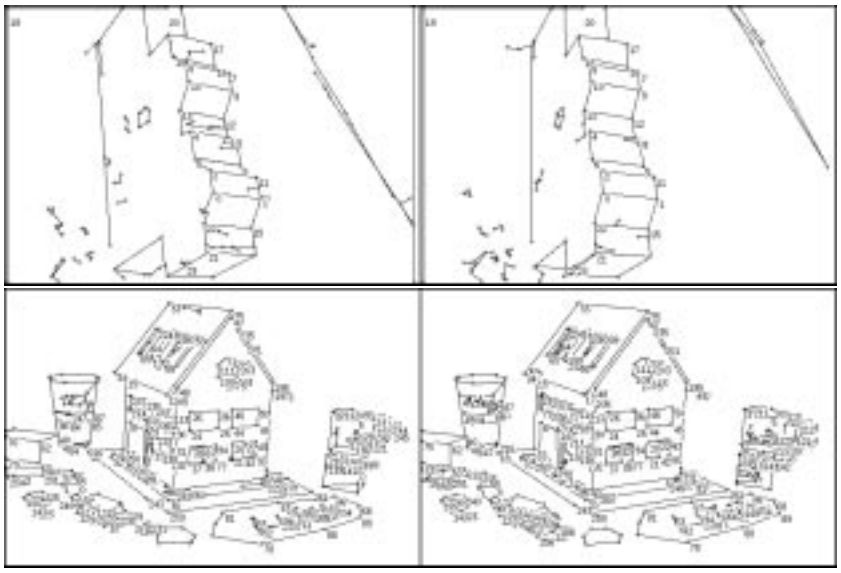

FIG. 2.3: *Amélioration des premiers appariements des* FIG. *3 et 4*

Limites de l'algorithme. Il est toujours intéressant pour juger un algorithme d'étudier les cas limites où il sort de son domaine de validité. Dans le cas présent, l'hypothèse la plus forte concerne l'approximation du mouvement apparent par une similitude, et il est donc important de voir jusqu'à quel changement de point de vue une telle approximation est valide.

Pour tester cela, on a pris une séquence de 17 images d'un même objet, le point de vue changeant légèrement d'une image à l'autre. L'image 0 a été appariée successivement avec toutes les autres. Le tableau 2.2 donne le nombre de points appariés et le nombre d'erreurs parmi ces appariements pour quatre méthodes différentes : appariement simple avec les similitudes (S), appariement avec les similitudes puis amélioration (S+), appariement simple avec les affinités (A), appariement avec les affinités puis améliorations (A+).

TAB. 2.2: *Nombre de points appariés et nombre d'erreurs parmi ces appariements avec quatre méthodes différentes.*

Image	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Nb App S	27	40	24	27	19	20	20	21	14	15	10	9	7	11	6	7
Nb Err S	0	0	0	0	3	3	0	3	3	0	3	0	0	0	0	0
Nb App S+	105	96	99	89	102	99	68	72	47	52	55	82	49	46	56	51
Nb Err S+	2	5	3	4	3	4	7	8	8	12	8	12	15	12	15	19
Nb App A	31	26	21	20	24	14	14	13	14	10	8	4	6	9	8	8
Nb Err A	2	0	2	0	0	1	0	0	0	6	4	4	6	9	8	4
Nb App A+	100	102	91	67	82	64	59	56	51	14	8	51	0	15	1	3
Nb Err A+	1	1	6	9	11	9	10	11	4	10	1	13	0	15	1	0

Les images 16 et 9 apparaissent comme les images limites avec lesquelles l'appariement soit possible lorsque l'on utilise une approximation euclidienne ou affine respectivement.

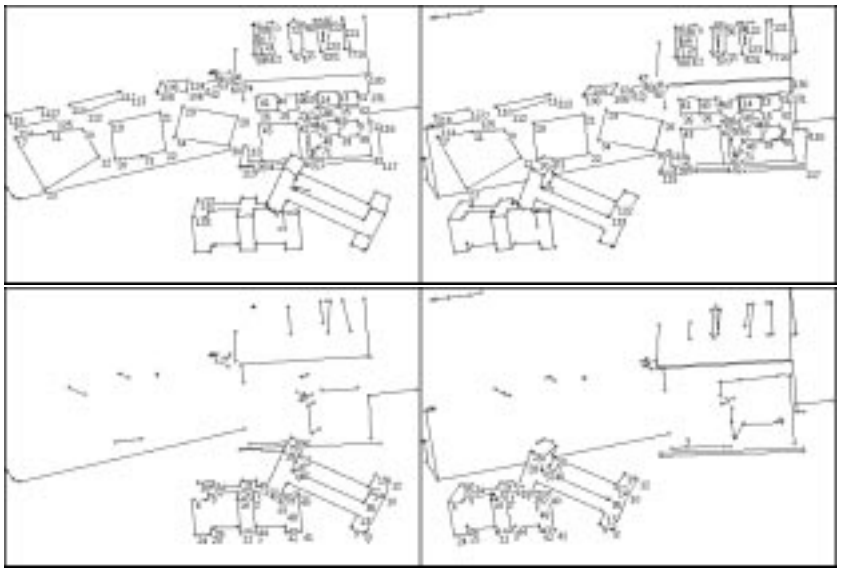

FIG. 2.4: *Résultat du premier (en haut) et du deuxième (en bas) appariement.*

Le cas des images avec des objets en mouvement. Une fois un premier appariement réalisé, de nombreuses possibilités sont ouvertes. On peut, par exemple, vouloir apparier des images avec des objets en mouvement.

Pour cela, on procède à un premier appariement qui donne le mouvement prépondérant dans l'image, puis on ôte les primitives correspondant à ce mouvement et on recommence un nouvel appariement qui donne cette fois-ci le mouvement secondaire. Il est bien sûr nécessaire que chacun des mouvements correspondent à un nombre suffisant de primitives.

La FIG. 2.4 montre les résultats obtenus sur un exemple : en haut le résultat du premier appariement et en bas celui du deuxième appariement.

2.1.4 Appariement de plus de deux images

Après avoir traiter du cas de l'appariement de deux images, un autre cas digne d'intérêt est celui de l'appariement simultané de n images. Deux solutions sont possibles : soit on essaie d'apparier toutes les images en même temps, soit on les apparie deux par deux puis, dans un second temps, on regroupe tous ces appariements partiels dans un appariement global.

Même si la première solution peut paraître plus satisfaisante, elle pose de redoutables problèmes de complexité. Elle nécessite, par ailleurs, une refonte très profonde de l'algorithme déjà développé.

La seconde méthode que nous avons retenue tire sa difficulté du fait que les appariements partiels ne sont pas cohérents entre eux. On peut représenter les appariements réalisés sous forme d'un graphe entre les primitives des différentes images dont les nœuds sont les primitives et les arêtes les appariements entre ces primitives. Le but est de déduire de ce graphe un hypergraphe dont les hyperarêtes représentent les appariements globaux. Autre manière d'exprimer le but, on cherche une partition du graphe des appariements dont chaque composante est un appariement global.

Partitionnement du graphe d'appariement

On définit la notion de *connexité forte* : deux primitives sont $k-$*fortement connectées*, s'il existe au moins k chemins de longueur inférieure ou égale à 2 entre ces deux sommets dans le graphe des appariements.

On peut alors chercher les classes d'équivalence de la fermeture transitive de cette relation de connexité forte pour un seuil donné. Pour ne pas avoir à fixer un seuil de manière arbitraire, nous proposons une méthode d'adaptation automatique de ce seuil. Notons $C_k = \{c_k^i\}$ l'ensemble des composantes obtenues avec la relation de connexité forte pour un seuil k. Il est clair que les composantes de C_{k+1} forment des partitions de celles de C_k. En particulier, toute composante c_{k+1}^i est incluse dans une composante c_k^j. Ces composantes peuvent donc être organisées dans un arbre : la racine contient l'ensemble des primitives de toutes les images. Au niveau 1, les nœuds sont les composantes c_1^i ; au niveau 2, on trouve les c_2^i... Les liens représentent les relations d'inclusion entre les composantes de deux niveaux consécutifs.

On utilise alors l'algorithme suivant où n est le nombre d'images, s_{haut} et s_{bas} sont deux seuils qu'on prend égaux à 1,2 et 0,8 dans la pratique.

- On commence à parcourir l'arbre à la racine.
- Si la composante c_k^i qu'on examine contient plus de $n.s_{haut}$ primitives, on examine alors récursivement ses fils. Si aucun d'eux ne peut former une composante finale, c_k^iest retenue comme composante finale.
- Si c_k^i contient moins de $n.s_{bas}$ primitives, elle n'est pas retenue comme composante finale.
- Si c_k^i contient entre $n.s_{haut}$ et $n.s_{bas}$ primitives, on la retient comme composante finale.

Post-traitements

On ajoute à cette recherche des composantes finales deux post-traitements.

Dilatation : lorsqu'une composante finale d'une part, n'a pas de frère dans l'arbre retenu comme composante finale et d'autre part, ne comporte pas de primitive

d'une image contrairement à son père, on ajoute à cette composante la primitive de l'image manquante qui est dans la composante père.

Réduction : lorsqu'une composante finale comporte deux primitives ou plus d'une même image, on ne laisse dans cette composante que la primitive de cette image la plus connectée au reste de la composante.

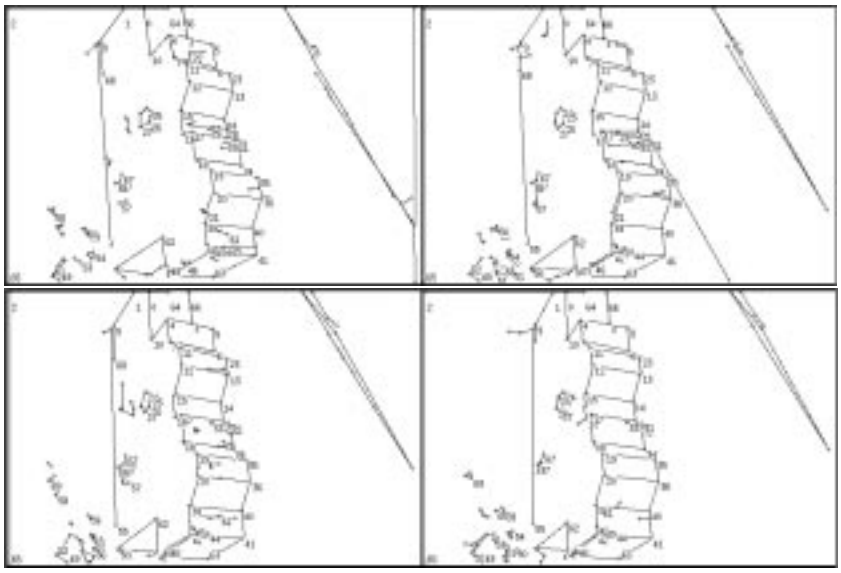

FIG. 2.5: *Appariement global d'un ensemble de 10 images (les images présentées sont les images 1, 4, 7 et 10 de la séquence).*

Résultats expérimentaux

On a utilisé l'algorithme précédent basé sur la connexité forte, l'adaptation automatique du seuil et les deux post-traitements, en prenant comme graphe de départ les appariements de points uniquement Les appariements de segments ont ensuite été déduits de ceux des points.

On a apparié 10 images de l'objet montré sur les FIG. 2.1 et 2.3. La FIG. 2.5 montre les résultats de l'appariement global sur les images 1, 4, 7 et 10. Les sommets appariés entre les images ont les mêmes numéros.

Lors de ce calcul, 70 appariements globaux ont été formés : 50 contiennent une primitive de chaque image, 5 en contiennent une de 9 images, 4 de 8 images, 4 de 7 images, 4 de 6 images, 1 de 5 images et 2 de 4 images.

2.2 Appariement d'images texturées

La méthode d'appariement qui fait l'objet de la première partie de ce chapitre est spécialement adaptée pour les images représentant principalement des objets polyédriques. Dans le cas d'objets texturés, à forte courbure ou sans structure forte, les segments de droite extraits de l'image ne sont pas stables et ne permettent donc pas de représenter les objets présents de manière adéquate. Par suite, la méthode précédente ne marche pas.

Parallèlement à nos travaux et indépendamment de ceux-ci, C. SCHMID a développé une méthode pour l'appariement des objets texturés. L'idée de base est de ne pas

s'appuyer sur les contours, mais sur des points d'intérêt à l'intérieur de l'objet, et d'utiliser l'information photométrique.

Les résultats obtenus sont très bons, car l'information photométrique est très riche, tellement riche, qu'elle permet d'utiliser la méthode telle quelle pour faire de la reconnaissance d'objets. Pour les images les plus difficiles, telles les images aériennes de ville, la méthode ne suffit plus à donner le meilleur résultat. Il faut alors soit utiliser une contrainte semi-locale, soit utiliser des probabilités associées aux invariants.

Cette méthode échoue toutefois sur certaines images encore, telles des images de moteur de voiture où la texture des objets est très dépendante du point de vue, en particulier à cause des réflexions et de la forte spécularité des surfaces. La méthode de SCHMID qui utilise cette texture ne peut plus alors trouver d'éléments d'accroche pour l'appariement.

Deux voies sont alors possibles : le passage aux images en couleur qui permettent de mieux faire face aux problèmes d'ombres et qui fournissent une information plus riche, et l'utilisation de méthodes plus robustes.

Dans la présente partie, nous présentons brièvement la méthode de SCHMID, car elle a orienté beaucoup des travaux qui seront présentés après.

Hypothèses et modèle de changement d'illumination

Les hypothèses de la méthode sont très semblables à celles faites dans la partie précédente. On souhaite apparier des images qui peuvent être différentes par le point de vue ou les conditions d'illumination, les objets pouvant s'occulter partiellement les uns les autres.

Par contre, les objets sont texturés et non plus structurés. Cela oblige à abandonner les contours pour utiliser des points d'intérêt. Le choix a aussi été fait d'utiliser directement le signal en niveau de gris. Il faut alors disposer d'un modèle décrivant l'influence, sur les valeurs des pixels, des changements d'illumination afin de pouvoir s'en abstraire.

On distingue, en général, deux types de changements d'illumination : les premiers sont les changements d'intensité ou de couleur de ces sources de lumière ; les seconds sont les changements de position ou d'orientation des sources. Pour reprendre une terminologie proche de celle en vigueur pour les paramètres de caméras, on appellera les changements du premier type changements internes et les seconds, changements externes de l'illumination.

Pour les images en niveau de gris, le modèle retenu pour les changements internes est le modèle affine. Les pixels sont supposés suivre une loi de type $I' = aI + b$ quand l'illumination varie. Ce modèle est une bonne approximation de la partir linéaire de la courbe qui fournit la réponse d'un pixel en fonction de l'intensité lumineuse qu'il reçoit (cf. FIG. 2.6). En revanche, il n'y a pas de modèle pour les changements externes.

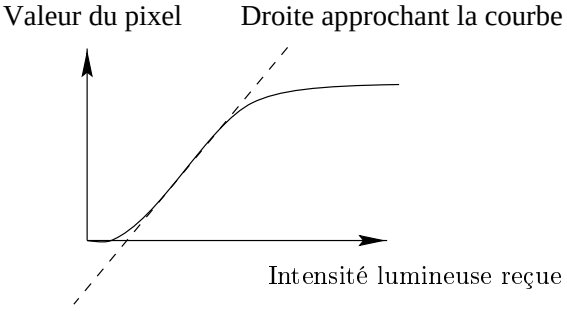

FIG. 2.6: *Courbe de réponse d'un pixel à une excitation lumineuse.*

Algorithme d'appariement

La méthode d'appariement est donc composée des étapes suivantes.

1° Les points d'intérêt sont extraits de l'image. Pour cela, on utilise un détecteur de HARRIS amélioré qui calcule en chaque point de l'image l'autocorrélation du signal : les points d'intérêt sont définis comme les points où cette autocorrélation est minimale, ce qui correspond à des points de singularité bidimensionnelle du signal.

2° Autour de chacun de ces points, on choisit un support pour le calcul des invariants. Afin d'obtenir des quantités invariantes à la rotation de l'image, les supports sont choisis circulaires.

3° Par des convolutions du signal des supports avec des dérivées de la fonction gaussienne, on calcule les dix premières dérivées du signal en chaque point d'intérêt. Si le signal de l'image est $I(x, y)$, on calcule donc I, I_x, I_y, I_{xx}, I_{xy}, I_{yy}, I_{xxx}, I_{xxy}, I_{xyy} et I_{yyy}.

4° Ces dix quantités sont ensuite mélangées afin d'obtenir des invariants géométriques et photométriques. Comme cela fait trois paramètres à éliminer (un pour la rotation et deux pour les changements d'illumination), cela donne sept invariants, écrits ici à l'aide de la notation d'EINSTEIN (sommation sur les indices répétés : $I_i I_i = I_x I_x + I_y I_y$ par exemple) :

$$\begin{pmatrix} I \\ I_i I_i \\ I_i I_{ij} I_j \\ I_{ii} \\ I_{ij} I_{ij} \\ \varepsilon_{ij} \left(I_{jkl} I_i I_k I_l - I_{jkk} I_i I_l I_l \right) \\ I_{iij} I_j I_k I_k - I_{ijk} I_i I_j I_k \\ -\varepsilon_{ij} I_{jkl} I_i I_k I_l \\ I_{ijk} I_i I_j I_k \end{pmatrix}$$

où $\varepsilon_{12} = -\varepsilon_{21} = 1$ et $\varepsilon_{11} = \varepsilon_{22} = 0$.

5° La comparaison directe des invariants entre eux fournit l'appariement final.

Cette méthode peut être améliorée à l'aide d'une contrainte semi-locale. Soient A et B deux points potentiellement appariables vu les valeurs de leurs invariants. On cherche les cinq plus proches voisins de chacun d'eux et l'on cherche si, d'une part, il y a deux autres paires de points appariables parmi ces voisins et si, d'autre part, les angles et les rapports de distances entre les points appariables de la première image sont les mêmes que ceux définis entre les points appariables de la deuxième image.

La prise en compte du facteur d'échelle entre les deux images ne peut pas être directe. Le problème ne vient pas des invariants, car on peut trouver des invariants qui ne sont pas sensibles à ce facteur d'échelle, mais du support de calcul. Il est en effet nécessaire de comparer des invariants qui intègrent les mêmes informations, et donc sont calculés sur des supports qui se correspondent. Dans l'ignorance du facteur d'échelle entre les images, on est dans l'impossibilité de trouver de tels supports.

Pour pallier à cela, on se sert du fait que les invariants, tels que présentés, offrent une petite robustesse au changement d'échelle : on calcule alors ces invariants pour une famille de supports de taille variable dans les deux images, et on compare les deux familles d'invariants ainsi obtenues. Cela permet de retrouver le facteur d'échelle.

La FIG. 2.7 montre un exemple de résultat obtenu avec cette méthode. Dans les images, les croix claires indiquent les points correctement appariés et les croix noires ceux qui ont été incorrectement appariés.

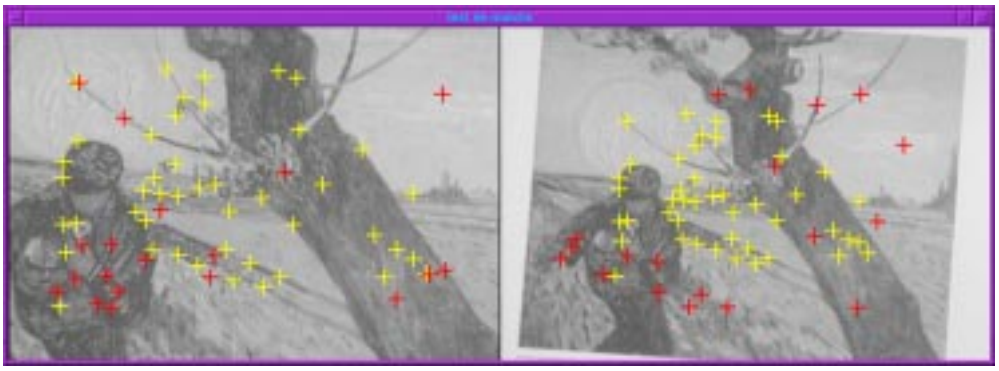

FIG. 2.7: *Appariement de deux images par la méthode de C.* SCHMID. *Le taux des appariements corrects est de 70,31 %.*

2.3 Appariement d'images colorées

Le cas des images en couleur est assez similaire à celui des images en niveaux de gris et on peut vouloir utiliser la même méthode qu'avec les images en niveaux de gris. La principale différence vient du modèle de changement d'illumination, qui ne doit plus décrire une unique valeur par pixel, mais trois composantes qui codent l'information de couleur. De tels modèles ont déjà été proposés dans la littérature, mais aucune validation expérimentale n'est généralement fournie.

Notre travail a consisté à évaluer expérimentalement divers modèles, pour en retenir un. Cela fait, nous avons adapté la méthode de SCHMID aux images en couleur.

2.3.1 Choix d'une représentation de l'information de couleur

Le choix d'un modèle de couleur peut se décomposer en deux parties : d'une part, il faut choisir le système de représentation de l'information de couleur et, d'autre part, décrire dans ce système comment varient les valeurs associées à un pixel lorsque les conditions d'illumination changent.

Le choix du système de représentation de l'information de couleur est tout d'abord contraint par les capteurs : les caméras couleur ont généralement trois séries de capteurs sensibles à des spectres lumineux centrés sur les longueurs d'ondes rouge, verte et bleue respectivement. Ce choix est dicté par le fait que la majorité de ces caméras a été développée pour enregistrer des images qui devront être ensuite visualisées par des observateurs humains. L'homme n'ayant que trois types de cellules réceptrices dans l'œil, il est inutile de stocker l'information de couleur avec plus de trois composantes puisque l'œil ne pourra pas percevoir de différence.

Ce choix pourrait être remis en cause pour des caméras destinées à un usage automatique, et les satellites proposent ainsi des observations dans un spectre beaucoup plus large. Le prix des capteurs n'est toutefois plus le même !

Les types de représentation de la couleur sont nombreux et classiques. On pourra en trouver un description dans [Poy97] par exemple. Pour l'analyse d'images et le calcul d'invariants photométriques en particulier, le choix d'une représentation dépend aussi de l'existence d'un modèle de changement d'illumination. Dans la suite, nous avons donc principalement utilisé le système RGB et marginalement le système HSV.

2.3.2 Évaluation expérimentale des modèles de changement d'illumination

Le problème est le suivant : on prend deux images d'une même scène. Entre ces deux prises d'image, ni la caméra ni la scène n'ont bougé. Seules les conditions d'éclairage de la scène ont varié. On peut distinguer deux types de variations. Tout d'abord, la lumière a pu varier en intensité ou en spectre de couleur ; on appellera une telle variation un variation interne des sources de lumière. Ensuite, les sources de lumière ont pu se déplacer ou changer d'orientation : on parlera dans ce cas de variations externes des sources de lumière. Un modèle de changement d'illumination décrit alors comment les valeurs (r, g, b) d'un pixel ont varié entre les deux images.

D'une manière générale, il n'est pas possible de trouver de modèle général de changement d'illumination. À cela plusieurs raisons : il faudrait disposer d'un modèle de réflectance de toutes les surfaces de la scène, et ce pour toutes les longueurs d'onde. On se contente donc le plus souvent de faire des hypothèses simplificatrices : on suppose ainsi les surfaces lambertiennes avec un spectre de réflectance très pointu. On utilise d'autre part deux modèles distincts pour les changements internes et externes des sources de lumière.

2.3.2.1 Modèle pour les changements internes des sources de lumière

Nous sommes partis des premiers travaux de FINLAYSON [FDF94]. Celui-ci propose de n'utiliser qu'un modèle diagonal : chaque vecteur $\mathbf{p} = (r, g, b)$ est multiplié par une matrice diagonale.

Pour vérifier ces résultats et les compléter, nous avons donc comparé douze modèles, comprenant chacun une partie multiplicative par la matrice identité, une matrice scalaire, diagonale ou pleine et une partie translation soit vide, soit composée d'un vecteur ayant ses trois composantes égales (translation simple), soit d'un vecteur avec trois composantes différentes :

Modèle n° 1 :	$\mathbf{p}' = \mathbf{p}$	identité
Modèle n° 2 :	$\mathbf{p}' = \mathbf{p} + \mathbf{t}$	translation simple
Modèle n° 3 :	$\mathbf{p}' = \mathbf{p} + \mathbf{T}$	translation
Modèle n° 4 :	$\mathbf{p}' = \alpha\mathbf{p}$	scalaire
Modèle n° 5 :	$\mathbf{p}' = \alpha\mathbf{p} + \mathbf{t}$	scalaire avec translation simple
Modèle n° 6 :	$\mathbf{p}' = \alpha\mathbf{p} + \mathbf{T}$	scalaire avec translation
Modèle n° 7 :	$\mathbf{p}' = \mathbf{D}\mathbf{p}$	diagonal
Modèle n° 8 :	$\mathbf{p}' = \mathbf{D}\mathbf{p} + \mathbf{t}$	diagonal avec translation simple
Modèle n° 9 :	$\mathbf{p}' = \mathbf{D}\mathbf{p} + \mathbf{T}$	diagonal avec translation
Modèle n° 10 :	$\mathbf{p}' = \mathbf{M}\mathbf{p}$	linéaire
Modèle n° 11 :	$\mathbf{p}' = \mathbf{M}\mathbf{p} + \mathbf{t}$	linéaire avec translation simple
Modèle n° 12 :	$\mathbf{p}' = \mathbf{M}\mathbf{p} + \mathbf{T}$	linéaire avec translation

Pour comparer ces modèles, nous avons pris une séquence d'images présentant une variation de l'intensité de la source lumineuse. Nous avons alors utilisé chaque modèle pour superposer au mieux chacune des images sur la première et nous avons mesuré l'erreur résiduelle.

Comme il est clair que les modèles ayant le plus de paramètres donnent les erreurs les plus petites, nous avons complété l'évaluation par un test statistique vérifiant si chacun des paramètres évalué est significatif ou non. Pour cela, on suppose que les erreurs ont une moyenne nulle et on calcule alors des régions de confiance pour chaque paramètre, c'est à dire des intervalles autour des paramètres estimés, tels que l'on soit sûr à 90% ou 95% qu'ils contiennent la valeur réelle du paramètre.

La formule donnant le rayon de l'intervalle de confiance pour un taux de confiance

de $\alpha\%$ est :

$$\sigma^2(p_i) = \sqrt{\chi^2(\alpha, m)}\sqrt{\frac{f}{n}}\sqrt{C_{ii}}$$

où :

- $\chi^2(\alpha, m)$ est le quantile de la distribution χ^2 pour un taux de confiance de $\alpha\%$ et m degrés de liberté ; ce nombre est le nombre de paramètres du modèle estimé ;
- f est la somme des erreurs sur tous les pixels et n est le nombre de pixels de chaque image ;
- C_{ii} est la variance du paramètre estimé p_i.

Si un paramètre p_i est compris entre $-\sigma^2(p_i)$ et $\sigma^2(p_i)$, on peut alors estimer que 0 qui appartient à la région de confiance aurait fourni une estimation aussi valable, et le paramètre p_i est jugé non significatif. Dans le cas contraire, il est jugé significatif.

Le tableau suivant présente les résultats obtenus en terme d'erreurs résiduelles après correction des images par l'un des modèles. Il ressort principalement de ces résultats que même les modèles les plus simples, pour autant qu'ils comportent une partie multiplicative, se montrent très adaptés.

Modèle	Moyenne des erreurs entre deux images				
	Im. 1 et 2	Im. 1 et 3	Im. 1 et 4	Im. 1 et 5	Im. 1 et 6
1	13.5362	38.6435	60.5171	83.4653	108.794
2	7.59494	15.3892	24.0688	35.0397	49.4871
3	7.37127	13.8359	20.8556	29.5884	40.9449
4	6.2232	6.98909	7.44423	7.29016	6.86809
5	6.17188	6.61222	7.00892	7.17807	6.5142
6	6.10906	6.26885	6.53342	6.7874	6.35143
7	6.08232	6.14477	6.27132	6.39796	6.81627
8	6.07727	6.07263	6.20241	6.39816	6.19346
9	6.07466	6.0483	6.13661	6.2572	6.09289
10	6.07104	6.09724	6.20927	6.29894	6.63859
11	6.06153	6.03857	6.15636	6.29847	6.0787
12	6.05874	6.02175	6.10728	6.20183	6.01641

Les meilleurs modèles sont les 9, 11 et 12. Dans les deux derniers cas, l'étude statistique sur les paramètres montre que les coefficients non diagonaux des matrices ne peuvent être évalués de manière significative que pour des images de taille relativement importante, au moins 60×60. Le modèle 9 apparaît donc comme le meilleur compromis entre sa précision et son nombre de paramètres. C'est celui que nous avons retenu. Par rapport au modèle de FINLAYSON, il comporte une partie additive supplémentaire qui permet de mieux rendre compte des courbes de réponse des pixels (cf. FIG. 2.6).

2.3.2.2 Modèle pour les changements externes des sources de lumière

Pour ces changements, nous avons simplement repris le modèle proposé par FINLAYSON. Dans ce modèle, chaque pixel est multiplié par une constante, constante différente pour chacun de ces pixels.

2.3.3 Normalisation des images

Il y a deux façons d'exploiter les modèles présentés à la section précédente. Soit on calcule des descripteurs des images qui sont invariants aux transformations utilisées pour le modèle, c'est ce qui est fait dans la section suivante, soit on normalise les

images. Cette normalisation consiste à transformer chaque image de telle sorte que le résultat de cette transformation soit indépendant des paramètres du modèle étudié.

Appelons images similaires toutes les images qui peuvent se déduire l'une de l'autre par un modèle de transformation tel que ceux étudiés précédemment. Une telle relation de similarité entre les images est une relation d'équivalence entre images. Calculer un descripteur invariant pour une image revient à calculer un descripteur pour la classe d'équivalence de cette image. Si ce descripteur peut être calculé de la même manière pour toutes les images, et varie d'une classe à l'autre, il permet de retrouver à quelle classe appartient une image.

Dans ce cadre, normaliser une image consiste à choisir une image particulière dans chaque classe comme représentante de cette classe d'équivalence pour la relation de similarité. Le choix de ce représentant doit être fait de sorte qu'on puisse le retrouver ou le calculer à partir de toute image de sa classe.

Nous avons proposé divers schémas de normalisation correspondant aux divers modèles de changements d'illumination exposés ci-dessus. Par exemple, si l'on étudie les problèmes du déplacement de la source, on peut calculer une image normalisée en multipliant chaque pixel par un facteur tel que le résultat vérifie $r+g+b=3$. L'image normalisée obtenue peut alors être calculée de manière analogue à partir de toute autre image prise dans les mêmes conditions que la première, sauf un éventuel changement de position de la source de lumière, dans la mesure où le modèle utilisé est correct.

Pour un changement interne de la source, il faut normaliser chaque canal. Supposons que l'image soit constituée d'un ensemble de n pixels (r_i, g_i, b_i). On peut alors transformer toutes les composantes rouges des pixels de l'image par $r_j = (r_j - m_r)/\sigma_r$, où m_r et σ_r sont respectivement la moyenne et l'écart type des valeurs des pixels rouges de l'image. De même pour les composantes vertes et bleues. On obtient alors une image qui ne dépend plus de la couleur ni de l'intensité de la source de lumière, dans la mesure où le modèle utilisé est juste, bien entendu. On peut aussi calculer des images normalisées pour les deux types de modèles à la fois.

Une autre alternative porte sur la région de normalisation : on peut soit faire une normalisation globale des images, mais qui devient alors dépendante de toute l'image et n'est plus robuste à de petits mouvements de la caméra, soit faire cette normalisation dans de petites fenêtres autour de chaque pixel. Ce dernier procédé est, bien entendu, beaucoup plus coûteux, mais il est plus robuste aussi (cf. FIG. 2.9).

Ces méthodes de normalisation ont deux intérêts : elles permettent tout d'abord d'évaluer les modèles : si ceux-ci étaient exacts, les images normalisées calculées à partir de quelques images ne présentant que des changements d'illumination devraient être égales, ce qui n'est pas le cas. Par ailleurs, réduire les différences entre images, même si ces dernières ne sont pas totalement supprimées, permet de meilleurs résultats avec certains algorithmes. Ainsi, la normalisation permet une plus grande répétabilité du détecteur de points d'intérêt par exemple.

2.3.4 Calcul d'invariants locaux

La deuxième manière d'utiliser les modèles de changement d'illumination présentés, outre la normalisation, est de calculer des invariants locaux similaires à ceux que SCHMID a proposé pour les images en niveaux de gris.

On peut suivre le même schéma de calcul : on détecte les points d'intérêt sur un des canaux ou sur une image gris calculée par combinaison linéaire des trois canaux RGB puis on calcule les dix dérivées d'ordre 0, 1, 2 et 3 de chacun des canaux. Il s'agit alors de combiner ces dérivées pour éliminer les paramètres de rotation et du modèle de changement d'illumination considéré.

Pour l'angle de rotation, on peut procéder avec chaque canal comme cela a été fait

pour les niveaux de gris. Cela donne neuf invariants pour chaque canal. L'angle étant identique pour les trois canaux, il existe deux invariants indépendants supplémentaires que l'on peut choisir parmi les 3 suivants :

$$R_xG_x + R_yG_y \qquad G_xB_x + G_yB_y \qquad B_xR_x + B_yR_y$$

Pour le modèle de changement d'illumination que nous avons retenu, qui comporte une matrice diagonale et un vecteur de translation, on peut remarquer qu'il se réduit à une transformation affine sur chacun des trois canaux. L'élimination de ces paramètres peut donc être menée comme cela a été fait pour les niveaux de gris. On peut remarquer que le fait d'avoir un terme de translation dans le modèle oblige seulement à retirer les invariants sans dérivées, ce qui n'ajoute pas de complication pour le gain que cela apporte.

Les trois invariants mélangeant les canaux doivent aussi être rendus invariants au changement d'illumination :

$$\frac{R_xG_x + R_yG_y}{(R_xR_x + R_yR_y)^{1/2}(G_xG_x + G_yG_y)^{1/2}} \qquad \frac{G_xB_x + G_yB_y}{(G_xG_x + G_yG_y)^{1/2}(B_xB_x + B_yB_y)^{1/2}}$$

$$\frac{B_xR_x + B_yR_y}{(B_xB_x + B_yB_y)^{1/2}(R_xR_x + R_yR_y)^{1/2}}$$

Nous avons réalisé diverses expériences pour tester ces invariants, en utilisant des séquences avec rotation de la scène autour de l'axe optique ou des séquences avec changement de l'illumination. Un des résultats les plus intéressants est celui montré ci-dessous. Ni la caméra ni la scène n'ont bougé entre les prises de vue, seule la position de la source principale de lumière a changé.

Ce cas est particulièrement difficile, car il provoque un fort déplacement des ombres. Bien qu'il ne soit pas parfaitement adapté, nous avons utilisé le modèle matrice diagonale et translation comme modèle de changement d'illumination.

La FIG. 2.8 montre les deux images extrêmes de la séquence. Chacune des images de la séquence est appariée avec la première. Les résultats sont montrés à droite de la même figure. Sur cette figure, l'abscisse indique le numéro d'image dans la séquence et, de haut en bas, les courbes indiquent le nombre de points d'intérêt dans la première image, le nombre de points dans l'image courante, le nombre d'appariements entre invariants et le nombre d'appariements corrects entre les invariants. On obtient donc de 10 à 15 appariements corrects.

On a ensuite recommencé la même expérience mais en normalisant préalablement toutes les images de la séquence par une technique locale pour le même modèle de changement d'illumination. Les deux images extrêmes normalisées et les résultats obtenus avec ces images sont montrées sur la FIG. 2.9. Dans ce cas, on obtient autour de 50 appariements corrects.

La différence peut s'expliquer par plusieurs facteurs. On peut tout d'abord remarquer que très peu de points ont été extraits sur la face sombre de la maison dans la deuxième image de la FIG. 2.8. Le détecteur de points d'intérêt n'a pas de connaissance des changements d'illumination et ne peut donc pas prendre en compte le fait que le signal est moins contrasté dans une région particulière. La normalisation assure une plus grande répétabilité de ce détecteur. D'autre part, même si la marque de l'ombre de la maison sur le sol n'a pas disparu, les effets des ombres sur la maison elle-même ont été fortement atténués.

2.4 Conclusion

Dans leur domaine respectif, chacune des trois méthodes présentées fournit de bons résultats, même si des améliorations sont toujours possibles. Le problème est

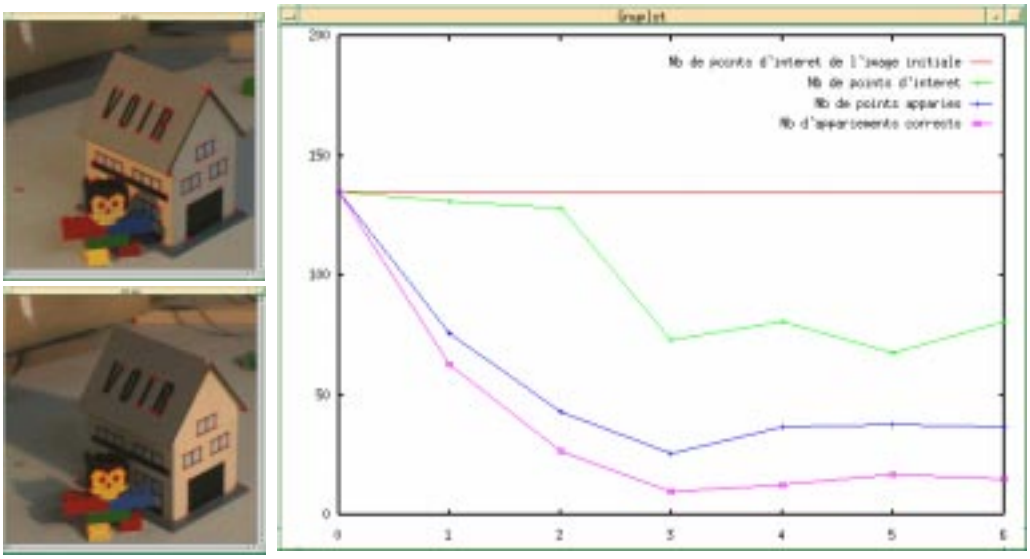

FIG. 2.8: *2 images d'une séquence prise avec déplacement de la source d'illumination et résultats de l'appariement en utilisant les invariants à la rotation et au changement interne de la source.*

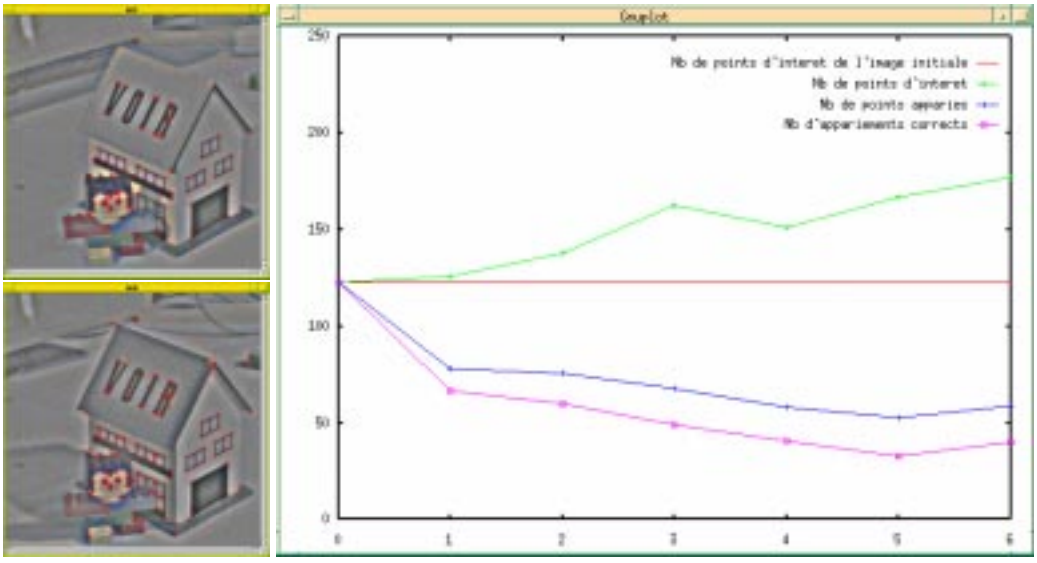

FIG. 2.9: *2 images de la séquence précédente, après normalisation locale et résultats de l'appariement.*

que chacun de ces domaines n'est pas délimité précisément, et qu'il n'est pas facile de déterminer a priori auquel appartient une image, ou comment faire si elle n'appartient à aucun des trois. Une autre difficulté est liée aux applications de ces algorithmes à l'indexation de grandes bases d'images. La variabilité de ce qui est observé peut être alors très grande, et aucune méthode ne sait vraiment la prendre en compte. Diverses propositions sont faite à ce sujet dans le dernier chapitre.

Chapitre 3

Modélisation

Je ne suis quand même pas assez insensé
pour être tout à fait assuré de mes certitudes.
Jean Rostand.

DANS CE CHAPITRE, nous nous intéressons tout d'abord à la construction de modèles d'objets pour la reconnaissance. Pour notre part, nous avons choisi de faire cette construction à partir d'images des objets concernés. Le problème difficile n'est pas tant celui de la construction effective du modèle à partir de quelques images, mais de choisir automatiquement ces images parmi toutes celles de l'objet qui sont disponibles. Notre travail a donc principalement porté sur les techniques de regroupement d'images, avec applications dans deux situations distinctes : la reconnaissance et la catégorisation.

On peut donner la définition suivante : la modélisation a pour but de construire quelques images permettant la meilleure reconnaissance possible à partir d'un ensemble d'images représentant toutes le même objet, éventuellement sous des points de vue variés.

Une première situation est celle où l'on ne dispose que d'une seule image ou que de quelques unes présentant des points de vue différents sur l'objet. Il n'y a pas alors de redondance de l'information et les images sont utilisées directement comme modèles. Cela veut dire qu'on leur fait subir le même traitement qu'une image à reconnaître : extraction des contours, approximation polyédrique, calcul des invariants, et on peut alors directement passer à l'étape d'indexation décrite dans le chapitre 4.

Le deuxième cas est celui où l'on dispose de nombreuses images de l'objet à reconnaître. Le but de la modélisation est de réduire l'information disponible pour réduire la taille de la base de modèles, tout en utilisant la redondance des données pour obtenir un modèle permettant une meilleure reconnaissance. On peut donc parler d'apprentissage puisqu'il s'agit d'obtenir un modèle plus performant que les données de départ.

L'hypothèse faite que toutes les images représentent le même objet n'est pas centrale : les divers points de vue vont être séparés, et que ces points de vue ou aspects correspondent au même objet ou à des objets différents n'intervient pas dans le processus de reconnaissance lui-même. Cela peut toutefois être utile d'associer à l'objet un nom ou des données auxiliaires : la notion qui est apprise de l'objet dépend alors des images – exemples qui sont fournies au système. Si elles sont incohérentes, la notion ou l'objet sera lui même incohérent.

D'une manière générale, la tâche de modélisation fait appel à trois opérations : il faut pouvoir comparer les données initiales, regrouper les données les plus proches, puis concentrer l'information de chacun des groupes formés pour en faire un modèle.

Dans ce chapitre, nous traitons de deux applications. La première est celle où nous disposons d'images d'un même objet et où l'on cherche une modélisation précise de cet objet. La deuxième est celle où l'on dispose d'un ensemble d'images représentant un concept plus général qu'un objet particulier, par exemple des immeubles (par opposition à un immeuble unique). Nous commençons par décrire l'algorithme de regroupement qui est commun aux deux cas cités, avant de revenir sur chacun d'eux.

3.1 Techniques de regroupement d'images

Modéliser nécessite de savoir faire trois choses : comparer les données de départ, regrouper ces données et déduire un modèle de chaque groupe de données. Si les étapes de comparaison des données et d'établissement du modèle sont spécifiques au type des données traitées, la phase centrale de regroupement des données est un problème général connu sous le nom anglais de *clustering*.

On dispose d'un ensemble de données et d'une mesure de similarité, le but est de partitionner ces données en des ensembles de données proches. De nombreux algorithmes existent pour résoudre ce problème : les nuées dynamiques, l'algorithme k-means, les méthodes agglomératives ou divisives... À vrai dire, la difficulté ne vient pas des méthodes elles-mêmes, mais du fait qu'elles ont besoin, sous une forme ou un autre, d'une indication sur la taille des groupes à former.

Nous présentons donc brièvement ces méthodes, avant de revenir plus en détail sur ce problème de l'arrêt.

3.1.1 Quelques méthodes de regroupement

Le problème du regroupement de données est classique en reconnaissance de formes, et on peut donc trouver de nombreuses références synthétiques sur le sujet : comme livres de références, on pourra se reporter à [DLPT82] ou à [LMF82] qui est plus simple d'accès ; dans les références plus récentes, citons [LMP95], nouvelle édition du précédent, [Rip96] qui nécessite de bonnes connaissances préalables en statistiques ou [MN96]. Divers algorithmes plus adaptés à la vision ont aussi été proposés [DS76, SLH90].

Les méthodes les plus classiques sont : k-means et ses variantes, k-medoids et le regroupement flou et les méthodes hiérarchiques, quelles soient divisives ou agglomératives.

Dans chacune des méthodes suivantes, on considère que l'on dispose d'un ensemble de données initiales et, suivant le cas, soit d'une matrice de distance fournissant les distances entre toutes ces données, soit de la représentation de ces données dans un espace de type $\mathbb{R}^n$. Il est clair qu'une telle représentation est bien plus riche que la simple donnée d'une matrice de distances et autorise des méthodes spécifiques à ce cas là. Cela explique que certaines méthodes ne puissent être applicables dans le cas des images.

3.1.1.1 k-means et variantes

k-means. L'algorithme de base est très simple [For65, Jan66, Mac67]. On suppose que l'on veut former k classes. On commence donc à diviser de manière arbitraire les données en k classes. On calcule alors, pour chaque classe, son centre de gravité G_i. On réaffecte alors chaque donnée P_j à la classe correspondant au centre de gravité le plus proche de cette donnée. Le processus est alors répété tant que les classes formées ne sont pas stables.

Formellement, si $C(P_j)$ est le numéro de la classe où est affectée la donnée P_j, la méthode consiste à minimiser l'expression :

$$\sum_j \|P_j - G_{C(P_j)}\|^2$$

Cet algorithme peut être légèrement modifié : à chaque étape, on essaye de changer une donnée de classe et on regarde si cela permet de diminuer l'expression à minimiser. Sous cette forme, il devient clair que l'algorithme qui minimise à chaque étape une quantité positive doit converger. Par contre, son caractère combinatoire apparaît nettement aussi.

Cet algorithme appelle quelques commentaires :

- il suppose que les données sont représentables par des vecteurs dans $\mathbb{R}^n$ pour que l'on puisse calculer des centres de gravité ;
- il faut lui fournir le nombre de classes souhaitées ;
- il faut remarquer qu'une des classes peut devenir vide, et donc au maximum k classes sont formées ;
- la solution trouvée peut dépendre du premier découpage fait ;
- pour que la somme à minimiser ne soit pas dominée par les distances les plus grandes, on peut minimiser la somme des distances plutôt que la somme des carrés des distances.

Dans le cas où l'on utilise des images, cet algorithme présente donc plusieurs difficultés. Tout d'abord, on utilise des descriptions d'images plus complexes qu'un espace $\mathbb{R}^n$ où la dimension de la description varie d'une image à l'autre. D'autre part, le nombre de classes souhaitées n'est pas toujours connu.

k-medoids. Pour résoudre le premier problème, une variante de l'algorithme précédent, appelée «k-medoids» ou «k-medians», a été proposée [Vin69, KR90]. Dans cette variante, on se restreint à choisir les centres des classes parmi les données. Par contre, l'algorithme reste le même : on choisit k centres, on affecte chaque donnée au centre le plus proche. On essaye alors de remplacer un centre par une autre donnée qui n'en est pas un et on garde cette interversion si elle réduit le critère à minimiser. Ce processus est répété jusqu'à convergence.

Dans cette variante, si d_{ij} est la matrice des distances ou des dissimilarités entre les données, l'expression à minimiser est de la forme $\sum_j d^2_{jC(P_j)}$, ou bien $\sum_j d_{jC(P_j)}$ si on veut limiter l'influence des grandes distances. La convergence est assurée de la même manière que pour l'algorithme précédent, et le caractère combinatoire demeure. En dehors du cas $k = 2$ où l'algorithme est assuré de converger vers le minimum global, ce dernier trouve généralement un minimum local.

Autres variantes. De nombreuses autres variantes ont été proposées. Bezdek [Bez74] et Dunn [Dun74] ont proposé des versions de l'algorithme k-means en utilisant la logique floue. Chaque donnée n'est plus affectée à une unique classe, mais peut être affectée à plusieurs, selon des coefficients dont la somme doit être égale à 1. Dans la même veine, Kaufman et Rouseeuw [KR90] ont proposé une version floue de l'algorithme k-medoids.

L'inconvénient de ces méthodes est de garder les inconvénients des algorithmes initiaux et d'en ajouter un supplémentaire : lorsque les classes ne sont pas clairement

séparées, chaque donnée peut appartenir à plusieurs classes, sans qu'il soit facile de n'en choisir qu'une seule à la fin.

Autre adaptation possible, le fait d'utiliser des distances différentes dans chaque classe [FR67]. Cela permet de simuler le fait que les classes auraient été obtenues selon des densités de probabilité différentes et permet donc d'obtenir des classes de formes variées. On trouvera encore d'autres variantes dans [SLH90, Rip96].

3.1.1.2 Les algorithmes hiérarchiques

Les algorithmes hiérarchiques partent sur une idée très différente des précédents. Ils ne sont, en effet, pas basés sur un nombre k de classes défini a priori, mais cherchent plutôt à proposer une hiérarchie de classes emboîtées sous forme d'arbre. À la base de l'arbre, chaque donnée est dans une classe différente. Au sommet, toutes les données sont dans la même classe. Entre les deux, on assiste à un regroupement progressif des classes, deux par deux. Pour obtenir un ensemble de classes, il suffit alors de couper l'arbre à une certaine hauteur.

Le problème du nombre de classes se situe donc en finale et est clairement distinct de la manière dont l'arbre des classes lui même a été construit. La figure 3.1 montre un nuage de points et la hiérarchie de classes correspondante (d'après DIDAY [DLPT82]).

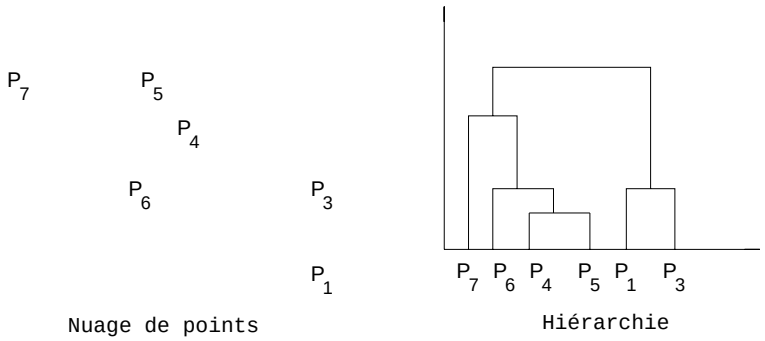

FIG. 3.1: *Un nuage de points et une représentation de la hiérarchie de classes correspondante.*

La construction d'une telle hiérarchie suppose qu'on dispose d'une mesure de ressemblance entre données individuelles, entre classes, entre classes et données individuelles. Une telle mesure est une distance si elle vérifie les trois axiomes des distances. Si elle ne vérifie pas l'inégalité triangulaire, on parle alors de dissimilarité. Cette mesure est souvent dénommée **indice d'agrégation**.

Indices d'agrégation. Ils sont le plus souvent calculés à partir de la matrice des distances entre données. Mari [MN96] indique divers indices classiques. Ainsi, la distance entre deux classes h_1 et h_2 contenant respectivement $n(h_1)$ et $n(h_2)$ données peut être :
- le minimum, le maximum ou la moyenne des distances entre éléments de ces classes ;
- la distance entre les centres de gravité de ces classes :

$$\delta(h_1, h_2) = d(G(h_1), G(h_2))$$

- la variation de l'inertie :

$$\delta(h_1, h_2) = \frac{n(h_1)n(h_2)}{n(h_1) + n(h_2)} d(G(h_1), G(h_2))$$

- la variation de la variance :

$$\delta(h_1, h_2) = \frac{n(h_1)n(h_2)}{n(h_1) + n(h_2)} d(G(h_1), G(h_2))^2$$

Le choix de cet indice influe fortement sur les classes formées. Ainsi, utiliser le minimum des distances entre éléments ne favorise pas la compacité des classes, contrairement à ce que donne l'emploi du maximum.

Méthodes divisives. Les méthodes hiérarchiques se distinguent les unes des autres selon la manière dont elles construisent l'arbre des classes. Les méthodes divisives, dont la plus connue est la méthodes des nuées dynamiques de DIDAY [DLPT82], consiste à couper le nuage des données par divisions successives.

Au départ, on cherche le centre de gravité du nuage des données. Par perturbation de ce dernier, on trouve deux points qui servent à affecter chaque donnée à une classe. On cherche alors le centre de gravité de chacune de ces deux classes, puis on recommence la division.

À cet algorithme de base sont souvent adjointes des heuristiques pour ne pas couper indûment des classes cohérentes par exemple.

Méthodes agglomératives. Elles procèdent à l'inverse des précédentes. Au départ, chaque donnée est dans une classe différente. À chaque étape, on fusionne les deux classes les plus proches. Il suffit alors de mettre à jour les indices d'agrégation, puis de recommencer.

La recherche des deux groupes les plus proches se fait par recherche de l'élément minimal dans la matrice des distances. C'est un processus de complexité quadratique.

La phase de mise à jour de la matrice est, par contre, simple. Il suffit de définir la valeur de la distance entre deux groupes formés de plusieurs images. On peut, par exemple, prendre cette distance égale à la moyenne des distances entre les images des deux groupes. On pourrait aussi prendre le maximum.

Il suffit alors de supprimer les deux lignes et deux colonnes de la matrice correspondant aux deux groupes fusionnés et de calculer une nouvelle ligne et une nouvelle colonne correspondant au nouveau groupe formé. Si I, J et K sont trois groupes d'images, et si on note $|L|$ le cardinal du groupe L, on remarque que :

$$\begin{aligned} d(I \cup J, K) &= \frac{1}{|I \cup J|.|K|} \sum_{i \in I \cup J, k \in K} d(i,k) \\ &= \frac{|I|}{|I \cup J|} d(I,K) + \frac{|J|}{|I \cup J|} d(J,K) \end{aligned}$$

Cette mise à jour est donc de complexité linéaire.

Le paramètre déterminant de la méthode est bien sûr le seuil qui va déterminer quand le processus doit être arrêté, avant que toutes les données ne se retrouvent dans un seul et même groupe.

Méthodes agglomératives linéaires. La méthode qui vient d'être présentée est très rapide, mais elle nécessite de stocker au début une matrice de taille quadratique en fonction du nombre des données, ce qui peut être un problème pour de très grandes quantités de données. Dans ce cas, il existe des algorithmes linéaires dont les résultats peuvent dépendre de l'ordre d'utilisation des données. De tels algorithmes sont classiques en recherche d'information [SM].

3.1.2 Le problème de l'arrêt

Toutes les méthodes de regroupement ont donc besoin d'un seuil qui indique, sous une forme qui peut être fort variée, un point d'arrêt. Cela vient du fait qu'il n'y a en fait que deux solutions triviales et naturelles au problème : celle où chaque élément est dans son groupe particulier et celle où tous les éléments sont dans le même groupe. Entre les deux, il y a une famille d'autres possibilités, mais il n'est pas possible d'en choisir une sans autre information.

Cette information peut être donnée sous de nombreuses formes :

- le nombre de groupes à former,
- la distance maximale entre deux éléments d'un même groupe,
- la distance minimale entre deux éléments de groupes différents,
- un critère de longueur minimale pour le résultat,
- le coût d'un groupe supplémentaire ou d'un groupe en moins,
- l'incertitude affectant chacune des données.

Dans le cas d'images, ces informations ne sont pas connues : par exemple, étant donné un ensemble d'images d'un objet, on ne connaît pas le nombre des aspects qu'il sera pertinent de garder, sachant que c'est justement l'une des informations que l'on cherche à déterminer. De même, la distance utilisée est peut être assez abstraite et peu intuitive, et donner des seuils portant sur cette distance peut être très délicat, car on ne maîtrise pas ce que ces seuils signifient en terme de ressemblance entre images.

Nous avons donc proposé un critère basé sur l'utilisation qui est faite des groupes formés. En effet, les groupes formés sont utilisés pour créer des modèles qui eux-mêmes servent à faire de la reconnaissance ou de la catégorisation d'images. Cette information, en soi, ne fournit pas de seuils, mais indique que l'on souhaite apprendre quelque chose lors de la phase de reconnaissance. Si l'on met toutes les images dans un même groupe, toute requête donnera ce groupe comme réponse et l'on n'apprendra rien.

Nous proposons donc d'évaluer un ensemble de groupes formés par une mesure de type entropie qui mesure l'information que l'on pourra obtenir lors de la phase de reconnaissance. On utilise ce critère comme suit : on fait fonctionner la méthode agglomérative en calculant l'entropie associée aux groupes formés à chaque étape. On garde alors comme résultat final les groupes qui donnent le résultat optimal. Plusieurs résultats obtenus avec cette méthode sont présentés dans la suite.

3.1.2.1 Entropie liée à un ensemble de groupes

En théorie de l'information, l'entropie associée à un ensemble d'événements possibles est :

$$\mathcal{S} = -\sum_i p_i \ln p_i$$

où p_i est la probabilité d'occurrence de l'événement i. On doit donc avoir $\sum_i p_i = 1$.

De manière imagée, rechercher le maximum de l'entropie revient à chercher à répartir une quantité (les probabilités) rare (du fait de $\sum_i p_i = 1$) en k ensembles, de manière la plus uniforme possible. Dans notre cas, cela se traduit par le fait de faire des classes ayant un critère commun le plus égal possible. On peut prendre comme critère la taille, la densité ou le nombre d'éléments. Nous proposons l'utilisation de critères basés sur la probabilité qu'une image inconnue soit classée dans un des groupes formés par l'algorithme de regroupement. Nous cherchons donc à faire des classes dont la

probabilité de reconnaissance est la même. La difficulté consiste donc à évaluer cette probabilité.

Hypothèse uniforme. Plusieurs hypothèses de calcul sont possibles. Supposons tout d'abord que les images soit réparties de manière uniforme dans « l'espace des images» et forment une population représentative des images qui seront à reconnaître. Il est clair qu'une telle hypothèse a un côté qualitatif qu'il peut être impossible de vérifier formellement la plupart du temps.

Dans cette hypothèse, la probabilité pour une nouvelle image d'appartenir à un des groupes est donc proportionnelle à la taille de ce groupe. Cette taille peut, par exemple, être estimée par la distance moyenne entre les images de ce groupe. Une telle distance peut être facilement maintenue à jour dans l'algorithme de regroupement, en utilisant les termes diagonaux de la matrice des distances. On a alors :

$$p_i \propto d_{ii}$$

Pour obtenir des probabilités, il faut ajouter un facteur de normalisation et prendre aussi en compte le fait qu'une image puisse n'appartenir à aucune image. Notons p_0 la probabilité correspondante. Intuitivement, on voudrait donc que p_0 soit une mesure de l'espace entre les groupes, ce qui est mesuré par les termes non diagonaux de la matrice de distances. Pour obtenir une distance correcte, il faut tenir compte du fait que ces termes sont en nombre quadratique par rapport au nombre des termes diagonaux. On peut donc prendre :

$$p_0 \propto \frac{1}{n} \sum_{i \neq j} d_{ij}$$

où n est le nombre de groupes formés.

Au final, cela donne :

$$\begin{aligned} p_i &= \frac{d_{ii}}{\sum_i d_{ii} + \frac{1}{n} \sum_{i \neq j} d_{ij}} \\ p_0 &= \frac{\frac{1}{n} \sum_{i \neq j} d_{ij}}{\sum_i d_{ii} + \frac{1}{n} \sum_{i \neq j} d_{ij}} \end{aligned} \tag{3.1}$$

Le facteur de pondération entre p_0 et les p_i est de grande importance. Si p_0 est trop fort, le système aura tendance à ne faire qu'un ou deux grands groupes pour contrebalancer l'importance du terme $-p_0 \ln p_0$ dans l'entropie. Une sous évaluation de p_0 bien que non souhaitable a un effet moins catastrophique.

Hypothèse non – uniforme. L'hypothèse d'uniformité est contestable : lorsqu'on modélise quelques objets, on dispose de nombreuses images proches de ces objets, mais cela ne donne guère idée de la diversité des objets que l'on n'a pas modélisés. On peut donc vouloir modifier les probabilités pour tenir compte du fait que quelques images très proches peuvent fournir un groupe aussi intéressant qu'un nombre plus faible d'images réparties dans une zone plus grande de l'espace.

Nous proposons de traduire cela en rendant p_i non seulement proportionnel à la taille du groupe, mais aussi au nombre d'images qui le forment. Cela donne :

$$\begin{aligned} p_i &= \frac{n_i d_{ii}}{\sum_i n_i d_{ii} + \frac{1}{n} \sum_{i \neq j} d_{ij}} \\ p_0 &= \frac{\frac{1}{n} \sum_{i \neq j} d_{ij}}{\sum_i n_i d_{ii} + \frac{1}{n} \sum_{i \neq j} d_{ij}} \end{aligned} \tag{3.2}$$

où n_i est le nombre d'images du groupe i.

Conclusion sur la méthode. La méthode d'arrêt que nous proposons n'est pas la panacée : il faut bien fournir une information à l'algorithme de regroupement pour l'arrêter, et notre méthode n'y déroge pas. Son originalité vient de la manière dont cette information est fournie : l'emploi de l'entropie est significative dans le cadre d'un système de reconnaissance où l'on souhaite que cette étape de reconnaissance soit le plus informative possible.

Cela déplace la difficulté qui est de donner des formules les plus plausibles possible pour les probabilités p_i et p_0. Certes, cet exercice est difficile, mais il nous paraît plus sensé que d'inventer des seuils qui ont de fortes chances de ne fonctionner correctement que pour les quelques deux exemples avec lesquels ils auront été testés.

3.1.3 Un exemple d'utilisation

Pour illustrer la méthode précédente, nous présentons ici des résultats avec des données synthétiques (des points du plan). Dans la suite du chapitre, deux autres exemples de vision seront plus détaillés. La difficulté de l'évaluation vient du fait que la réponse attendue de l'algorithme de regroupement n'est pas clairement formalisable : sinon il n'y aurait qu'à utiliser cette formalisation pour programmer la méthode.

Surtout dans le premier cas, il faut se rappeler que le but n'est pas de faire des groupes les plus séparés, mais les groupes qui permettront ensuite une classification informative.

Pour cet exemple, les points appartenant à une même classe sont liés entre eux par des segments de droite dans les figures.

On a utilisé une méthode de regroupement hiérarchique agglomérative, en utilisant l'inertie comme indice d'agrégation.

On a utilisé les formules 3.1 pour obtenir les classes montrées sur la partie gauche de la FIG. 3.2, et les formules 3.2 pour celles de la partie droite de la figure. On se trouve dans un cas où les points sont répartis de manière assez homogène, les classes contiennent des nombres de données proches, et les deux ensembles de formules donnent les mêmes résultats.

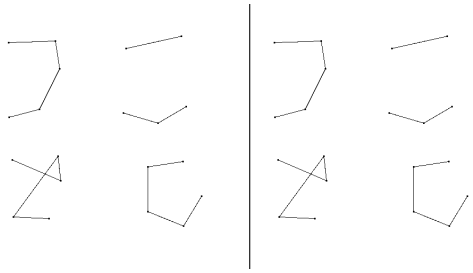

FIG. 3.2: *Catégorisation de points répartis de manière homogène.*

Il n'en est plus de même avec le FIG. 3.3 où on a utilisé les mêmes formules que précédemment, mais où les points sont placés de manière moins homogène. Avec la première formule, à gauche de la figure, la méthode a tendance à regrouper abusivement les classes pour lutter contre l'influence du terme p_0 dans l'entropie.

3.2 Modélisation des aspects d'un objet structuré

Disposant d'une méthode de regroupement, il est maintenant possible de traiter facilement de la modélisation des aspects d'un objet. Lorsqu'on veut reconnaître un

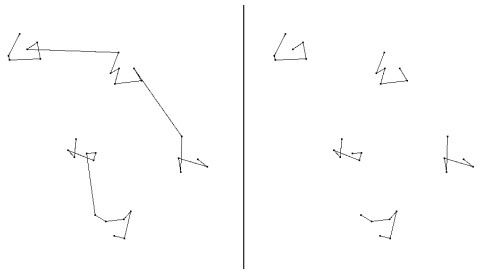

FIG. 3.3: *Catégorisation de points répartis de manière inhomogène.*

objet que l'on connaît à partir d'un ensemble d'images, on peut, bien entendu, mémoriser cet ensemble d'images. L'inconvénient est que cela revient à stocker tout le bruit présent dans les données et ne permet pas facilement d'utiliser la redondance de l'information présente dans les images.

Nous proposons donc d'abord de construire des modèles des aspects de l'objet et de ne stocker que ces modèles. Cela permet, d'une part, de réduire la quantité d'information à stocker et, d'autre part, de rendre le processus de reconnaissance plus robuste.

Pour cela, on dispose des outils nécessaires : une mesure de ressemblance entre images qui est fournie par l'évaluation des résultats de l'appariement réalisé à l'aide de l'algorithme du premier chapitre, et la méthode de regroupement présentée ci-dessus. Il ne manque plus alors que la phase de construction effective du modèle qui ne présente pas de difficulté majeure.

3.2.1 Principe de la modélisation

On dispose d'images de différents aspects d'un objet, et on aimerait établir automatiquement une modèle de chacun des aspects de l'objet présent dans les images. On utilise pour cela l'algorithme suivant.

1° Toutes les images sont appariées deux par deux.

2° Une mesure de dissemblance est évaluée pour chaque paire d'images, et le résultat est stocké sous forme d'une matrice de dissemblance.

3° L'algorithme de regroupement hiérarchique est utilisé pour rassembler en composantes les images présentant des aspects identiques ou très proches de l'objet.

4° Dans chacune de ces composantes, un appariement global des images est effectué à partir des appariements partiels obtenus à l'étape 1.

5° De chacun de ces appariements globaux de composantes, on tire un modèle de l'aspect représenté, ce qui termine le processus.

Les étapes 1 et 4 utilisent des algorithmes déjà décrits. La suite de ce paragraphe décrit donc les trois autres, et un exemple d'utilisation est montré.

3.2.2 Dissemblance entre images

Cette mesure a pour but d'évaluer la ressemblance ou la dissemblance du contenu de deux images. Pour réaliser cela, on évalue simplement le résultat de l'appariement des deux images : s'il est complet, les images se ressemblent. On utilise la formule suivante pour deux images I_1 et I_2 :

$$d(I_1, I_2) = \frac{2}{3} \frac{\text{nbsom}(I_1) \times \text{nbsom}(I_2)}{\text{nbsom}(I_1, I_2)^2} + \frac{1}{3} \frac{\text{nbseg}(I_1) \times \text{nbseg}(I_2)}{\text{nbseg}(I_1, I_2)^2}$$

Dans cette formule, nbsom(I) et nbseg(I) représentent respectivement le nombre de sommets et le nombre de segments de l'image I, et nbsom(I_1, I_2) et nbseg(I_1, I_2) le nombre d'appariements réalisés respectivement entre les sommets des images I_1 et I_2 et entre les segments de ces mêmes images.

Cette dissemblance vaut donc 1 quand les deux images sont complètement appariées et ∞ pour des images n'ayant aucun segment ou sommet apparié. Cela n'est donc pas une distance au sens mathématique du terme, mais est suffisant pour l'algorithme de regroupement.

3.2.3 Modélisation

À la dernière étape de l'algorithme, on dispose donc d'images regroupées en une partition de composantes et appariées globalement au sein de chacune de ces composantes. Pour établir un modèle à partir d'une des composantes, on choisit une image de celle-ci. Toutes les autres images (de la composante en question) sont transformées par homographie pour se rapprocher le plus possible de l'image choisie (en utilisant l'homographie calculée pour améliorer les appariements d'images deux par deux).

Le modèle est alors une nouvelle image formée des sommets et des segments présents dans au moins 60% des images de la composante. Chaque fois qu'une primitive (et les primitives qui lui sont appariées) est choisie pour faire partie du modèle, on calcule sa position moyenne dans toutes les images transformées par homographie. Cette position définit la position de la primitive dans le modèle.

Le taux de 60% choisi consiste à mettre dans le modèle les primitives présentes dans 2 images quand on part de 2 ou 3 images, présentes dans 3 quand on part de 4 ou 5 images, dans 4 sur 6, dans 5 sur 7 ou 8...

3.2.4 Résultats

Pour tester l'algorithme de séparation des aspects, on a utilisé l'objet montré à la FIG. 3.4, dont on a pris 78 images en faisant tourner la caméra autour de lui comme indiqué sur la figure. On obtient alors 10 ensembles d'images :

$$
\begin{aligned}
C_1 &= \{1, 2, 3, 33, 34, 35, 36, 37, 67, 68, 69, 70\} \\
C_2 &= \{4, 5, 6, 7, 38, 39, 40, 41, 71, 72, 73, 74\} \\
C_3 &= \{8, 9, 42, 43, 44, 45, 75, 76\} \\
C_4 &= \{10, 11, 12, 77, 78\} \\
C_5 &= \{13, 14\} \\
C_6 &= \{46, 47\} \\
C_7 &= \{15, 16, 17, 18, 48, 49, 50, 51, 52\} \\
C_8 &= \{19, 20, 21, 22, 23, 24, 53, 54, 55, 56\} \\
C_9 &= \{25, 26, 27, 57, 58, 59, 60, 61, 63\} \\
C_{10} &= \{28, 29, 30, 31, 32, 62, 64, 65, 66\}
\end{aligned}
$$

La FIG. 3.5 montre pour chacun des ensembles quelques unes des images qui en font partie, ainsi que le modèle de l'aspect correspondant de l'objet. Il faut remarquer que, l'objet étant presque symétrique par rapport à un plan vertical, chaque ensemble contient des vues prises de part et d'autre de l'objet.

Pour illustrer l'algorithme de modélisation, on a utilisé les images montrées aux paragraphes précédents qui offrent de plus nombreuses primitives. À partir de 10 images de l'objet de la FIG. 2.1, on a obtenu le modèle montré sur la gauche de la FIG. 3.6. Ce modèle contient 70 sommets.

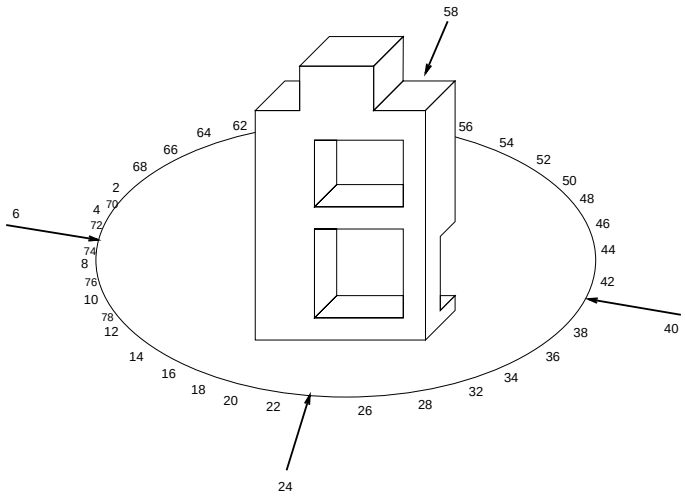

FIG. 3.4: *78 images d'un objet.*

À partir de 5 images de la maison de la FIG. 2.2, on a obtenu le modèle montré sur la droite de la FIG. 3.6. celui-ci comporte 181 sommets.

3.3 Modélisation d'ensembles d'images

La deuxième application de ce chapitre est la catégorisation des images. Dans ce cas, si l'on cherche toujours à regrouper des images suivant leur ressemblance, on ne cherche plus à établir d'appariements entre elles, mais on utilise des critères plus globaux. La tâche consiste alors à choisir un ensemble de descripteurs pertinents pour les images, puis à en déduire une mesure de distance ou de dissimilarité entre ces images.

Pour tester cette idée, nous avons choisi un ensemble simple de descripteurs. On commence par extraire de chaque image ses niveaux de couleur, ses niveaux d'intensité, ses contours, les segments approchant ces contours et les paires de segments parallèles.

À partir de ces primitives, on calcule les distributions dans chaque image des grandeurs suivantes : les niveaux de couleur et d'intensité, la densité, l'orientation, la longueur et la position des contours, des segments et des segments parallèles, ainsi que les angles formés entre les segments adjacents. Ces distributions sont calculées sous forme d'histogrammes.

La ressemblance entre deux images est alors estimée en comparant les histogrammes associés à ces images. La distance entre deux histogrammes (h_i) et (h'_i) est définie par $d(h,h') = \sum_i \frac{(h_i - h'_i)^2}{(h_i + h'_i)}$. Pour obtenir la distance entre les images, on fait une somme pondérée des distances entre histogrammes : $d(I,I') = \sum_h k_h d(h(I), h(I'))$. On peut prendre comme coefficients de pondération l'inverse de la variance des distances entre histogrammes calculée sur une grande base d'images. On utilise alors la matrice des distances entre images obtenue comme donnée d'entrée de l'algorithme de regroupement.

La figure 3.7 présente un exemple de résultats obtenus avec cette méthode. On part d'une séquence de 44 images prise à bord d'un véhicule à intervalles réguliers. Les images ont été classées en 4 groupes, et 3 images se sont retrouvées isolées dans 3 autres groupes ne contenant qu'une seule d'entre elles.

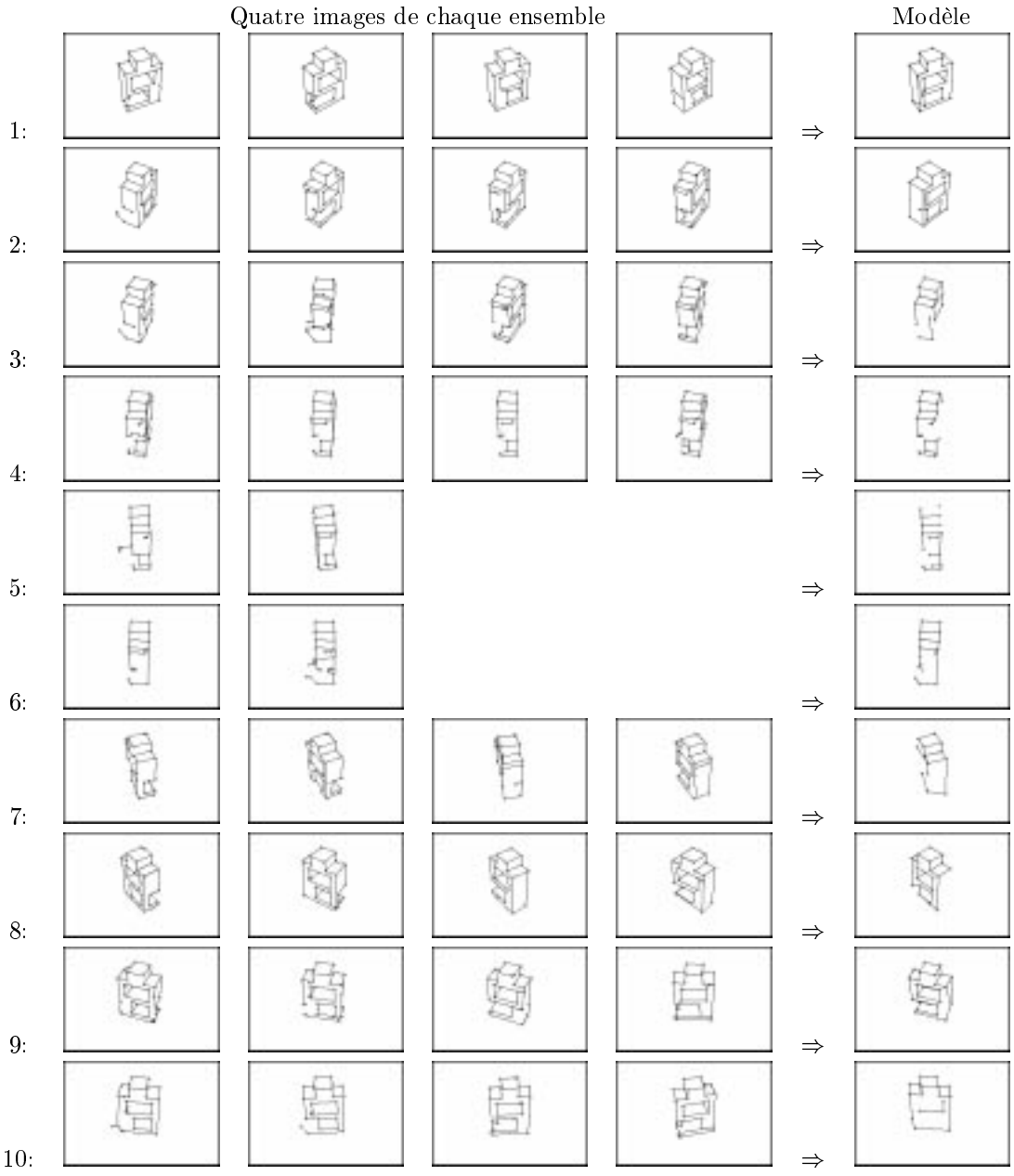

FIG. 3.5: *Pour chaque ensemble, quatre images et le modèle obtenu.*

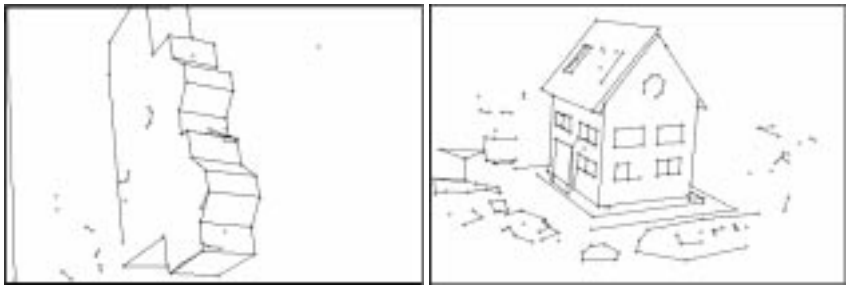

FIG. 3.6: *Les deux modèles obtenus.*

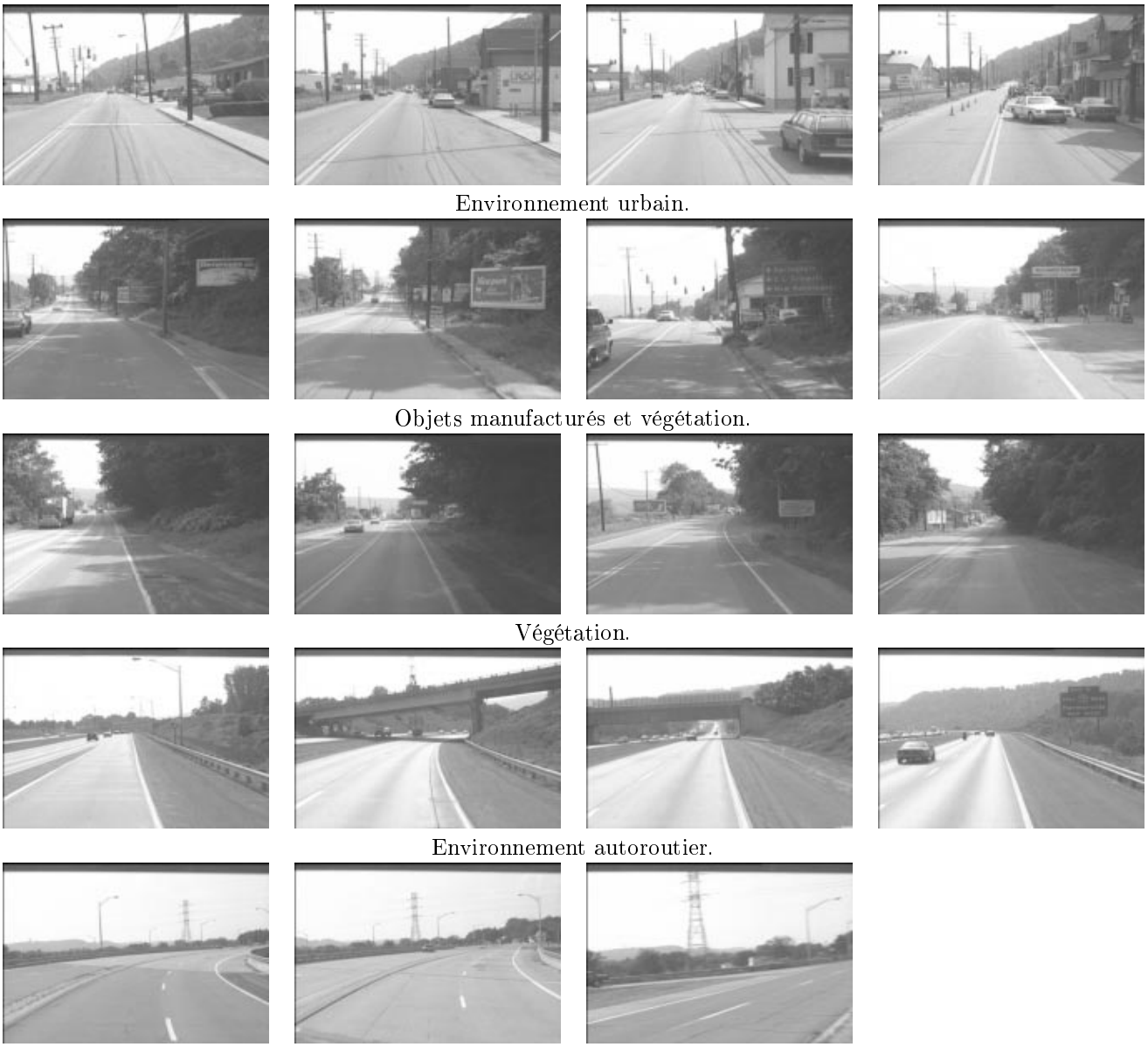

FIG. 3.7: *Quatre classes et 3 trois images singulières.*

3.4 Conclusion

L'évaluation des techniques présentées dans ce chapitre est très difficile. Le résultat attendu d'un algorithme de regroupement d'images est souvent subjectif : l'utilisateur aimerait que telle image soit avec celle-là, mais sans être en général capable de justifier son choix en termes de primitives des images. Il y a donc un véritable travail de fond à mener pour choisir la description la plus pertinente des images dans un contexte applicatif donné.

La deuxième difficulté vient du choix de la méthode et du test d'arrêt. Partant de simples matrices de distance, les méthodes agglomératives ou k-medoids sont les choix les plus raisonnables au premier abord. L'inconvénient de la première est d'être quadratique lors de la recherche de l'élément minimal de la matrice de distance. Celui

de la deuxième est de ne pas fournir des résultats stables du premier coup : il est nécessaire de faire des itérations. D'autre part, si la complexité est plus faible que quadratique à chaque itération, elle s'en rapproche lorsque le nombre de germes est grand, et il faut de plus déterminer un critère d'arrêt des itérations.

Au final, seule l'expérience peut trancher sur l'intérêt de telle méthode ou de tel test d'arrêt. L'utilisation de benchmark peut sembler intéressante, mais il ne mène le plus souvent qu'au développement de méthodes adaptées pour ces benchmarks !

Un cas particulier qu'il serait intéressant d'explorer est celui où l'on traite une séquence d'images que l'on souhaite couper en plans élémentaires. Ce problème peut être vu comme un problème de regroupement, mais où l'ordre des images doit alors être pris en compte. On peut utiliser la méthode agglomérative qui est bien plus simple dans ce cas, car la recherche du minimum peut être restreinte aux composantes se situant juste au dessus de la diagonale principale. Le critère de coupure peut être dans ce cas un critère d'homogénéité des plans formés.

Chapitre 4

Indexation d'une base d'images

Ce que nous savons est le plus grand obstacle
à l'acquisition de ce que nous ne savons pas.
Claude Bernard.

Ce chapitre présente le troisième axe important de notre travail. Il s'agit de transformer l'algorithme d'appariement en un algorithme d'indexation pour pouvoir faire de la reconnaissance. Mais une telle indexation ne peut être totalement utile que si elle peut être utilisée dans un système de gestion d'images en mémoire secondaire et si la réponse à une requête garde une complexité acceptable.

Nous abordons donc ces différents points :

1° la mise au point d'un algorithme d'indexation,

2° l'étude de la complexité de cet algorithme,

3° l'utilisation de la mémoire secondaire pour le stockage des données.

4.1 Un algorithme d'indexation

L'objectif de l'algorithme est de pouvoir comparer une image inconnue à un ensemble d'images. Vu que cet ensemble peut être volumineux, la comparaison ne peut s'effectuer image par image, il faut faut trouver un moyen de factorise cette tâche, c'est à dire de traiter les images de la base toutes en même temps, et non les unes après les autres.

Pour cela, on essaie de précalculer le maximum d'éléments sur les images de la base indépendamment de l'image inconnue. Ce sont ces éléments que l'on va stocker en mémoire, pour qu'il reste le moins de travail possible à faire lorsque l'image inconnue sera présentée au système.

Dans les algorithmes d'appariement que nous avons présentés, la phase critique qui nécessite absolument de disposer de l'image inconnue est celle de comparaison entre les invariants de l'image inconnue avec ceux des images de la base. On va donc précalculer tous les invariants de toutes les images de la base dans un premier temps et on va les stocker dans un même espace ou une même structure. En une seule recherche, il sera donc possible de retrouver tous les invariants de la base semblables à un invariant donné.

L'autre phase qui mérite attention est celle de vote dans le cas des images structurées. Pour chaque image, il faut stocker des transformations, puis rechercher un point d'accumulation dans l'espace de représentation de ces transformations.

Dans chacun de ces deux cas, on est amené à stocker des vecteurs de réels imprécis, puis à faire des requêtes sur le nombre de ces vecteurs qui sont égaux à un vecteur donné, ceci à un seuil près. Vu que les différentes composantes de ces vecteurs ont des ordres de variation très différents, il est possible d'utiliser la distance de MAHALANOBIS : si $\mathbf{C}$ est la matrice de covariance des vecteurs, la distance entre deux vecteurs $\mathbf{v}$ et $\mathbf{w}$ est : $\sqrt{(\mathbf{v}-\mathbf{w})\mathbf{C}^{-1}(\mathbf{v}-\mathbf{w})}$. On décomposant cette matrice sous forme diagonale, $\mathbf{C}^{-1} = {}^t\mathbf{P}\mathbf{D}\mathbf{P}$, on peut ramener cette distance à une distance euclidienne entre vecteurs :

$$\sqrt{(\mathbf{v}-\mathbf{w})\mathbf{C}^{-1}(\mathbf{v}-\mathbf{w})} = \|\sqrt{\mathbf{D}}\mathbf{P}\mathbf{v} - \sqrt{\mathbf{D}}\mathbf{P}\mathbf{w}\|$$

En appliquant préalablement la fonction $\mathbf{v} \mapsto \sqrt{\mathbf{D}}\mathbf{P}\mathbf{v}$ à tous les vecteurs, on peut alors travailler dans l'espace euclidien classique $\mathbb{R}^n$. Une autre solution est d'utiliser des seuils ε_i différents pour chacune des composantes. On suppose dans la suite que la première solution a été choisie.

Le problème est alors le suivant : étant donné un vecteur $\mathbf{v}$, trouver tous les vecteurs $\mathbf{w}$ de la base tels que $\|\mathbf{v}-\mathbf{w}\| \leqslant \varepsilon$. Ce problème se rapproche d'un autre problème, très étudié dans la littérature [Yia93, NN97, IM98], celui de la recherche des k plus proches voisins d'un point dans un espace multidimensionnel. La différence est que nous cherchons pas un nombre fixé de voisins, mais ceux qui se trouvent à une distance inférieure à un seuil donné. Au delà des différences, cette similarité entre les deux problèmes peut servir de source d'inspiration.

Les algorithmes de recherche de plus proches voisins se divisent en deux classes : ceux qui fonctionnent par partition des données, et ceux qui fonctionnent par partition de l'espace. Dans la première catégorie, on trouve par exemple les R-trees et les M-trees, dans la seconde les quad-trees et les *kd*-trees [Ben75, PdSMS95, PTSE95]. Les limites des premières étant déjà connues [BW97], et les secondes nous ayant semblé plus faciles à mettre en œuvre dans un premier temps, nous avons donc commencé nos expérimentations avec ces dernières. C'est aussi pour ces méthodes que nous avons mené l'étude de complexité. Celle-ci montre clairement quelles sont leurs limites, qu'il faudra mettre en balance avec celles des premières méthodes. Notre choix n'est donc pas définitif.

4.1.1 Indexation de vecteurs imprécis par découpage de l'espace

Principe

Nous avons donc testé une méthode de découpage de l'espace. Dans une telle méthode, on adopte un découpage a priori de l'espace en hypercubes dans lesquels on range les vecteurs. Si les données étaient exactes, il suffirait alors de rechercher, pour chaque vecteur requête, l'hypercube correspondant qui fournirait la liste des tous les vecteurs potentiellement égaux dans la base. Quelque soit la taille de l'espace, cette recherche peut être menée de manière efficace, le problème étant plutôt celui du nombre de hypercubes à stocker.

Malheureusement, les vecteurs ne sont pas exacts. Pour trouver tous les vecteurs proches d'un vecteur donné, on dispose de plusieurs solutions. Soit il faut chercher dans plusieurs hypercubes voisins, soit il faut que chaque hypercube contiennent non seulement ses vecteurs, mais aussi la liste des vecteurs qui sont situés à moins de ε de ces bords. Cela revient à prendre des hypercubes plus grands qui se chevauchent et à mettre chaque vecteur de la base dans plusieurs hypercubes. On voit tout de suite le coût additionnel pour la complexité en place d'une telle méthode quand le nombre de vecteurs à stocker augmente et que la dimension est élevée.

Pour la première solution, il faut veiller à la complexité aussi. Si on cherche les

invariants à ε près, le fait de prendre des hypercubes de 2ε de côté permet de ne visiter que 2 hypercubes par dimension, au lieu de 3 pour des hypercubes de ε de côté. Ce gain peut paraître futile, mais vu la dimension des espaces et le nombre d'invariants à chercher, il est en fait très appréciable dans la pratique.

Un inconvénient majeur de cette méthode est que l'on cherche en fait les invariants se trouvant dans une hypersphère, et que l'on approche cette hypersphère par l'hypercube englobant. Le volume d'une hypersphère de rayon r dans un espace de dimension n est $(r^n \pi^{n/2})/(n/2)!$. Le volume de l'hypercube de côté $2r$ est $(2r)^n$. Le rapport entre ces deux volumes tend rapidement vers 0 quand la dimension n de l'espace augmente.

Il faut donc s'attendre à une surcoût croissant avec la dimension, du fait que l'on va visiter un volume de plus en plus grand en pure perte. Ceci est contrebalancé par le fait que l'espace sera d'autant plus vide que la dimension augmente, et que chaque hypercube contiendra donc moins de vecteurs. Tout ceci est évalué plus précisément au paragraphe 4.2.

Implémentation du découpage

Ce découpage est mis en œuvre sous forme d'un *kd*-tree : à chaque niveau de l'arbre correspond un découpage de l'espace selon l'une de ses dimensions. L'arbre a ainsi une profondeur fixe égale a la dimension de l'espace. Les noeuds ne sont développés que s'ils contiennent des vecteurs. Cela permet un accès rapide à une noeud donné en limitant l'espace mémoire utilisé.

4.1.2 Reconnaissance d'objets

Algorithme

L'algorithme de reconnaissance complet comprend donc les étapes suivantes.

1° Les invariants de toutes les images de base sont calculés hors ligne et rangés dans une structure implémentant le découpage de leur espace de représentation. On peut à ce stade, calculer plusieurs types d'invariants pour chaque image : chaque type est stocké dans un espace particulier.

2° Les invariants de l'image inconnue sont calculés, puis confrontés à ceux de la base. Pour chaque invariant, il faut donc rechercher l'hypercube correspondant, puis les hypercubes voisins, puis tous les invariants contenus par ces hypercubes. L'espace visité étant bien plus grand que nécessaire, il faut comparer séquentiellement tous ces invariants avec l'invariant requête, pour ne calculer que les transformations correspondant aux paires d'invariants distants de moins de ε.

3° On associe à chaque image de la base un espace de vote, qui est un espace de représentation des transformations. Cet espace est prédécoupé comme ceux servant pour l'indexation. On associe à chaque hypercube un compteur qui dénombre les transformations présentes dans cet hypercube et celles se trouvant proche de ses bords. Chaque transformation calculée à l'étape précédente correspond ainsi un vote pour une image, qui permet de mettre à jour les compteur de l'espace correspondant.

4° La mise à jour des compteurs qui est effectuée à chaque vote permet de connaître à chaque instant le point d'accumulation maximale dans chacun des espaces de vote des images de la base, ainsi que l'image ayant la plus grosse accumulation et qui est donc l'image la plus proche de l'image inconnue.

Pour ses invariants, SCHMID a utilisé un algorithme assez similaire : *kd*-tree pour l'indexation des invariants de texture, comparaison en tenant compte des impréci-

sions, mais le vote est soit plus simple, car il suffit de compter le nombre d'invariants semblables entre deux images, soit un peu plus complexe en prenant en compte une contrainte semi-locale.

Ceci peut donc aussi s'étendre naturellement au cas des invariants de texture en couleur. La difficulté, comme le montre l'étude de complexité, vient de leur grande dimension.

Résultats

Cette technique n'offre pas de nouvelles possibilités d'appariement par rapport à ce qui a été vu dans le premier chapitre. La seule chose intéressante est d'évaluer les performances en termes de nombre d'images qui ont pu être utilisées à la fois dans la base, et de performance de reconnaissance. Dans le cas des images structurées, diverses expérimentations ont été réalisées avec des images de difficulté variable.

Dans une première expérience, on a utilisé 200 images obtenues à partir du modèle CAO d'un objet simple. Les liens entre le modèle CAO et les images ont été mémorisés. Les images ont été appariées et 34 d'entre elles ont été sélectionnées comme modèles 2D de l'objet, selon la procédure décrite au chapitre 2. On a alors présenté au système des images montrant cet objet, afin de reconnaître son orientation, puis de calculer sa pose et de le saisir à l'aide d'une pince.

Le taux de bonne reconnaissance obtenu est supérieur à 95 %. On peut classer les échecs en deux catégories. Tout d'abord, ceux où la reconnaissance a fourni un mauvais appariement, empêchant la saisie. De tels cas sont le plus souvent dus à des vues critiques où l'image de l'objet se réduit, par exemple, à un rectangle. Ensuite, on trouve les cas où la reconnaissance n'a pas retrouvé l'image la plus proche en terme de point de vue, mais où l'appariement est toutefois correct, autorisant la saisie. De tels cas sont liés à la dégradation des structures dans l'image inconnue, mais ne prête pas à conséquence. Cela souligne toutefois l'intérêt d'une certaine redondance dans la base d'image.

La FIG. 4.1 montre les images qui ont été sélectionnées et la FIG. 4.2 montre un exemple de reconnaissance : on y voit l'image de départ, le résultats de la segmentation et les trois modèles qui ont obtenu le plus de votes.

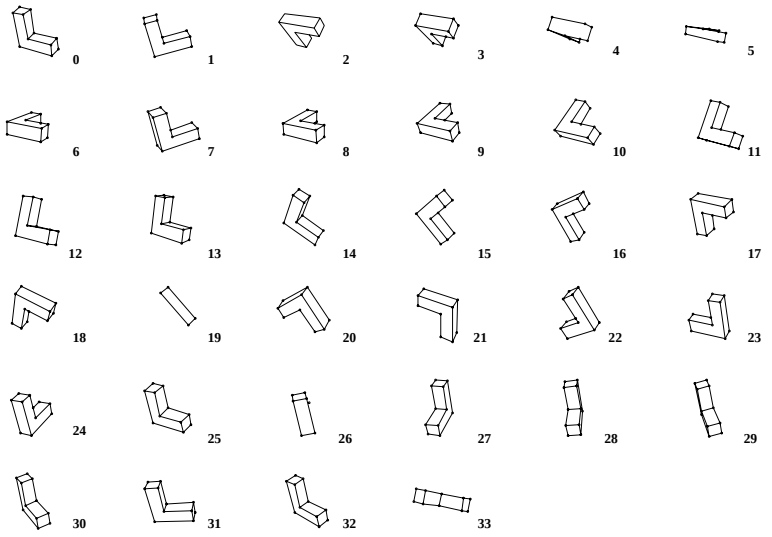

FIG. 4.1: *Les images sélectionnées comme modèles 2D de l' objet.*

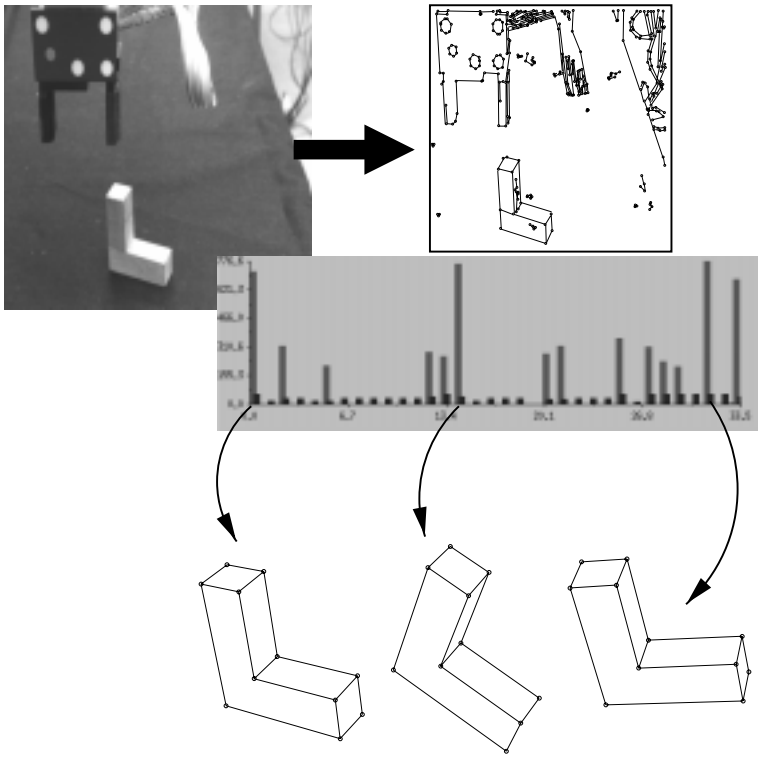

FIG. 4.2: *Reconnaissance d'un objet pour la saisie.*

Une deuxième expérience a été faite avec des images de moteurs de voitures. Dans ce cas, la base n'était constitué que de 9 images. Les images de la base ainsi que les requêtes ayant désigné ces images sont montrées sur la FIG. 4.3.

Dans cette expérience, deux problèmes sont apparus. D'une part les images sont complexes et la segmentation ne fournit pas de bons résultats. Si les quelques primitives qui peuvent être appariées suffisent à réaliser un appariement qui peut ensuite être amélioré en utilisant des outils tels la géométrie épipolaire par exemple, la mise en concurrence des plusieurs images amène un taux de reconnaissance bien plus faible que précédemment.

D'autre part, la segmentation, si elle ne fournit guère d'information pertinente, donne naissance à de nombreux petits segments et points d'intérêt, qui augmentent énormément la complexité en temps et en place (on peut saturer la mémoire avec une dizaine d'images complexes).

Ces difficultés ont amené à des recherches dans deux directions : une étude plus précise de la complexité de l'algorithme, détaillée dans le paragraphe suivant, qui montre que la plupart des invariants utilisés sont de trop faible dimension, et l'utilisation de la mémoire secondaire pour le stockage des données, qui fait l'objet de la section 4.3.

4.2 Complexité de la reconnaissance

Nous avons cherché à étudier la complexité de l'algorithme de reconnaissance, pour évaluer l'influence des divers paramètres du problème, et en particulier celle de la dimension des invariants. Cette étude n'est valable que pour les algorithmes procédant par découpage de l'espace des descripteurs des images.

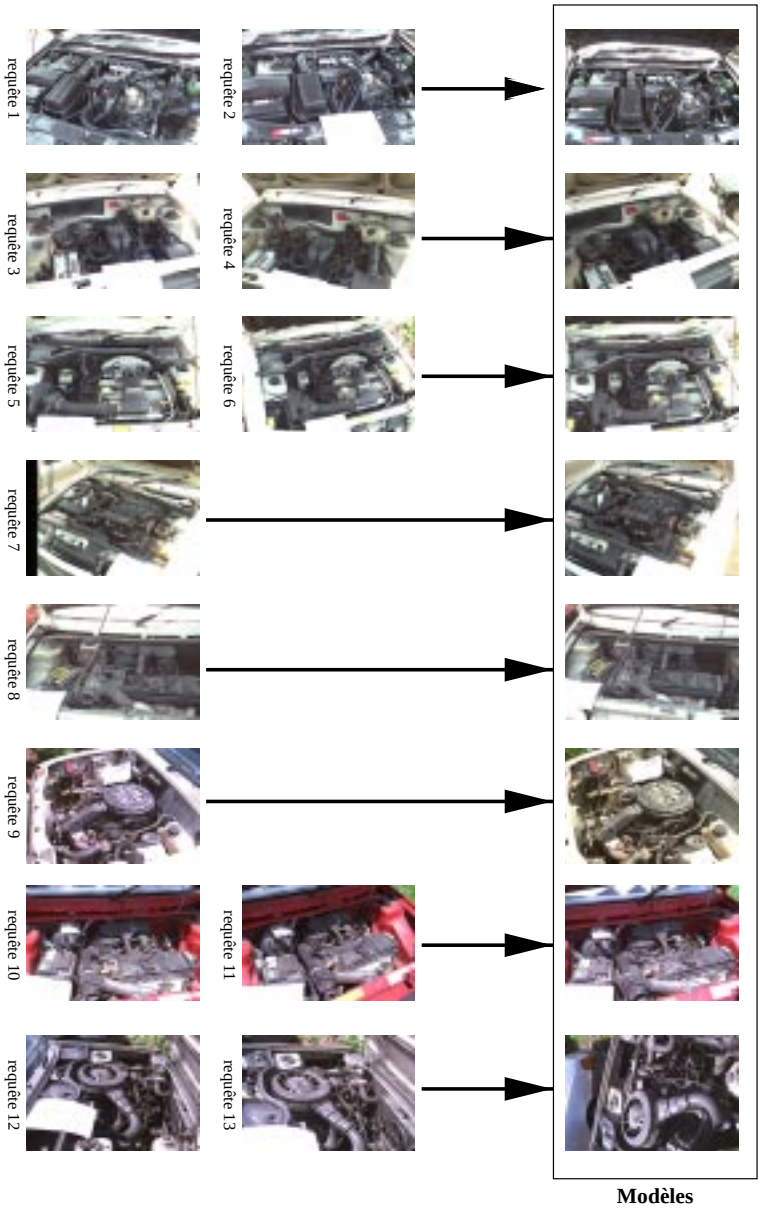

FIG. 4.3: *Reconnaissance de moteurs.*

On suppose que la base contient les invariants de M modèles, dont on été extraits en moyenne D invariants. On procède à une partition de l'espace, l'axe i étant découpé en k_i intervalles. L'espace est donc découpé en $\prod_i k_i$ hypercubes, et un de ces hypercubes contient en moyenne $DM/\prod_i k_i$ invariants.

4.2.1 Cas d'invariants exacts

Dans le cas hypothétique où les invariants seraient calculés de manière exacte, la comparaison des invariants de l'image inconnue, dont on peut supposer qu'ils dont aussi au nombre de D, amène à devoir considérer $D^2M/\prod_i k_i$ paires d'invariants.

À cette complexité, il faut ajouter le coût d'accès aux hypercubes correspondant aux invariants de l'image inconnue. On peut considérer que dans le meilleur des cas ces

hypercube sont accessibles en temps constant, et la complexité est donc proportionnelle à D. Cela donne donc une complexité finale de :

$$D + \frac{D^2 M}{\prod_{i=1}^{n} k_i}$$

Au total, on obtient une complexité linéaire par rapport au nombre de modèles, qui montre que même si l'on a réussi à traiter les images ensemble plutôt que les unes après les autres, la sublinéarité n'est pas atteinte dans ce cas. Il y a une dépendance quadratique vis à vis de la complexité des images, qui est traduite dans la formule par le nombre D d'invariants extraits en moyenne d'une image.

À l'inverse, la augmentation d'un quelconque des nombres k_i, voire l'augmentation de la dimension n des invariants, induit une très forte diminution de la complexité, et ce sans limite.

4.2.2 Cas d'invariants imprécis

Dans le cas d'invariants imprécis, il est nécessaire de visiter plusieurs hypercubes dans chaque dimension pour ne pas rater d'invariants proches. Soit η_i ce nombre pour la dimension i. La complexité de la comparaison est alors $D^2 M / \prod_i (k_i/\eta_i)$.

Comme dans le cas précédent, il faut ajouter le coût de l'accès aux hypercubes, en tenant compte du fait que pour chaque invariant de l'image inconnue, ce sont maintenant $\prod_i \eta_i$ hypercubes qui doivent être visités. La complexité totale est donc :

$$D \prod_{i=1}^{n} \eta_i + \frac{D^2 M}{\prod_{i=1}^{n} \frac{k_i}{\eta_i}}$$

Dans ce cas, la linéarité en M ne s'exerce plus que sur un des deux termes, qui n'est pas le terme dominant quand n est grand. Dans ce cas précis, on peut dire que la sublinéarité est atteinte. La dépendance vis à vis de D reste forte, linéaire dans un terme, quadratique dans l'autre.

Le plus intéressant est d'étudier l'influence de la dimension des invariants. On peut considérer que k_i et η_i sont imposés par les invariants que l'on manipule et la stabilité de leur calcul. On cherche alors à déterminer l'influence de n.

Pour de faibles valeurs de n, le comportement est identique à celui du cas des invariants exacts. Mais pour de grandes valeurs de n, le premier terme se met à croître de manière exponentielle.

Nous avons alors cherché à définir quel était le minimum de la complexité. Pour cela, nous avons simplifié le problème en prenant $k_i = k$ et $\eta_i = \eta$. En notant $x = \ln k / \ln \eta$, on trouve une complexité minimale proportionnelle à :

$$\left(e^{\frac{\ln(x-1)}{x}} \frac{x}{x-1} \right) D^{\frac{x+1}{x}} M^{\frac{1}{x}}$$

Pour un n optimal, l'algorithme est donc sublinéaire en M et subquadratique en D. Le facteur multiplicatif est majoré par 2. Par contre, il n'existe pas de n optimal indépendamment des autres paramètres du problème. Dans les ordres de grandeur de D habituels dans les images, on obtient de bons résultats de complexité pour des n compris entre 4 et 9, l'optimum se situant vers 6 ou 7.

Comparaison avec la recherche séquentielle. Il est intéressant de comparer cette complexité avec celle de la recherche séquentielle exhaustive. La complexité de cette dernière est $D^2 M$. Elle est plus faible dès que $n \geqslant \log_\eta(D^2 M)$. Pour $\eta = 2$, $M = 1000$ et $D = 300$, cela donne $n \geqslant 19$, ce qui rejoint les résultats données dans [BKK96, BW97].

4.2.3 Réduction de la dimension des invariants de couleur

La dimension des invariants utilisés a donc une importance déterminante sur la complexité de l'indexation et de la reconnaissance. Vu qu'il existe de nombreux cas où cette dimension est importante, 23 ou 24 dans le cas des invariants de texture en couleur, beaucoup plus dans le cas d'histogrammes de couleur, nous avons étudié l'application de deux techniques pour réduire la dimension de ces données : l'analyse en composantes principales et l'analyse en composantes curvilignes [DH97].

L'idée commune de ces deux techniques est de trouver un hyperplan ou une hypersurface de plus petite dimension sur lesquels projeter les données et qui conservent au mieux la dispersion des données.

Dans le cas des invariants de texture en couleur, nous avons ainsi montré [Dru98] qu'une ACP permet de réduire à 9 la dimension des invariants tout en gardant plus de 70 % de l'inertie des données. Il semble, par contre, que l'utilisation de l'analyse curviligne n'apporte pas d'avantage significatif qui justifierait le surcoût de son utilisation.

Cette voie paraît intéressante, mais nécessite encore un peu de mise au point. Sur quelles données faut-il calculer l'ACP ? Lorsque la base augmente, est-il possible de mettre à jour cette analyse ? D'autre part, les performances des invariants réduits n'ont pas encore été évaluées sur des données de grande taille, et leur performance n'a pas été comparée à celle des invariants originaux.

4.2.4 Conclusion

L'évaluation de la complexité telle que menée ici admet quelques limites évidentes. La première est que les invariants sont loin d'être distribués uniformément dans leur espace. Les études menées sur deux des composantes des invariants de texture pour les images en niveaux de gris montrent un très fort pic dans la distribution [MPS97].

Le nombre d'invariants par modèle n'est pas constant, et les invariants de l'image inconnue ne sont distribués de manière uniforme non plus. Il y a là de nombreux détails qui affaiblissent l'étude menée, même si cela ne remet pas en cause sa validité générale, mais qui peuvent permettre des améliorations : on pourrait ainsi regrouper les invariants de l'image inconnue qui doivent visiter les mêmes hypercubes, afin de ne pas dupliquer le travail. D'autres pistes d'amélioration sont sûrement possibles.

4.3 Indexation en mémoire secondaire

La nécessité de pouvoir stocker images et invariants en mémoire secondaire ne fait pas doute pour tout système réaliste. Cette indexation n'est pourtant pas chose acquise, les SGBD actuels ne sachant pas indexer des vecteurs imprécis de grande dimension. Leur spécialité est plutôt l'indexation et la recherche de données exactes.

Il est aussi nécessaire d'étudier la complexité en place mémoire des schémas d'indexation proposés, afin que l'indexation d'un grand nombre d'images (plus de 100 000) soit possible sans devoir recourir à des systèmes de stockage hors du commun.

Notre travail dans ce domaine ne fait que commencer. Nous ne présentons donc que quelques pistes dans ce paragraphe. Principalement, nous avons testé quelques schémas de base de données que nous avons testé sur deux SGBD. Ces premiers tests ont surtout à vérifier expérimentalement les résultats de complexité. Suite à cela, nous proposons donc la méthode suivante : chaque hypercube de l'espace des invariants peut être désigné par un code constitué de manière non ambigüe à partir de ses coordonnées. On peut indexer les hypercubes à partir de leur code dans une table de hachage qui assure un accès rapide. Seuls les hypercubes non vides on besoin d'être effectivement présents dans la table.

Pour ne pas avoir à calculer un trop grand nombre de codes, et ne pas parcourir systématiquement tous les voisins d'un hypercube, nous proposons que chaque hypercube contienne un pointeur sur deux de ses voisins par dimension. On restreint ainsi le nombre de pointeurs par hypercube à $2n$ au lieu de 2^n.

Pour parcourir les voisins, on se sert de ces pointeurs (en ne visitant pas deux fois le même voisin !). Le fait de garder un pointeur vide indique qu'il n'y a pas de voisins à regarder d'un côté. Ceci permet de ne pas stocker un pointeur sur tous les voisins d'un hypercube, ce qui réduirait la complexité en temps d'accès, mais aurait un prix trop fort en place mémoire. D'un autre côté, l'utilisation des pointeurs vides permet d'élaguer des branches de recherche et donc de ne pas systématiquement visiter tous les voisins, ce qui est une des causes principales de complexité en grande dimension.

Le prix à payer est qu'il faut parcourir tous les voisins d'un hypercube chaque fois qu'on ajoute un hypercube dans la base, pour pouvoir mettre à jour les pointeurs. Si douloureuse soit-elle, cette opération est la conséquence du rejet de la complexité sur la phase hors-ligne de construction de la base.

Une première expérimentation de ce schéma a débuté en utilisant le SGBD O_2. Une des difficultés de cette mise en œuvre est que l'algorithme d'indexation est imposé par O_2, et qu'il est basé sur les B-tree. Modifier cet algorithme est possible, et ce travail fait l'objet d'une collaboration que nous avons commencé avec l'équipe STORM du laboratoire LSR.

Chapitre 5

Perspectives et applications

I think there is a world market for maybe five computers.
Thomas Watson, chairman of IBM, 1943

Les trois chapitres précédents ont présenté dans les grandes lignes le travail que nous avons effectué. Ce travail répond à un certain nombre de questions, fournit divers algorithmes, mais ouvre aussi de nombreuses pistes de recherche pour le futur. Ce chapitre commence donc par faire quelques propositions de travail pour continuer ce qui a été entrepris, tant dans le champs de l'appariement des images que dans celui de l'indexation. Ensuite, nous présentons deux prototypes en cours de développement, qui ne manquent pas, eux aussi, de poser divers problèmes encore à résoudre.

5.1 Quelques perspectives pour l'appariement des images

Les méthodes présentées dans le deuxième chapitre ont été développées dans un cadre applicatif précis. Sachant qu'il est illusoire de chercher une méthode d'appariement universelle, ces méthodes visent d'abord à obtenir de bons résultats pour des types d'images particuliers et les résultats montrent que le but est généralement atteint.

Deux types d'images échappent à ces méthodes : celui où l'hypothèse d'un mouvement apparent assimilable à une similitude n'est plus valable et celui où les primitives ne sont plus assez fiables. Le premier type comprend, entre autres, les images représentant des objets légèrement différents, ou celui des objets déformables.

Pour traiter ces images, quelques possibilités existent : la méthode basée sur les segments peut être étendue au cas des invariants projectifs. La difficulté est alors que les birapports calculés sont instables et nécessitent une détection bien plus précise des contours. D'autre part, la comparaison des homographies pour trouver le point d'accumulation dans l'espace des transformations n'est pas aisée. Mais avec un peu de soin, en particulier en utilisant des détecteurs paramétriques de coins, cela doit tout de même être réalisable. Cela pourrait permettre d'apparier des images de scènes planes avec de très fortes distorsions projectives.

Avec la méthode basée sur les points d'intérêt, outre les difficultés qui viennent d'être relevées, le principal problème est celui du support de calcul. Pour prendre en compte le facteur d'échelle, il faut mettre en place une stratégie multi-échelle, c'est à dire utiliser comme supports une série de cercles concentriques pour calculer les invariants à plusieurs échelles.

Étendre cela à des transformations affines est possible en prenant pour support une famille d'ellipses centrées sur chaque point d'intérêt. Dans ce cas, c'est donc déjà une famille de dimension 3. Pour passer au projectif, il faut alors considérer toutes les coniques, sachant, par exemple, que les ellipses ne seront plus forcément centrées sur les points d'intérêt : la famille des supports est alors de dimension 5. Cela crée un indéniable problème de complexité tant en espace qu'en temps.

La solution est donc à chercher ailleurs : lorsque deux images sont trop éloignées, il faut utiliser une image intermédiaire.

Pour le deuxième type d'images, celui des images où les primitives ne permettent plus de calculer les invariants présentés précédemment, diverses stratégies sont possibles. Nous avons choisi de faire collaborer nos diverses techniques entre elles. Une telle collaboration est possible à plusieurs niveaux.

1° On peut tout d'abord faire marcher les méthodes séparément pour l'appariement des invariants, puis confronter leurs résultats dans un même espace de représentation des similitudes. Une telle collaboration est possible dès lors que chaque appariement d'invariants permet de calculer une similitude entre les deux configurations de primitives sous-jacentes.

   Le succès peut alors venir d'une accumulation d'évidences dans cet espace plus grande que lors de l'emploi d'une seule technique, les résultats de chacune d'elles se combinant pour la recherche de l'approximation du mouvement apparent.

2° On peut ensuite combiner les méthodes au niveau de leur invariants, en fabriquant des configurations mixtes et en calculant de nouveaux invariants basés sur ces configurations. Ces derniers peuvent montrer plus de robustesse dans certains cas.

   Ainsi, nous avons développé de nouveaux invariants à partir de configurations formées d'un segment et de deux points d'intérêt ou de deux segments et d'un seul point. L'intérêt de ces configurations est de ne pas utiliser les extrémités des segments qui sont de fait considérés comme des droites. Pour ce genre de configurations, il est important d'imposer des critères de restriction pour limiter la combinatoire de l´énumération des configurations. Dans le cas présent, nous avons imposé une distance maximale entre les segments et les points, distance qui est définie en fonction de la longueur des segments pour éviter le problème des changements d'échelle.

Ces deux procédés ont été testés. Par exemple, la FIG. 5.1 montre deux images et le résultat de leur appariement en utilisation des invariants mixtes. Les primitives mises en commun sont en rouge sur la figure.

Ces résultats montrent qu'il est possible de traiter correctement des images où ni la structure seule, car elle ne peut être extraite correctement, ni la texture seule ne peuvent fournir de résultats corrects. Tous les appariements ainsi réalisés ne sont pas corrects, mais il y en a suffisamment pour qu'une méthode robuste puisse détecter les erreurs, et qu'il soit possible ensuite de calculer la géométrie épipolaire et de poursuivre ainsi l'appariement.

Tout n'est pas réglé pour autant, et divers domaines de l'appariement ne sont pas couverts par les techniques et algorithmes qui viennent d'être présentés. Ainsi les objets déformables ou articulés présentent des difficultés particulières qui nécessitent probablement des algorithmes spécifiques.

Nous pensons qu'il est nécessaire de disposer d'une palette d'outils, chacun spécialisé pour un type d'images. Il est alors important que chaque technique sache automatiquement évaluer ses résultats et leur pertinence, pour que seuls les résultats les meilleurs soient retenus. La collaboration entre méthodes paraît aussi une méthode très prometteuse. Nos espaces de transformation semblent être un des outils possibles

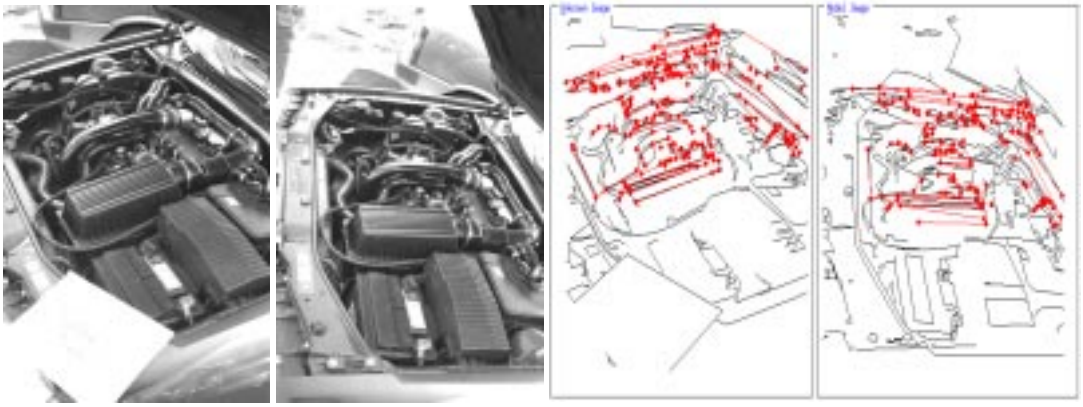

FIG. 5.1: *Deux images de moteurs et leur appariement. Les éléments appariés apparaissent en rouge.*

pour faire collaborer ces méthodes, en leur donnant un moyen commun de confronter leur résultats et d'arriver ensemble à des conclusions qui ne sont à la portée d'aucune d'entre elles.

D'autre part, apparier des images représentant le même type d'objets, mais pas le même objet reste le plus souvent un défi encore à relever. Côté texture, l'emploi de la micro texture n'est sans doute plus possible. La texture fine d'un objet est souvent la caractéristique d'un objet particulier (les empreintes digitales en sont un cas prototypique). Se baser dessus permet de différencier les objets, mais ne permet pas de caractériser l'ensemble des objets. Quelques résultats [Ser96] semblent toutefois montrer que, dans certains cas, les invariants calculés en des points semblables d'objets différents, par exemple au coin externe de l'œil droit de différents visages, sont assez proches. Ces résultats et leur utilisabilité restent à confirmer.

Côté structure, il faut pouvoir prendre en compte la variabilité de ces structures, ce qui n'est pas chose aisée. Nous restons toutefois convaincu que les segments restent un bon moyen de caractériser la structure des objets manufacturés, mais que le problème est de savoir établir des modèles qui sachent décrire la variabilité des structures.

5.2 Quelques perspectives pour l'indexation d'images

Le succès des techniques d'indexation dépend de deux facteurs : la capacité des méthodes d'appariement sous-jacentes, dont on a discuté au paragraphe précédent, et le fait de pouvoir implémenter ces méthodes efficacement en utilisant la mémoire secondaire. Sur ce second point, beaucoup de travail reste à faire.

La plupart des SGBD commerciaux, qu'ils soient relationnels ou à objets, offre des possibilités pour le stockage de données multimédias. Le point non résolu est l'accès à ces données par leur contenu, et non pas seulement par des mots clés ou des descriptions textuelles qui leur soient associées.

Un premier pas est franchi avec l'association de systèmes comme Virage et des SGBD tels Informix. ou Oracle. Le pas suivant sera l'intégration des techniques de reconnaissance d'objets. Le travail à réaliser consiste surtout, d'après nous, à développer une technique d'indexation (au sens des bases de données) pour nos invariants. Ceci doit permettre des tests à plus grande échelle (plus de 10 000 images dans la base) et une validation dans des cas pratiques.

L'autre grand défi, qui rejoint ce qui a été dit pour l'appariement, concerne la modélisation de la variabilité des données. Le seul domaine où quelques résultats ont déjà été publié est celui de la reconnaissance des visages. Mais dans ce cas, on fait généralement des hypothèses sur les images, qui sont généralement cadrées assez précisément pour ne contenir que le visage à reconnaître.

5.3 Deux applications

5.3.1 Recherche d'images fixes

Nous avons commencé à construire un système de recherche d'objets et d'images dans une base d'images. L'idée est de développer un prototype qui permette d'une part de tester nos algorithme en plus grande dimension, d'autre part de se placer en situation plus proche de l'utilisateur, ce qui ouvre de nouvelles pistes de recherche.

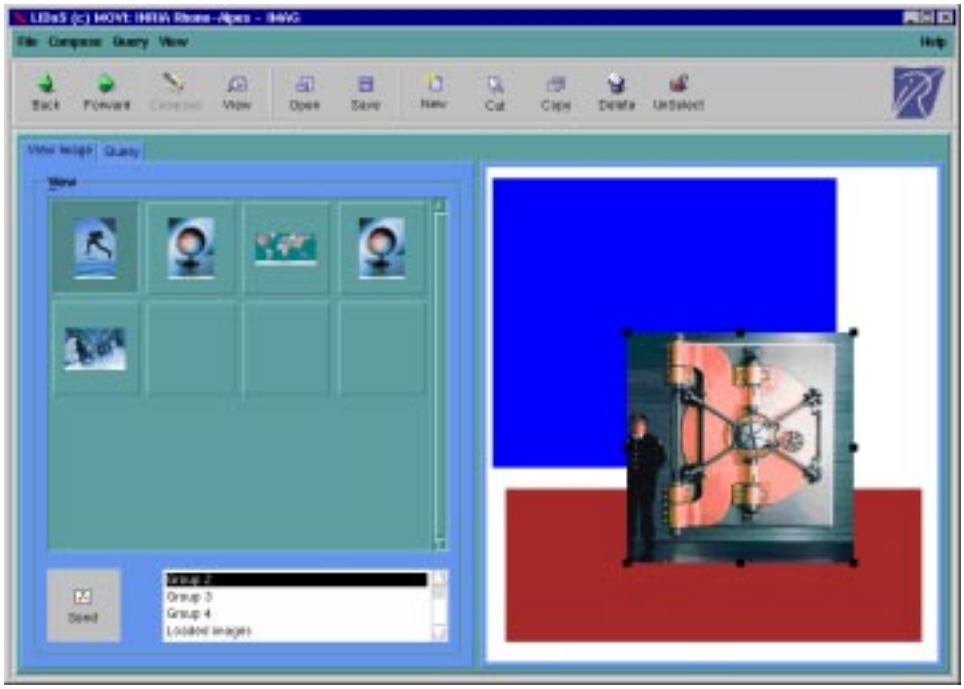

FIG. 5.2: *Interface de requête du système d'indexation d'images fixes.*

L'architecture de ce système est la suivante.

- Une interface (cf. FIG. 5.2) permet de formuler une requête à partir d'éléments de couleur, d'images, ou (c'est encore à venir) de textures. Chacun de ces éléments peut être pris en compte pour sa couleur ou ses invariants.
- Cette requête sera traduite dans un langage de description de requête, et envoyée au SGBD contenant les données.
- Les images sélectionnées par le SGBD sont alors présentées dans un plan, ce qui permet une meilleure perception du résultat (cf. FIG. 5.3).
- Il est possible, à partir de ces premiers résultats, de reformuler la requête.

Les points forts de ce systèmes seront bien sûr la richesse des invariants utilisés, et l'interface. Le fait de pouvoir reformuler les requêtes impose un certain nombre

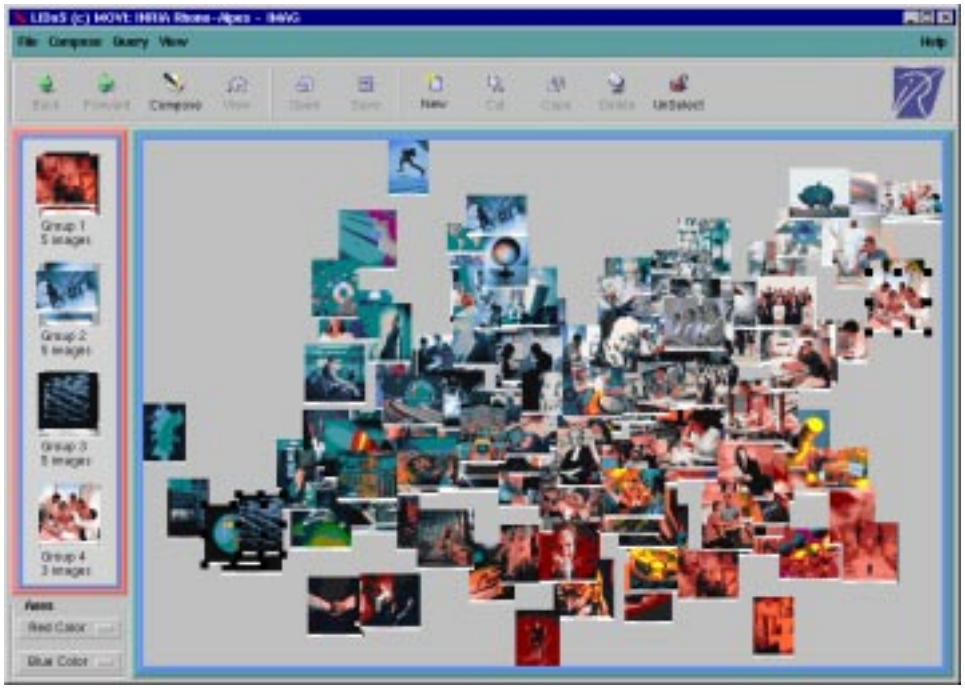

FIG. 5.3: *Interface de présentation des résultats du système d'indexation d'images fixes.*

de contraintes, telles celle de devoir exprimer ces requêtes dans un langage sur lequel il sera possible de raisonner ensuite lors de l'étape de reformulation ou celle d'offrir des possibilités d'interaction graphique supérieures à ce qui est généralement proposé, c'est à dire une présentation linéaire des images par ordre décroissant de score.

5.3.2 Structuration de vidéos

Dans le cadre d'une collaboration avec la société Alcatel, nous participons au développement d'un système de structuration de vidéos. Il comporte un découpage de vidéos en plans, une détection des objets en mouvements, ainsi que la possibilité de désigner des zones d'une image qui sont alors répercutées dans les autres images de la séquence, ainsi que la possibilité de créer des objets, en retrouvant les apparitions d'un même objet dans des plans différents (cf. FIG. 5.4).

Une interface permet alors de jouer des vidéos cliquables, vidéos dans lesquelles certaines zones sont actives et font apparaître de données quand on clique dans l'une d'entre elles.

Le défi principal posé par ce système est celui, déjà mentionné plusieurs fois, de la variabilité des données. Une personne entre dans un scène, se retourne et disparaît. On l'a donc vu de face et de dos. Qu'y a-t-il de commun entre les deux?

L'atout de la vidéo est d'offrir des séquences d'images et donc une redondance très forte des données, ainsi que la possibilité de suivre les objets par *tracking*. Même à partir d'une seule apparition d'un objet, il est donc possible d'utiliser la redondance des diverses vues où il apparaît.

Le nombre de descripteurs calculables est très grand, et il est facile dans quelques cas particuliers de définir quel est le descripteur pertinent pour retrouver un objet : une voiture jaune reste souvent jaune, et peut être le seul objet mobile jaune d'un film. Pas de difficulté donc pour la retrouver de plan en plan. Mais le problème est de

FIG. 5.4: *Les objets retrouvés dans une vidéo.*

FIG. 5.5: *Interface de présentation des vidéos cliquables.*

savoir comment déterminer automatiquement quel est les descripteur à utiliser dans un tel cas.

Les statistiques doivent pouvoir apporter une réponse, mais il faut garder à l'esprit que le problème est difficile car on ne dispose que de quelques images exemples, de beaucoup de contre-exemples, et que le nombre de descripteurs envisageables est très important.

Les applications de ce système sont nombreuses : fabrication de résumés, navigation rapide dans une vidéo offrant plus de possibilités que la seule avance rapide, mise au point de vidéos cliquables, voire, mais cela reste encore lointain, indexation de vidéos dans une médiathèque.

Conclusion

The answer is yes, but what is the question?
Woody Allen

Le développement d'un système d'appariement, de modélisation et de reconnaissance d'objets reste, encore aujourd'hui, un objectif ambitieux. Les systèmes dont des démonstrateurs sont disponibles sur le web ne contiennent souvent qu'une indexation rudimentaire basée sur la couleur, la texture ou des indices globaux extraits des images. On trouve toutefois quelques systèmes proposant des indices plus locaux, tels des indices de forme, mais qui nécessitent le plus souvent une extraction manuelle de ces formes. De tels systèmes sont intéressants pour la catégorisation d'images, mais la reconnaissance d'objet reste hors de leur portée.

Nous avons présenté dans ce mémoire divers éléments d'élaboration d'un système automatique de reconnaissance d'objets. Basée sur la technologie de base qu'est l'appariement des images, nous avons mis au point des algorithmes de modélisation d'images, et de reconnaissance d'images à partir de ces modèles. Deux applications sont en cours de développement pour valider ces algorithmes à plus grande échelle.

Le travail n'est, néanmoins, pas fini. Trois défis nous paraissent particulièrement importants à relever :

- la possibilité de stocker nos données en mémoire secondaire, tout en gardant de bonnes performances ;
- l'amélioration des techniques d'appariement pour traiter des types d'images de plus en plus différents ;
- la possibilité de prendre en compte la variabilité des données.

Le premier des trois est probablement le plus à portée de main. Les deux autres nous ramènent à ce qui est probablement la question de base de toute la vision : comment trouver et extraire les meilleures caractéristiques d'une image? Le travail n' est donc pas fini.

Bibliographie

Toute chose nécessaire est par nature ennuyeuse.
Aristote, Métaphysique, IV.5.

[AF86] N. Ayache and O.D. Faugeras. HYPER: a new approach for the recognition and positioning of 2D objects. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 8(1):44–54, 1986.

[AF87] N. Ayache and B. Faverjon. Efficient registration of stereo images by matching graph descriptions of edge segments. *International Journal of Computer Vision*, 1(2):107–132, April 1987.

[ÅM92] K. Åström and L. Morin. Random cross ratios. Technical Report RT 88 IMAG - 14 LIFIA, LIFIA–IRIMAG, October 1992.

[Ana89] P. Anandan. A computational framework and an algorithm for the measurement of visual motion. *International Journal of Computer Vision*, 2:283–310, 1989.

[Åst93] K. Åström. Fundamental difficulties with projective normalization of planar curves. In *Proceeding of the* DARPA–ESPRIT *workshop on Applications of Invariants in Computer Vision, Azores, Portugal*, pages 377–389, October 1993.

[BA90a] J. Ben-Arie. Probabilistic models of observed features andd aspect graphs. *Pattern Recognition Letters*, June 1990.

[BA90b] J. Ben-Arie. The probabilistic peaking effect of viewed angles and distances with application to 3D object recognition. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 12(8):760–774, August 1990.

[BB81] H.H. Baker and T.O. Binford. Depth from edge- and intensity- based stereo. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, pages 631–636, August 1981.

[Ben75] J.L. Bentley. Multi-dimensional binary search trees in databases applications. *Communications of the* ACM, 18(9):509–517, September 1975.

[Bey92] H.A. Beyer. Accurate calibration of CCD cameras. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Urbana-Champaign, Illinois, USA*, pages 96–101, 1992.

[Bez74] J.C. Bezdek. Cluster validity with fuzzy sets. *Journal of Cybernetics*, 3:58–72, 1974.

[BH86] R.C. Bolles and R. Horaud. 3DPO : A three-dimensional Part Orientation system. *The International Journal of Robotics Research*, 5(3):3–26, 1986.

[Bha87] B. Bhanu. Guest editor's introduction. *Computer (Special Issue on CAD-Based Robot Vision)*, August 1987.

[BJ85] P.J. Besl and R.C. Jain. Three-dimensional object recognition. ACM *Computing Surveys*, 17(1), 1985.

[BKK96] S. Berchtold, D.A. Keim, and H.P. Kriegel. The X-tree: An index structure for high-dimensional data. In *Proceedings of the 22nd VLDB Conference, Mumbai (Bombay), India*, pages 28–39. the Very Large Database Endowment, 1996.

[BL93] T.O. Binford and T.S. Levitt. Quasi-invariants: Theory and exploitation. In *Proceedings of* DARPA *Image Understanding Workshop*, pages 819–829, 1993.

[BM95] B. Boufama and R. Mohr. Epipole and fundamental matrix estimation using the virtual parallax property. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 1030–1036, June 1995.

[Bow91] K. Bowyer. Why aspect graphs are not (yet) practical for computer vision. In *Proceedings of the* IEEE *workshop on Direction on automated* CAD*-based Vision, Maui, Hawaii, USA*, pages 97–104, 1991.

[BW97] S. Blott and R. Weber. A simple vector-approximation file for similarity in high-dimensional vector spaces. Technical Report 19, ESPRIT project HERMES (no. 9141), March 1997. Postscript version available by `ftp` [1].

[Can86] J. Canny. A computational approach to edge detection. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[CM91] C.H. Chen and P.G. Mulgaonkar. CAD-based feature-utility measures for automatic vision programming. In *Direction in Automated CAD-Based Vision*, pages 106–114. IEEE Computer Society Press, 1991.

[CSF86] R.T. Chin, H. Smith, and S.C. Fralik. Three-dimensional object recognition. ACM *Computing Surveys*, 17(1):75–145, 1986.

[Der87] R. Deriche. Using Canny's criteria to derive a recursively implemented optimal edge detector. *International Journal of Computer Vision*, 1(2):167–187, 1987.

[DH97] P. Demartines and J. Herault. Curvilinear component analysis: A self–organizing neural network for nonlinear mapping of data sets. IEEE *Transactions on Neural Networks*, 8(1):148–154, 1997.

[DLPT82] E. Diday, J. Lemaire, J. Pouget, and F. Testu. *Éléments d'analyse de données*. Dunod, 1982.

[Dru98] N. Druga. Indexation par des indices d'images. Rapport de DEA IVR, ENSIMAG, June 1998.

1. http://www-dbs.ethz.ch/~weber/paper/VAFILE.ps.gz

[DS76] E. Diday and J.C. Simon. Clustering analysis. In K.S. Fu, editor, *Communication and Cybernetics*. Springer-Verlag, 1976.

[Dun74] J.C. Dunn. A fuzzy relative of the ISODATA process and its use in detecting compact well separated clusters. *Journal of Cybernetics*, 3:32–57, 1974.

[Fau93] O. Faugeras. *Three-Dimensional Computer Vision - A Geometric Viewpoint*. Artificial intelligence. The MIT Press, Cambridge, MA, USA, Cambridge, MA, 1993.

[FBF+94] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3:231–262, 1994.

[FDF94] G.D. Finlayson, M.S. Drew, and B. Funt. Color constancy: Generalized diagonal transforms suffice. *Journal of the Optical Society of America A*, 11(11):3011–3019, November 1994.

[FF95] B.V. Funt and G.D. Finlayson. Color constant color indexing. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 17(5):522–529, 1995.

[FJ91] P.J. Flynn and A.K. Jain. CAD-based computer vision: from CAD models to relational graphs. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 13(2):114–132, February 1991.

[FMZ90] D. Forsyth, J.L. Mundy, and A. Zisserman. Transformationnal invariance - a primer. In *Proceedings of the British Machine Vision Conference, Oxford, England*, pages 1–6, September 1990.

[FMZR91] D. Forsyth, J.L. Mundy, A. Zisserman, and C. Rothwell. Invariant descriptors for 3D object recognition and pose. In *Proceeding of the* DARPA–ESPRIT *workshop on Applications of Invariants in Computer Vision, Reykjavik, Iceland*, pages 171–208, March 1991.

[For65] E.W. Forgy. Cluster analysis of multivariate data: Efficiency vs. interpretability. *Biometrics*, 21:768–769, 1965.

[FP86] W. Förstner and A. Pertl. Photogrammetric standard methods and digital image matching techniques for high precision surface measurements. In E.S. Gelsema and L.N. Kanal, editors, *Pattern Recognition in Practice II*, pages 57–72. Elsevier Science Publishers B.V., 1986.

[FR67] H.P. Friedman and J. Rubin. Some invariant criteria for grouping data. *Journal of American Statistics Association*, pages 1159–1178, 1967.

[Fua90] P. Fua. A parallel stereo algorithm that produces dense depth maps and preserves image features. *Machine Vision and Applications*, 1990.

[Gen80] D.B. Gennery. *Modelling the Environment of An Exploring Vehicle by Means of Stereo Vision*. Ph.d. thesis, Stanford University, June 1980.

[GHM91] P. Gros, R. Horaud, and R. Mohr. How to recover line drawings from raw data. In *Proceeding of the* DARPA–ESPRIT *workshop on Applications of Invariants in Computer Vision, Reykjavik, Iceland*, pages 259–271, March 1991.

[GMD+97] P. Gros, G. Mclean, R. Delon, R. Mohr, C. Schmid, and G. Mistler. Utilisation de la couleur pour l'appariement et l'indexation d'images. Technical Report 3269, INRIA, September 1997.

[GMGB97] P. Gros, R. Mohr, M. Gelgon, and P. Bouthemy. Indexation de vidéos par le contenu. In *Actes de la conférence CORESA 97*. CNET, March 1997.

[Goa86] C. Goad. Fast 3D model-based vision. In A.P. Pentland, editor, *From Pixels to Predicates*, pages 371–391. Ablex, Norwood, NJ, 1986.

[Gri81] W.E.L. Grimson. A computer implementation of the theory of human stereo vision. *Philosophical Tansactions of the Royal Society of London*, B292(1058):217–253, 1981.

[Gri85] W.E.L. Grimson. Computational experiments with a feature based stereo algorithm. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 7(1):17–34, 1985.

[Gro91] P. Gros. Modélisation d'images par appariement et correction. In *Actes du 8ème Congrès AFCET de Reconnaissance des Formes et Intelligence Artificielle, Lyon – Villeurbanne, France*, volume 3, pages 1335–1344, November 1991.

[Gro92] P. Gros. Modélisation d'images en vision par ordinateur. *Technique et Science Informatique*, 11(4):43–68, 1992.

[Gro93a] P. Gros. Matching and clustering: Two steps towards automatic model generation in computer vision. In *Proceedings of the AAAI Fall Symposium Series: Machine Learning in Computer Vision: What, Why, and How?, Raleigh, North Carolina, USA*, pages 40–44, October 1993.

[Gro93b] P. Gros. *Outils géométriques pour la modélisation et la reconnaissance d'objets polyédriques*. Thèse de doctorat, Institut National Polytechnique de Grenoble, July 1993.

[Gro95] P. Gros. Matching and clustering: Two steps towards object modelling in computer vision. *The International Journal of Robotics Research*, 14(6):633–642, December 1995.

[Gru85] A.W. Gruen. Adaptative least squares correlation: a powerful image matching technique. *S. Afr. Journal of Photogrammetry, Remote Sensing and Cartography*, 14(3):175–187, 1985.

[Hér91] L. Hérault. *Réseaux de neurones récursifs pour l'optimisation combinatoire*. Thèse de doctorat, Institut National Polytechnique de Grenoble, France, 1991.

[Har95] R. Hartley. In defence of the 8-point algorithm. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 1064–1070, June 1995.

[HJ95] D.P. Huttenlocher and E.W. Jaquith. Computing visual correspondence: Incorporating the probability of a false match. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 515–522, 1995.

[HKR93] D.P. Huttenlocher, G.A. Klanderman, and W.J. Rucklidge. Comparing images using the Hausdorff distance. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, September 1993.

[HLO96] D.P. Huttenlocher, R.H. Lilien, and C.F. Olson. Object recognition using subspace methods. In B. Buxton and R. Cipolla, editors, *Proceedings of the 4th European Conference on Computer Vision, Cambridge, England*, pages 536–545. Springer-Verlag, April 1996.

[HR93] D.P. Huttenlocher and W.J. Rucklidge. A multi-resolution technique for comparing images using the Hausdorff distance. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, New York, USA*, pages 705–706, 1993.

[HSV90] R. Horaud, T. Skordas, and F. Veillon. Finding geometric and relational structures in an image. In *Proceedings of the 1st European Conference on Computer Vision, Antibes, France*, Lecture Notes in Computer Science, pages 374–384. Springer-Verlag, April 1990.

[Ike87] K. Ikeuchi. Generating an interpretation tree from a CAD model for 3D object recognition in binpicking tasks. *International Journal of Computer Vision*, pages 145–165, 1987.

[IM98] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *30th Symposium on Theory of Computing*, 1998.

[Jan66] R.C. Jancey. Multidimensional group analysis. *Australian Journal of Botany*, 14:127–130, 1966.

[Kas83] M. Kass. A computational framework for the visual correspondence problem. In *Proceedings of the 8th International Joint Conference on Artificial Intelligence, Karlsruhe, Germany*, pages 1043–1045, August 1983.

[Kas88] M. Kass. Linear image features in stereopsis. *International Journal of Computer Vision*, 1(4):357–368, January 1988.

[Kle74] F. Klein. *Le programme d'Erlangen.* Collection "Discours de la méthode". Gauthier-Villars, Paris, 1974.

[KMM77] R.E. Kelly, P.R.H. McConnel, and S.J. Mildenberger. The Gestalt photomapper. *Photogrammetric Engineering and Remote Sensing*, 43:1407–1417, 1977.

[KR90] L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data. An Introduction to Cluster Analysis.* John Wiley & Sons, Ltd., New York, 1990.

[Lam98] B. Lamiroy. *Reconnaissance et modélisation d'objets* 3D *à l'aide d'invariants projectifs et affines.* Thèse de doctorat, Institut National Polytechnique de Grenoble, July 1998.

[LG97] B. Lamiroy and P. Gros. Object indexing is a complex matter. In *Proceedings of the 10th Scandinavian Conference on Image Analysis, Lappeenranta, Finland*, volume I, pages 277–283, June 1997. Postscript version available by `ftp`[2].

[LM97] Z.D. Lan and R. Mohr. Non-parametric invariants and application to matching. Technical Report 3246, INRIA, September 1997.

[LMF82] L. Lebart, A. Morineau, and J.P. Fenelon. *Traitement des données statistiques : méthodes et programmes.* Dunod, 1982.

2. ftp://ftp.imag.fr/pub/MOVI/publications/Lamiroy_scia97.ps.gz

[LMP95] L. Lebart, A. Morineau, and M. Pinon. *Statistique exploratoire multidimensionnelle*. Dunod, 1995.

[LSW90] Y. Lamdan, J.T. Schwartz, and H.J. Wolfson. Affine invariant model-based object recognition. IEEE *Journal of Robotics and Automation*, 6:578–589, 1990.

[LW88] Y. Lamdan and H.J. Wolfson. Geometric hashing: a general and efficient model-based recognition scheme. In *Proceedings of the 2nd International Conference on Computer Vision, Tampa, Florida, USA*, pages 238–249, 1988.

[Mac67] J. MacQueen. Some methods for classification and analysis of multivariate observations. In L.M. Le Cam and J. Neyman, editors, *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297. University of California Press, 1967.

[MGL+97] R. Mohr, P. Gros, B. Lamiroy, S. Picard, and C. Schmid. Indexation et recherche d'images. In *Actes du 16^e colloque* GRETSI *sur le traitement du signal et des images, Grenoble, France*, September 1997.

[MN85] G. Médioni and R. Nevatia. Segment-based stereo matching. *Computer Vision, Graphics and Image Processing*, 31:2–18, 1985.

[MN93] H. Murase and S.K. Nayar. Learning and recognition of 3D objects from brightness images. In *Proceedings of the AAAI Fall Symposium Series: Machine Learning in Computer Vision: What, Why, and How?, Raleigh, North Carolina, USA*, pages 25–29, October 1993.

[MN96] J.F. Mari and A. Napoli. Aspects de la classification. Technical Report 2909, INRIA, June 1996.

[Mor93] L. Morin. *Quelques contributions des invariants projectifs à la vision par ordinateur*. Thèse de doctorat, Institut National Polytechnique de Grenoble, January 1993.

[MP76] D. Marr and T. Poggio. Cooperative computation of stereo disparity. *Science*, 194:283–287, 1976.

[MP79] D. Marr and T. Poggio. A computational theory of human stereo vision. *Proceedings of the Royal Society of London*, B 204:301–328, 1979.

[MPS97] R. Mohr, S. Picard, and C. Schmid. Bayesian decision versus voting for image retrieval. In *Proceedings of the 7th International Conference on Computer Analysis of Images and Patterns, Kiel, Germany*, pages 376–383, 1997.

[MZ92] J.L. Mundy and A. Zisserman, editors. *Geometric Invariance in Computer Vision*. The MIT Press, Cambridge, MA, USA, 1992.

[MZ93] J.L. Mundy and A. Zisserman, editors. *Proceedings of the Second* ESPRIT - ARPA *Workshop on Applications of Invariance on Computer Vision, Ponta Delagada, Azores, Portugal*, October 1993.

[Nag95] K. Nagao. Recognizing 3D objects using photometric invariant. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 480–487, 1995.

[NB93] S.K. Nayar and R.M. Bolle. Reflectance ratio: A photometric invariant for object recognition. In *Proceedings of the 4th International Conference on Computer Vision, Berlin, Germany*, pages 280–285. IEEE, May 1993.

[NBE+93] W. Niblack, R. Barber, W. Equitz, M. Fickner, E. Glasman, D. Petkovic, and P. Yanker. The QBIC project: Querying images by content using color texture and shape. In *Proceedings of the* SPIE *Conference on Geometric Methods in Computer Vision II, San Diego, California, USA*, February 1993.

[Nis84] H.K. Nishihara. PRISM, a pratical real-time imaging stereo matcher. Technical Report Technical Report A.I. Memo 780, Massachusetts Institute of Technology, 1984.

[NN97] S.A. Nene and S.K. Nayar. A simple algorithm for nearest neighbor search in high dimensions. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 19(9):989–1003, 1997.

[OK85] Y. Ohta and T. Kanade. Stereo by intra and inter-scanline search using dynamic programming. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 7(2):139–154, 1985.

[Ols95] C.F. Olson. Probabilistic indexing for object recognition. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 17(5):518–522, May 1995.

[PdSMS95] J.P. Peloux, G. Reyval de St Michel, and M. Scholl. Evaluation of spatial indices implemented with the dbms o2. *Engineering of Information Systems*, 3(4):517–554, 1995.

[PMF85] S.B. Pollard, J.E.W. Mayhew, and J.P. Frisby. PMF: A stereo correspondence algorithm using a disparity gradient constraint. *Perception*, 14:449–470, 1985.

[Pol85] S.B. Pollard. *Identifying Correspondences in Binocular Stereo*. PhD thesis, University of Sheffield, 1985.

[Poy97] C.A. Poynton. Frequently asked questions about color, 1997.

[PTSE95] D. Papadias, Y. Theodoridis, R. Sellis, and M.J. Egenhofer. Topological relations in the world of minimum bounding rectangles: A study with r-trees. In *Proceedings of the* ACM *Conference on the Modelling of Data (*SIGMOD*)*, May 1995.

[Rip96] B.D. Ripley. *Pattern Recongition and Neural Networks*. Cambridge University Press, 1996.

[Rot93] C.A. Rothwell. Hierarchical object descriptions using invariants. In *Proceeding of the* DARPA–ESPRIT *workshop on Applications of Invariants in Computer Vision, Azores, Portugal*, pages 287–303, October 1993.

[Rot95] C.A. Rothwell. *Object Recognition Through Invariant Indexing*. Oxford Science Publication, 1995.

[SB91a] L. Shapiro and K. Bowyer, editors. IEEE *Workshop on Directions in Automated CAD-Based Vision*, Maui, Hawai, 1991. IEEE Computer Society Press.

[SB91b] M.J. Swain and D.H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[SC96] B. Schiele and J.L. Crowley. Object recognition using multidimensional receptive field histograms. In *Proceedings of the 4th European Conference on Computer Vision, Cambridge, England*, pages 610–619, 1996.

[Ser96] X. Servantie. Recherche d'images utilisant une indexation par le contenu. Rapport de DEA, ENSIMAG, June 1996.

[SH96] D. Slater and G. Healey. The illumination-invariant recognition of 3D objects using color invariants. IEEE *Transactions on Pattern Analysis and Machine Intelligence*, 18(2):206–210, 1996.

[Sko88] T. Skordas. *Mise en correspondance et reconstruction stéréo utilisant une description structurelle des images.* Thèse de doctorat, Institut National Polytechnique de Grenoble, France, 1988.

[SLH87] M.O. Shneier, R. Lumia, and M. Herman. Prediction based vision for robot control. *Computer*, 20(8):46–55, August 1987.

[SLH90] G.L. Scott and H.C. Longuet-Higgins. Feature grouping by "relocalisation" of eigenvectors of the proximity matrix. In *Proceedings of the British Machine Vision Conference, Oxford, England*, pages 103–108, September 1990.

[SM] G. Salton and M.J. McGill. Introduction to modern information retrieval.

[SP95] I. Shimshoni and J. Ponce. Probabilistic 3D object recognition. In *Proceedings of the 5th International Conference on Computer Vision, Cambridge, Massachusetts, USA*, pages 488–493, 1995.

[SS94] M. Stricker and M. Swain. The capacity of color histogram indexing. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Seattle, Washington, USA*, 1994.

[TGHI97] Y. Takeuchi, P. Gros, M. Hebert, and K. Ikeuchi. Visual learning for landmark recognition. In *Proceedings of* DARPA *Image Understanding Workshop, New Orleans, Louisiana, USA*, pages 1467–1474. Morgan Kaufman, May 1997.

[TP91] M. Turk and A. Pentland. Face recognition using eigenfaces. In *Proceedings of the Conference on Computer Vision and Pattern Recognition, Maui, Hawaii, USA*, pages 586–591, 1991.

[Vin69] H. Vinod. Integer programming and the theory of grouping. *Journal of American Statistics Association*, 64:506–517, 1969.

[Wol90] H.J. Wolfson. Model-based object recognition by geometric hashing. In O. Faugeras, editor, *Proceedings of the 1st European Conference on Computer Vision, Antibes, France*, pages 526–536. Springer-Verlag, April 1990.

[Yia93] P.N. Yianilos. Data structures ans algorithms for nearest nieghbor search in general metric spaces. In *Proceedings of the 4th annual* ACM-SIAM *Symposium on Discrete Algorithms, Austin, Texas, USA*, pages 311–321, January 1993.

[ZSB91] S. Zhang, G.D. Sullivan, and K.D. Baker. Relational model construction and 3D object recognition from single 2D monochromatic image. In *Proceedings of the British Machine Vision Conference, Glasgow, Scotland*, pages 240–248, September 1991.

[ZW94] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondance. In *Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden*, pages 151–158. Springer-Verlag, May 1994.

Résumé

Nous présentons dans ce document l'ensemble de nos travaux concernant l'appariement, la modélisation et l'indexation des images. Cet ensemble de techniques concourt au développement d'un système de reconnaissance automatique d'images.

Dans un premier temps, nous présentons diverses méthodes d'appariement d'images adaptées spécifiquement aux images structurées, texturées en niveaux de gris ou en couleur. Nous montrons comment faire coopérer ces méthodes dans le cas d'images difficiles.

La deuxième partie est consacrée à la modélisation d'images et de concepts visuels, modélisation qui repose sur une technique de regroupement hiérarchique. La taille des groupes formés est calculée par une méthode basée sur l'entropie.

L'indexation des images occupe la fin du mémoire. Deux cas sont étudiés : celui du stockage des données en mémoire vive, cas pour lequel nous fournissons des résultats de complexité, et celui du stockage en mémoire auxiliaire, qui reste encore largement à explorer.

Le tout est largement illustré de cas concrets et ouvre de nombreuses pistes de travail.

Mots clés : vision par ordinateur, mise en correspondance, appariement, modélisation, invariants, indexation, reconnaissance.

Abstract

This document present all our work concerning image matching, modeling and indexing. This set of techniques was developed to build a system of automatic image recognition.

In the first part, several matching techniques are presented. Each of these techniques is specifically devoted to a kind of images, respectively structured, gray level textured and color textured images. It is possible to make all these techniques collaborate to treat difficult images.

The second part is devoted to the problem of modeling images and visual concepts. The basic tool is a hierarchical clustering algorithm, which uses an entropy based method to determine the number of cluster that should be formed.

Image indexing is the subject of the last part. Two cases are studied: in the first one, the images are stored in the RAM memory of the computer, and several results of complexity are provided, and in the second case, the images are stored in the auxiliary memory of the computer.

All the techniques presented are illustrated by examples using real images.

Keywords : computer vision, image matching, modeling, invariants, indexing, recognition.