05
M/004.441

République Algérienne Démocratique et Populaire

Ministère de l'enseignement supérieur et de la recherche scientifique

Université de Guelma

Faculté des Mathématiques, d'Informatique et des Sciences de la matière

Mémoire de fin d'étude Master LMD

Département d'Informatique

Spécialité : Ingénierie des médias

13/829

Titre : Comparaison des coefficients de corrélation pour une reconstruction 3D.

Présenté par : Bilal Djaghout

Amir Benteboula

Sous la direction de :

Mr Kouahla Nadjib

Juin 2013

Remerciement

Je tiens à remercier au premier lieu mes parents, ma famille et mes amis pour tout soutien qui m'a toujours manifestés. Un grand merci pour notre encadreur Dr. Mohammed Nadjib Kouahla pour sa patience, son encouragement et sa disponibilité.

Résumé

Ce mémoire se situé dans le cadre de la vision stéréoscopique et concerne plus précisément les méthodes de mise en correspondance stéréoscopique inspiré de la capacité binoculaire de l'être humain. Cette étape consiste de trouver les pixels homologues dans deux images d'une même scène, prises de deux points de vue différents. Une des manières pour réaliser la mise en correspondance est de faire appel à des mesures de corrélation.

Notre travail est basé sur une observation des cirrus faite par deux compagnes en France en 2009 et 2010. Ces cirrus sont naturel comme artificiel créer par les avions. Ces derniers sont considérés parmi les causes majeures de changement climatique. Les deux sites d'observation se situent en Marnay (47°17'31.5" N, 5°44'58.8" E; altitude 275 m) et en Saint-Thiébaud (46°58'31.5" N, 5°52'22.7" E; altitude 600 m). La distance entre ses deux sites est 36 Km. Notre système est conçu a base multiagent. On affecte le processus de traitement d'image qui compris la conversion d'image de RGB vers niveau de gris, une amélioration de contraste et une inversion de perspective pour obtenir une vue satellitaire a deux agents séparé, chaque agent traite une image. Un agent principal applique la triangulation sur une zone qui est une zone commune dans les deux champs de vue. Enfin, on affiche le résultat de la mise en correspondance en 3D.

Mot-clé : stéréovision, mise en correspondance stéréoscopique, reconstruction 3D, traitement d'image, système multiagent, plateforme JADE.

Table de matière

Abréviation

NCC	Normalized Cross Correlation
ZNNC	Zero mean Normalized Cross Correlation
MOR	Moravec
SAD	Sum of Absolute Differences
SSD	Sum of Squared Differences
NSSD	Normalized Sum of Squared Differences
ZSSD	Zero mean Sum of Squared Differences
ZNSSD	Zero mean Normalized Sum of Squared Differences

Liste des figures

Liste des tables

Introduction générale

De tout temps, l'homme a tenté d'élaborer des machines capables de reproduire et de faciliter son travail. L'évolution des techniques et des connaissances dans de nombreux domaines, comme la médecine, la physique, les mathématiques et plus récemment l'informatique, a permis d'inventer toutes sortes d'outils, du plus rudimentaire au robot le plus perfectionnée. Les progrès de la médecine ont permis de mieux comprendre le système visuel humain et, depuis quelques décennies, l'homme cherche à reproduire par une machine les mécanismes du système visuel humain pour des applications aussi variées que l'aide au déplacement d'un robot, l'exploration et l'étude des organes humains en médecine, la surveillance vidéo ou l'étude comportementale des animaux. L'être humain s'inspire de son système visuel et conçoit une technique qui s'appelle stéréovision, qui permit de récupéré l'information de profondeur perdu dans la phase de photographie, à l'aide de la mise en correspondance stéréoscopique.

Les méthodes de mise en correspondance par corrélation reposent sur la ressemblance de deux ensembles de niveaux de gris qui est quantifiée grâce à une mesure de corrélation. De très nombreuses mesures de corrélation ont été proposées pour prendre en compte les différentes difficultés de la mise en correspondance, comme les bruits ou les occultations. Il s'avère donc très difficile de choisir une mesure de corrélation.

Ce mémoire s'articule autour de quatre chapitres.

Chapitre 1 - état de l'art sur la mise en correspondance stéréoscopique de pixels. On parle aussi de la modélisation de caméra plus particulièrement en parle du modèle sténopé et ses paramètres intrinsèque et extrinsèque. Puis, on cite les familles des coefficients de corrélation ainsi qu'on cite quelque exemple utilisé dans notre application.

Chapitre 2 – Cirrus naturel et artificiel : nous parlerons de cirrus naturel et artificiel, sa composition, ses type et enfin les trainé d'avion. C'est autour de ce dernier que notre application se déroule.

Chapitre 3 – Système multiagent et plateforme JADE : dans ce chapitre on fait une introduction au monde SMA, puis on parle de la plateforme utilisé pour implémenté notre système.

Chapitre 4 – Conception et résultat : dans ce dernier chapitre nous allons expliquer les concepts utilisés pour bâtir notre application, et nous allons présenter les différents résultats retenus

Chapitre 1

La stéréovision

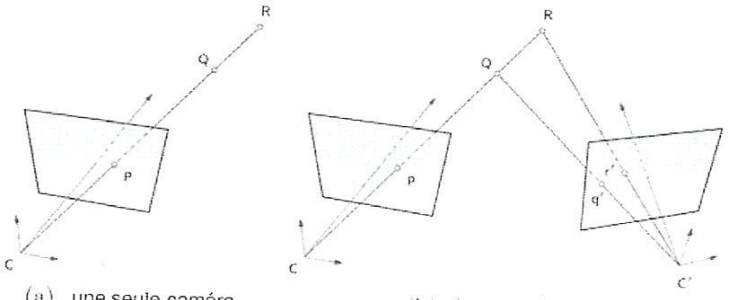
1. Introduction

Chez l'être humain, la vision binoculaire est un élément essentiel de la perception tridimensionnelle : chaque œil est un capteur qui fournit au cerveau sa propre image de la scène ; la connaissance simultanée de ces deux images lui permet de reconstituer la profondeur de la scène. Ce processus ne pose aucune difficulté pour l'être humain, mais constitue, en revanche, un enjeu majeur en vision par ordinateur. Une étape cruciale dans la résolution de ce problème est la mise en correspondance stéréoscopique, qui consiste à retrouver dans les images gauche et droite, les pixels homologues qui sont les projections d'un même point physique de la scène. La différence entre les coordonnées de pixels homologues est appelée disparité. Le problème de mise en correspondance se ramène ainsi à un problème d'estimation d'un champ de disparité. L'objectif de la stéréovision est de calculer les coordonnées 3D de chaque point de la scène à partir des coordonnées de leur projection sur les deux images fournies par deux caméras. L'étape essentielle des algorithmes de stéréovision est l'appariement de certaines primitives entre les deux images. Ces primitives, extraites des images, peuvent être des points de contour, des segments ou des sous-ensembles de pixels. L'appariement consiste alors à trouver, pour chaque primitive de l'une des deux images, son correspondant dans l'autre image.

2. Système stéréoscopique

Le principe de base de la stéréovision consiste à utiliser deux photographies d'un même objet obtenues depuis deux sites de vue différents afin d'apprécier la distance en profondeur des points de l'objet. Les images d'un objet formées par chacun des deux yeux d'un observateur constituent un couple stéréoscopique qui permet au cerveau de construire une image tridimensionnelle. Dans un premier temps, les deux informations 2D concernant le même point 3D situé dans le monde 3D doivent être mises en correspondance. Ensuite, la reconstruction tridimensionnelle est obtenue par un calcul géométrique.

En terme géométrique, une caméra est un transformateur 3D vers 2D, c.-à-d. tout " point visible " de l'espace tridimensionnel en un point dans l'espace bidimensionnel de l'image. Cette transformation supprime donc la profondeur (figure 1.1 a). Les points Q et R de l'espace se projettent tous deux sur le plan image en un seul et même point "p" car ils sont sur la même droite projective (C, p). C est appelé le centre de projection ou centre optique de la caméra. Ceci signifie qu'étant donné un point image "p", il existe une infinité de points Q, R etc… dans l'espace tridimensionnel qui peuvent être projetés en p. [1]



(a) une seule caméra (b) deux caméras

Fig.1.1 : Retrouver la troisième dimension par l'emploi de deux caméras

2.1. Modèle sténopé

Afin de pouvoir effectuer des calculs numériques ou des raisonnements géométriques à partir d'images, nous avons besoin d'un modèle qui décrit comment le monde 3-D se projette sur une image 2-D, qui est projeté par une caméra. Nous allons adopter dans notre travail le modèle sténopé. Car il permet de modéliser fidèlement la plupart des capteurs projectifs et aussi simplifier l'estimation des paramètres du modèle.

Le processus d'acquisition d'une image par une caméra peut être modélisé à l'aide d'un système à projection perspective, aussi appelé sténopé (fig. 1.2). Ce modèle de projection centrale, qui transforme un point de l'espace 3D en un point image 2D, correspond au modèle de caméra le plus couramment utilisé. Il se compose d'un plan rétinien et d'un centre de projection ou centre optique C. Le centre de projection correspond à la position de la caméra et le plan rétinien est le plan de formation des images. On appelle :

- Plan focal : le plan situé en C et parallèle au plan rétinien,
- axe optique : l'axe orthogonal au plan rétinien distance focale f : la distance du centre de projection C au plan rétinien,
- plan focal passant par C.

Ce modèle de caméra crée pour tout objet observé de la scène, une image inversée sur le plan rétinien situé derrière le centre de projection C. Cependant, il est courant de placer, par commodité, le plan image π à la même distance f devant le centre optique de manière à obtenir une image non inversée de la scène. L'axe optique intersecte π en un point c appelée point principal.

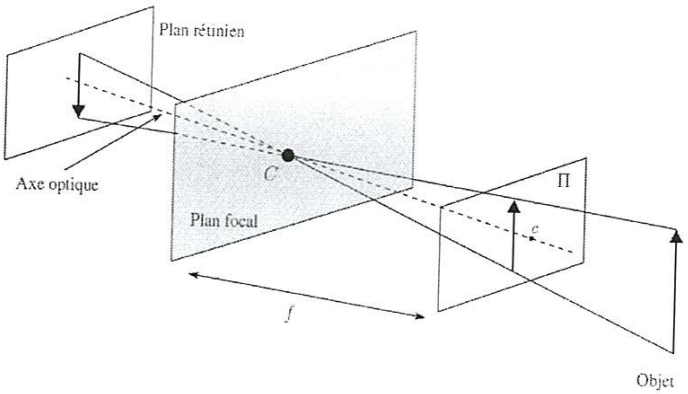


Fig. 1.2 Le modèle sténopé

Tout point M de la scène se projette en un point m sur le plan image par l'intersection de ce dernier avec le rayon optique (CM) (fig. 1.3). Seuls les points situés sur le plan focal n'ont pas de projection sur le plan image. L'expression de cette projection dépend des systèmes de coordonnées choisis dans l'espace et dans le plan image.

2.1.1. Paramètres intrinsèques et extrinsèques de la caméra

Le modèle sténopé permit d'extraire 11 paramètres de la caméra, cinq sont internes et concerne la fabrication de caméra, et six dépendent de la position de la caméra dans l'espace.

Les paramètres intrinsèques sont : k_u, k_v, u_0, v_0, f.

k_u, k_v: Les facteurs d'échelles horizontal et vertical (pixel/mm) ;

u_0, v_0: Les coordonnées de la projection du centre optique ;

f: Distance focal.

La projection d'un point de l'espace 3D en un point image 2D nécessite trois transformations :

- Une projection perspective de l'espace euclidien R^3 dans l'espace euclidien R^2,
- Un changement de repère,
- Une transformation des coordonnées métriques exprimées dans un repère lié à la caméra à des coordonnées pixelliques dans un repère lié à l'image.

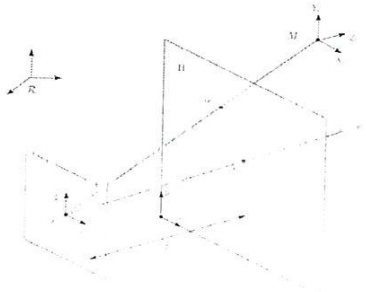


Fig. 1.3 La projection perspective

2.2. La projection perspective

Soit R_c un repère lié à la caméra, d'origine le centre optique C et tel que l'axe des Z est confondu avec l'axe optique. Le repère R_c est appelé système de coordonnées standard de la caméra et est formé des axes horizontal, vertical du plan image et de l'axe optique. Soient X_c, Y_c et Z_c les coordonnées d'un point M de l'espace dans le repère qu'on vient de décrire. Dans ce même repère, l'expression des coordonnées de m, projection de M sur le plan image, résulte de l'application du théorème de Thalès :

$$x_c = f\frac{X_c}{Z_c}$$

$$y_c = f\frac{Y_c}{Z_c} \qquad (1.1)$$

Ces équations ne sont pas linéaires par rapport aux coordonnées euclidiennes X_c, Y_c et Z_c mais elles le deviennent si l'on utilise les coordonnées projectives dans l'espace et dans le plan image.

Si (X, Y, Z) sont les coordonnées euclidiennes d'un point de R^3, ses coordonnées projectives (homogènes) sont définies dans l'espace projectif par $(\lambda.X, \lambda.Y, \lambda.Z, \lambda)$, pour tout réel λ non nul.

On note $M = (X, Y, Z, 1) \approx (\lambda.X, \lambda.Y, \lambda.Z, \lambda) \in P3$, ou $\lambda \neq 0$ arbitraire, les coordonnées projectives d'un point M de coordonnées euclidiennes (X, Y, Z) de R^3 et, de manière similaire, $m = (x, y, 1) \approx (\lambda.x, \lambda.y, \lambda) \in P^2$ celles d'un point m de coordonnées euclidiennes (x, y) dans le plan image.

En adoptant les coordonnées homogènes, la relation (2.1) peut s'écrire sous forme matricielle comme suit :

$$\begin{pmatrix} s.x_c \\ s.y_c \\ s \end{pmatrix} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix} \qquad (1.2)$$

Tous les points appartenant au rayon optique (CM) se projettent en un unique point m sur le plan image. Ils sont, de ce fait, équivalents en coordonnées projectives.

2.2.1. La transformation caméra-image

Soit Ra un repère affine du plan image, d'origine le point o (localisé par exemple au coin de l'image en bas à gauche) et dont les axes sont parallèles à ceux du repère associé à la caméra (fig. 1.3). Les coordonnées des points projetés sur le plan image dans le repère caméra R_c seront désormais exprimées en pixels dans le repère R_c lié à l'image.

La transformation des coordonnées métriques aux coordonnées pixelliques est réalisée par une changement affine de repère

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} k_x & 0 & x_0 \\ 0 & k_y & y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_c \\ y_c \\ 1 \end{pmatrix} \quad (1.3)$$

Ce passage du repère caméra au repère image repose sur la connaissance des paramètres intrinsèques de la caméra :

- (x_0, y_0) sont les coordonnées du point principal c exprimées en pixels dans le repère image,
- k_u et k_v sont respectivement les facteurs d'échelles horizontale et verticale exprimes en pixels/mm.

Ces paramètres sont qualifiées d'intrinsèques car ils ne dépendent que des caractéristiques de la caméra.

La relation entre les coordonnées du point M dans le repère caméra et les coordonnées de sa projection m exprimées dans le repère image s'écrit

$$\begin{pmatrix} s.x \\ s.y \\ s \end{pmatrix} = \begin{pmatrix} k_x & 0 & x_0 \\ 0 & k_y & y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix} \quad (1.4)$$

En introduisant $\alpha_x = k_x f$ et $\alpha_y = k_y f$, cette équation est souvent réécrite sous la forme suivante

$$\begin{pmatrix} s.x \\ s.y \\ s \end{pmatrix} = \begin{pmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_x & y_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix} \quad (1.5)$$

2.2.2. Le changement de repère

Soient R un repère orthonormé de l'espace et (X, Y, Z) les coordonnées du point M dans ce même repère. Les coordonnées (X_c, Y_c, Z_c) de ce point exprimées dans le repère R_c sont obtenues en introduisant les paramètres extrinsèques de la caméra qui traduisent le changement de repère de R à R_c. Ce changement de repère est un déplacement dans R^3, que l'on peut décomposer en une rotation R suivie d'une translation t, ainsi

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = R \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + t \quad (1.6)$$

Il s'exprime en coordonnées homogènes de la façon suivante

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix} = \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (1.7)$$

On appelle donc paramètres extrinsèques, l'expression de la position et de l'orientation de la caméra dans le repère de l'espace, c'est-à-dire le vecteur t de taille 3 x 1 et la matrice R de taille 3 x 3. Ils dépendent chacun de trois paramètres.

L'expression des coordonnées du centre optique C dans le système de coordonnées de l'espace R est donnée à partir de la relation (1.6) par

$$0_3 = R_c + t \tag{1.8}$$

Ce qui implique que

$$C = -R^{-1} + t \tag{1.9}$$

2.2.3. La transformation perspective

La transformation perspective est la composition des trois transformations précédentes.

Elle s'écrit sous la forme d'une matrice qui peut se décomposer comme suit

$$P = AP_0 K \tag{1.10}$$

Où

$$A = \begin{pmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_x & y_0 \\ 0 & 0 & 1 \end{pmatrix}, \qquad P_0 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad \text{et} \quad K = \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} \tag{1.11}$$

Les matrices A et K sont respectivement les matrices des paramètres intrinsèques et extrinsèques de la caméra, et P_0 est la matrice dite de projection centrale.

La matrice P peut être réécrite d'une façon plus concise comme suit :

$$P = A(R\ t) \tag{1.12}$$

a. **Définition :** On appelle ainsi matrice de projection perspective P associée à une caméra la matrice de taille 3x4 représentants, en coordonnées projectives, la projection des points de l'espace sur le plan image. On a alors

$$m = PM \tag{1.13}$$

La matrice P est dénie à un coefficient multiplicatif près. Estimer cette matrice constitue le problème de calibration de la caméra.

b. **Proposition :** Les coordonnées du centre optique C exprimées dans le repère de l'espace peuvent être obtenues à partir de la matrice de projection perspective associée en résolvant l'équation suivante :

$$P\begin{pmatrix} C \\ 1 \end{pmatrix} = 0_3 \tag{1.14}$$

c. Démonstration : En remplaçant P par son expression dans (2.12), on obtient

$$P\begin{pmatrix} C \\ 1 \end{pmatrix} = A(R \;\; t)\begin{pmatrix} C \\ 1 \end{pmatrix} = A\,(RC + t) = 0_3 \;\; \text{(d'après (1.8))}$$

Si l'on peut décomposer la matrice P de taille 3 x 4 en une matrice Q de taille 3 x 3 et un vecteur p de taille 3x1, telle que P = (Q p) et si on suppose que la matrice Q est de rang 3, l'équation (2.8) peut être réécrite comme suit:

$$C = -Q^{-1}\,p \qquad (1.15)$$

2.3. La mise en correspondance stéréoscopique

La mise en correspondance est l'étape clé de la reconstruction tridimensionnelle par stéréovision. Le problème de mise en correspondance reste un problème difficile à résoudre, en raison notamment de la présence de changements de luminosité entre les deux images, d'occultations et de surfaces non texturées. Pour pallier ces difficultés, les méthodes de mise en correspondance sont amenées à exploiter toutes les informations disponibles afin de faciliter la recherche et la détermination des correspondants.

Dans ce paragraphe, nous détaillons tout d'abord les difficultés de la mise en correspondance, puis nous présentons les primitives d'images et les contraintes les plus utilisées pour surmonter ces difficultés [2].

2.3.1. Les difficultés de la mise en correspondance

La mise en correspondance consiste à retrouver, à partir de deux images, les couples de points qui se correspondent. Si l'on dispose d'images de niveaux de gris, il parait raisonnable de faire l'hypothèse que les projections d'un même point de la scène ont la même intensité dans les deux images. Cette contrainte physique connue sous le nom de conservation de la luminance, est aujourd'hui utilisée dans la plupart des approches de mise en correspondance [3].

Un autre problème très important provient de la présence d'objets dont la surface est uniforme ou la présence de textures répétitives. En effet, les pixels dans ces régions ne représentent pas des primitives discriminantes puisque plusieurs correspondants potentiels ayant la même valeur de luminance existent dans l'autre image. Les régions homogènes sont donc difficiles à apparier. Elles sont prises en compte et détectées en utilisant le plus souvent le gradient de l'image.

Par ailleurs, une autre difficulté rencontrée par les méthodes de mise en correspondance concerne les occultations. En effet, certains points de la scène peuvent être visibles dans une image de la paire stéréoscopique mais pas dans l'autre. C'est à dire qu'une portion de la scène observée n'est visible que depuis un seul point de vue : il s'agit d'un problème d'occultation. Ce problème est illustré par la figure (1.4) : le point m est visible uniquement depuis la caméra de

gauche. Les occultations représentent une difficulté majeure pour les techniques de mise en correspondance car ces zones d'images n'ont pas de correspondants dans l'autre image et engendrent souvent des ambiguïtés d'appariement [3].

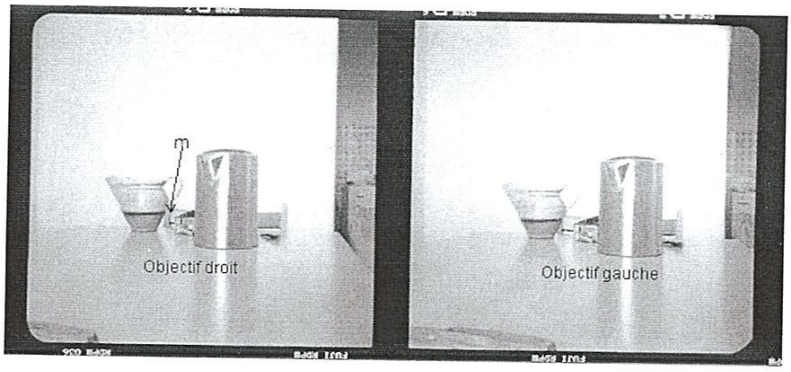


Fig. 1.4 Le problème d'occultation : le point m est visible dans l'image de gauche mais pas dans l'image de droite

2.4. Géométrie épipolaire

Nous nous intéressons maintenant aux relations géométriques qui existent entre les deux images stéréoscopiques d'une même scène. Plus précisément, nous établissons une contrainte géométrique forte entre un point dans une image et son correspondant dans l'autre image. Cette contrainte est illustrée par la figure (1.5).

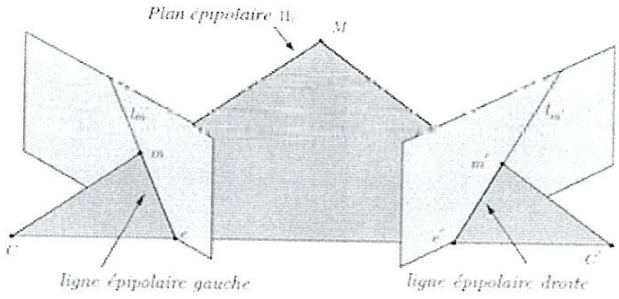


Fig. 1.5 La géométrie épipolaire

Soient C et C' les centres optiques des deux caméras de gauche et de droite. Nous pouvons observer sur cette figure qu'un point M de l'espace, ses projections m et m' dans chacune des deux images et les points C et C' sont coplanaires et définissent un plan πe appelé plan épipolaire.

Supposons à présent qu'on connaît seulement le point m de l'image de gauche et qu'on cherche à déterminer son correspondant dans l'image de droite. Le point m représente la projection dans l'image de gauche de tous les points de l'espace appartenant au rayon (Cm). Ce rayon définit avec la droite (CC'), appelée ligne de base, le plan π_e. Si m' est le correspondant de m dans l'image de droite alors le rayon $(C'm')$ se situe également dans le plan π_e. Le point m'

appartient donc nécessairement à la droite d'intersection de π_e avec l'image de droite, qui n'est autre que la projection du rayon (Cm) dans l'image de droite. Cette droite s'appelle ligne épipolaire associée à m et est notée $l_{m'}$.

La géométrie qui décrit la formation d'une paire d'images stéréoscopiques s'appelle géométrie épipolaire. Elle est essentiellement basée sur les définitions suivantes :

- l'épipole est le point d'intersection de la ligne de base (CC') avec le plan de l'image.
- L'épipole e (resp. e') correspond à la projection perspective dans l'image de gauche (resp. droite) du centre optique C' (resp. C) (Fig. 1.5).
- un plan épipolaire est un plan contenant la ligne de base.
- une ligne épipolaire est l'intersection d'un plan épipolaire avec le plan de l'image. Toutes les lignes épipolaires d'une image se coupent en l'unique point épipole. L'intersection d'un plan épipolaire avec chacune des deux plans image sont deux lignes épipolaires conjuguées.

La géométrie épipolaire fournit la contrainte épipolaire qui s'énonce comme suit.

Propriété Un point m, qui se trouve sur la ligne épipolaire gauche lm doit nécessairement correspondre à un point m' sur sa ligne épipolaire conjuguée $l_{m'}$, et réciproquement.

La contrainte épipolaire nous permet de restreindre le domaine de recherche des correspondants, de l'image entière à la ligne épipolaire. En effet, lorsqu'on cherche pour un point dans l'image de gauche un correspondant dans l'image de droite, on peut se limiter à effectuer cette recherche le long de la ligne épipolaire associée. La contrainte épipolaire est donc utilisée dans tous les travaux de mise en correspondance.

2.5. Définition de la disparité

Nous nous plaçons dans le cas où la mise en correspondance est effectuée de l'image de gauche vers l'image de droite et nous appelons image de référence l'image de gauche. Un point dans l'image de référence peut être associé à un point dans l'image de droite s'ils correspondent au même point physique de la scène : ce sont deux points homologues. La différence entre les coordonnées des pixels homologues est appelée disparité.

Nous pouvons donc assimiler le problème de mise en correspondance à la recherche d'une fonction u qui attribue une disparité à chaque pixel (x; y) de l'image de gauche :

$$u: \quad N^2 \rightarrow R^2$$
$$(x, y) \rightarrow u(x, y) = x - x', y - y'$$

ou (x', y') sont les coordonnées dans l'image de droite du pixel correspondant à (x, y).

Pour visualiser le résultat de la mise en correspondance, nous utilisons dans ce mémoire, comme dans tous les travaux de mise en correspondance, une image appelée carte de disparité.

Chaque pixel de la carte représente l'amplitude de la disparité : plus le pixel est clair et plus la disparité est importante. La figure 1.6 montre un couple d'images, dénommé cônes, et la carte de disparité associée.

(a) (b) (c)

Fig. 1.6 Couple d'images Cônes. (a) Image gauche. (b) Image droite. (c) Carte de disparité

2.5.1. Les primitives à mettre en correspondance

Le choix des primitives, c'est à dire les caractéristiques extraites des images, est déterminant dans le processus de mise en correspondance. Ces primitives peuvent être des pixels, des points d'intérêt, des points de contours, des segments, etc.

- **Les pixels de l'image**

Le pixel est la primitive la plus simple, dite de bas niveau. Sa mise en correspondance permet d'obtenir une carte de profondeur dense. Elle offre la plus grande densité mais c'est aussi la primitive la plus pauvre en information, dont les seuls attributs sont la position et le niveau de gris.

- **Les points d'intérêt**

Ce sont les pixels qui correspondent à des points particuliers dans l'image et qui possèdent des caractéristiques particulières les rendant plus discriminants pour la mise en correspondance. Comme type de pixel particulier, on peut citer les points de contours qui proviennent des changements brusques de niveau de gris, les coins, les jonctions, etc.
De nombreuses méthodes de mise en correspondance s'appuient sur une détection de points d'intérêt. Les détecteurs les plus populaires sont celui de Canny et de Harris et Stephens [3].

- **Les primitives structurées**

Ce sont par exemple les segments, les courbes, les régions, etc. Leurs attributs sont nombreux et peuvent être de types géométriques ou photométriques : la position,

l'orientation, la longueur, le niveau de gris, la surface, etc. Les travaux de mise en correspondance utilisant des segments sont nombreux, on peut citer à titre d'exemple. Cependant, les régions sont des primitives difficiles à utiliser pour la mise en correspondance car, d'une image à l'autre, elles peuvent avoir des formes et des tailles différentes, dues au phénomène de distorsion projective [4].

Les approches de mise en correspondance sont souvent classées en deux grandes catégories suivant les primitives à apparier. Les méthodes basées sur la mise en correspondance de pixels, appelées **pixel-based matching**, sont très sensibles aux occultations et aux changements d'illumination entre les deux vues. Elles sont aussi, souvent, coûteuses en temps de calcul.

Cependant, l'avantage de ces méthodes est qu'elles permettent d'obtenir directement un appariement très dense. Elles concernent de ce fait la plupart des publications sur la mise en correspondance. Toutefois, un certain nombre de publications utilise des primitives éparses, comme les points d'intérêt ou les régions, pour guider la mise en correspondance de pixels. Ces méthodes, appelées **feature-based matching**, nécessitent de détecter au préalable ces primitives qui doivent être caractérisées par des attributs suffisamment invariants et discriminants.

2.6. Système Stéréoscopique utilisé dans notre application

Notre système de vision stéréoscopique est composé de deux caméras réelles situées en deux sites différents et placées dans le mode vis-à-vis. Il est complété par deux caméras virtuelles situées chacune suivant la verticale des sites d'observation à une altitude L qui peut être ajustée selon la hauteur angulaire de l'axe optique des caméras réelles. Généralement, lorsque cette hauteur a pour valeur 28°, nous adoptons $L = 250\ Km$. Les paramètres des caméras virtuelles (distance focale principalement) sont supposés identiques à ceux des caméras réelles. Les deux images prises simultanément aux deux points de vue séparés sont reliées entre elles par la géométrie épipolaire du système stéréoscopique. Lorsque la géométrie est bien définie, le problème de la mise en appariement des deux images de type satellite s'en trouve facilité

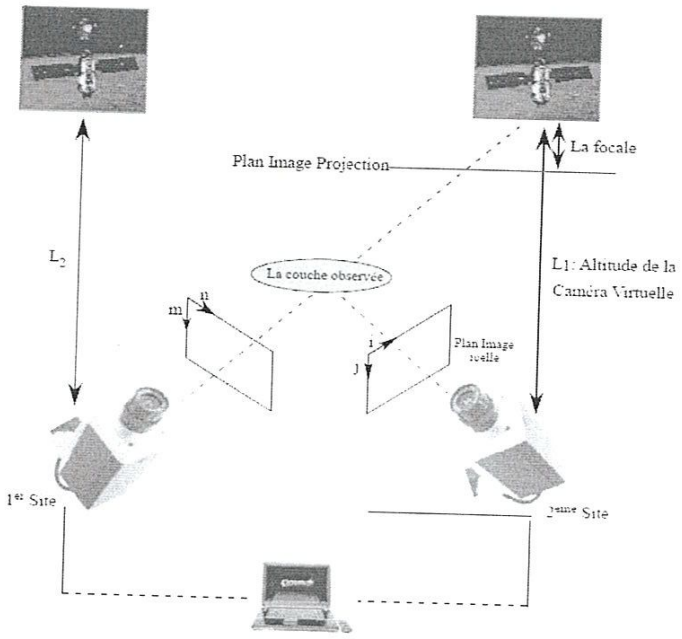


Fig. 1.7 illustration du système stéréoscopique utilisé dans notre application

2.7. Les coefficients de corrélation

Dans la littérature, l'habitude est de distinguer les méthodes locales et des méthodes globales. Une autre façon de procède est de décomposer les méthodes de mise en correspondance en éléments constituants. Cette description permet alors d'affiner la séparation en quatre categories : des méthodes locales, des méthodes globales, des méthodes mixtes (avec critère locale qui intervient dans un processus global) et des méthodes à passage multiple (qui font intervenir dans de manière séquentielle et complémentaire différentes méthodes locales et/ou globales).

Dans notre travail nous sommes intéressés par les méthodes locales. L'hypothèse de ces méthodes est que deux pixels qui ce correspondent, ainsi que leurs voisinages respectifs, se ressemblent d'un point de vue photométrique. De nombreuses difficultés se posent : les changements de luminosité, les zones uniformes, occultations. Ainsi, un très grand nombre de mesures ont été proposées afin de répondre à ces difficultés et nous avons, choisi de les classer en cinq grandes familles :

- Croisé – Il s'agit de toutes les mesures qui utilisent un produit scalaire.
- Classique – Cela concerne toutes les mesures qui s'appuient sur des outils des statistiques classiques.
- Dérivée – Dans cette catégorie, à la place des niveaux de gris, on utilise les dérivées des images.

- Non-paramétrique – Ce type de mesure, comme les mesures dérivées, ne s'appuient plus sur les niveaux de gris, mais, sur l'ordre des niveaux de gris dans le voisinage prise en compte.
- Robuste – afin d'être robuste au problème lié aux occultations, ces mesures font intervenir des outils des statistiques robustes.

2.7.1. Mesures de corrélation croisée : cette famille regroupe trois mesures utilisant le produit scalaire dont une version normalisée, NCC, et deux versions normalisées et centrées, ZNCC et MOR.

$$ZNCC(u,v,d_u,d_v) = \frac{\sum_{k=-p}^{p}\sum_{l=-p}^{p}(I_r(u+k,v+l)-\overline{I_r})(I_l(u+d_u+k,v+d_v+l)-\overline{I_l})}{\sqrt{\sum_{k=-p}^{p}\sum_{l=-p}^{p}(I_r(u+k,v+l)-\overline{I_r})^2}\sqrt{\sum_{k=-p}^{p}\sum_{l=-p}^{p}(I_l(u+du+k,v+l)-\overline{I_l})^2}} \quad (1.16)$$

$$NCC(u,v,d_u,d_v) = \frac{\sum_{k=-p}^{p}\sum_{l=-p}^{p}(I_r(u+k,v+l))(I_l(u+d_u+k,v+d_v+l))}{\sqrt{\sum_{k=-p}^{p}\sum_{l=-p}^{p}(I_r(u+k,v+l)-\overline{I_r})^2}\sqrt{\sum_{k=-p}^{p}\sum_{l=-p}^{p}(I_l(u+du+k,v+l))^2}} \quad (1.17)$$

$$MOR(u,v,d_u,d_v) = \frac{2\sum_{k=-p}^{p}\sum_{l=-p}^{p}(I_r(u+k,v+l)-\overline{I_r})(I_l(u+d_u+k,v+d_v+l)-\overline{I_l})}{\sqrt{\sum_{k=-p}^{p}\sum_{l=-p}^{p}(I_r(u+k,v+l)-\overline{I_r})^2}+\sqrt{\sum_{k=-p}^{p}\sum_{l=-p}^{p}(I_l(u+du+k,v+l)-\overline{I_l})^2}} \quad (1.18)$$

2.7.2. Les statistiques de la distribution des différences des niveaux de gris : cette famille regroupe douze mesures en trois sous-familles: les distances, qui peuvent être normalisées et/ou centrées, les mesures localement centrées, pour lesquelles la moyenne des niveaux de gris de la fenêtre droite est ramenée à la moyenne des niveaux de gris de la fenêtre gauche, et les mesures utilisant la variance, SAD,SSD,NSSD,ZSSD,ZNSSD.

$$SAD(u,v,d_u,d_v) = \sum_{k=-p}^{p}\sum_{l=-p}^{p}\left(I_r(u+k,v+l)\right) - \left(I_l(u+d_u+k,v+d_v+l)\right) \quad (1.19)$$

$$\mathrm{SSD}(u,v,d_u,d_v) = \frac{\sum_{k=-p}^{p}\sum_{l=-p}^{p}(I_r(u+k,v+l)-I_l(u+d_u+k,v+d_v+l))^2}{\sqrt{\sum_{k=-p}^{p}\sum_{l=-p}^{p}(I_r(u+k,v+l)-\overline{I_r})^2}\sqrt{\sum_{k=-p}^{p}\sum_{l=-p}^{p}(I_l(u+du+k,v+l))^2}} \quad (1.20)$$

$$\mathrm{ZSSD}(u,v,d_u,d_v) = \sum_{k=-p}^{p}\sum_{l=-p}^{p}\left((I_r(u+k,v+l)-\overline{I_r})-(I_l(u+d_u+k,v+d_v+l)-\overline{I_l})\right)^2 \quad (1.21)$$

$$\mathrm{ZNSSD}(u,v,d_u,d_v) = \frac{\sum_{k=-p}^{p}\sum_{l=-p}^{p}\left((I_r(u+k,v+l)-\overline{I_r})-(I_l(u+d_u+k,v+d_v+l)-\overline{I_l})\right)^2}{\sqrt{\sum_{k=-p}^{p}\sum_{l=-p}^{p}(I_r(u+k,v+l)-\overline{I_r})^2}\sqrt{\sum_{k=-p}^{p}\sum_{l=-p}^{p}(I_l(u+du+k,v+l)-\overline{I_l})^2}} \quad (1.22)$$

2.8. Reconstruction 3D par triangulation

La reconstruction 3D désigne la technique qui permet d'obtenir une représentation en trois dimensions d'un objet ou d'une scène à partir d'un ensemble d'images prises sous différents points de vue de l'objet ou de la scène.

D'une manière plus générale, le problème appelé reconstruction 3D est le suivant : on dispose d'une ou plusieurs représentations en 2D d'un objet et on souhaite déterminer les coordonnées des éléments visibles sur ces représentations dans un repère de l'espace réel 3D.

Les représentations 2D sont souvent des photographies, il s'agit de retrouver la profondeur des points visibles sur l'image, qui est perdue durant le processus de projection qui produit l'image.

Pour obtenir, comme souhaité, les coordonnées des points de la scène, il faut cependant résoudre un certain nombre de problèmes :

- Problème de calibration ou comment se projettent les points de la scène sur l'image. Pour cela, le modèle de sténopé est utilisé et il est alors nécessaire de connaître des paramètres dits intrinsèques de la caméra (distance focale, centre de l'image...). Ensuite, il est nécessaire de connaître la position relative des caméras pour pouvoir déterminer les coordonnées des points de l'espace dans un repère de la scène non lié à la caméra. Ces paramètres, dits extrinsèques, sont la position et l'orientation de la caméra dans l'espace.
- Problème d'association (matching): il faut être capable de reconnaître et d'associer les points qui apparaissent sur plusieurs photos.
- Problème de reconstruction : il s'agit de déterminer les coordonnées 3D des points à partir des associations faites et des paramètres de calibration.

2.9. Conclusion

On a présenté dans ce chapitre le système stéréoscopique et le fameux modèle sténopé qui représente les paramètres intrinsèque et extrinsèque d'une caméra, en suite on a parlé de la mise en correspondance stéréoscopique et ses difficultés. On a parlé aussi de la définition de la disparité et de la géométrie épipolaire. En fin on a parlé de la reconstruction 3D qui utilise les informations de la carte de disparité et les paramètres de la caméra pour avoir la forme 3D de notre objet étudié.

Le prochain chapitre intitulé cirrus artificiel et naturel parle des différents types de nuage ainsi que les cirrus artificiel.

Chapitre 2

Les cirrus naturel et artificiel

1. Introduction

La vapeur d'eau se dégage à partir des êtres vivants (en particulier des végétaux), de la surface des océans et de la croûte terrestre. Cette vapeur, constituée de très fines gouttelettes d'eau est plus légère que l'air. Cette légèreté est due aux différences de densité elles-mêmes dues aux différences de température. On a donc ascension de fines gouttelettes vers les diverses couches atmosphériques. Ces gouttelettes vont former des amas, les nuages.

Au contact les unes des autres, elles vont gagner en volume et donc en masse. Au-dessus d'un certain seuil : la masse critique, elles seront trop lourdes pour rester en suspension dans l'air et tomberont à pic. Dans leur chute, elles absorberont d'autres gouttes et à l'arrivée vous aurez droit à un beau crachin mais pas à du nuage.

2. Les cirrus

2.1. Définition

Les cirrus sont un genre de nuage appartenant à l'étage supérieur de la troposphère, ils sont situent selon la latitude entre 6 000 et 13 000 m d'altitude avec une épaisseur pouvant varier entre environ 300 m et quelques kilomètres. Ils sont constitués de bancs, de bandes ou de filaments séparés, blancs le plus souvent, qui revêtent un aspect fibreux ou un éclat soyeux [5]. (Figure 3.1)

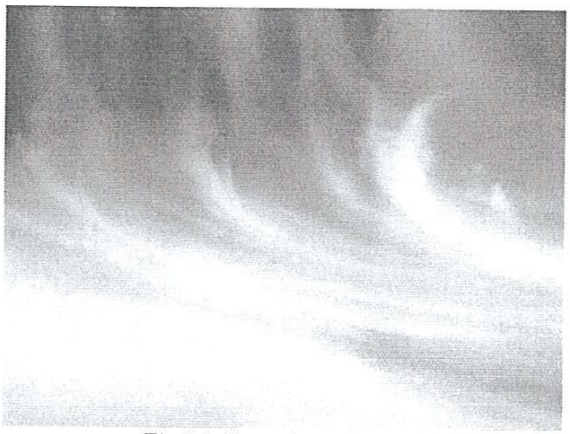


Fig.2.1. Exemple de cirrus

2.2 De quoi sont composés les Cirrus ?

Les cirrus sont des nuages peu épais, à structure filamenteuse, composés de petits cristaux de glace en forme de bâtonnets cylindriques de section principale hexagonale régulière. Les plus petits de ces cristaux (par exemple de taille inférieure à 20 micromètres) sont le siège d'un

mouvement erratique provoqué par le choc des molécules d'air sur eux ; de la sorte, ils ont toutes les orientations possibles dans l'espace [5].

2.3 Contexte météorologique

Dans la terminologie la plus courante, les nuages sont séparés en trois grandes classes :

- les stratus. Ils se présentent sous forme de couche très étalée et homogène, souvent de couleur grise. Ces nuages sont en général non précipitants.
- les cumulus. Ils se présentent comme des nuages fragmentés, détachés et denses, de forme souvent torturée.
- Ils peuvent être précipitants.
- les cirrus. Ce sont des nuages de haute altitude et par conséquent très froids. Leur aspect soyeux provient de leur composition : ils sont formés en majorité de cristaux de glace. Ils sont considérés comme non précipitants.

Afin de décrire la grande diversité de cas pouvant être observés en pratique, ces termes peuvent être combinés, par exemple stratocumulus, cirrostratus... Par ailleurs, lorsque le nuage décrit est précipitant (i.e. lorsqu'il est à l'origine de précipitations visibles depuis le sol), la racine nimbus est accolée au substantif, par exemple cumulonimbus, nimbostratus...

De façon plus objective, les nuages peuvent être classifiés par des paramètres mesurables expérimentalement.

Deux paramètres sont particulièrement significatifs :

- L'altitude. Il est ainsi courant de parler de nuage bas (nuage de basse altitude), de nuage moyen et de nuage haut (nuage de haute altitude). Les altitudes moyennes de ces nuages suivant la latitude sont indiquées dans le tableau 3.1. L'altitude des nuages bas reste constante, la latitude important peu, alors que l'altitude des nuages hauts varie énormément : ils suivent l'évolution de la tropopause, qui elle-même varie en fonction de l'intensité de la convection.
- La composition. Un nuage peut être composé de gouttelettes d'eau (phase liquide), de cristaux de glace (phase glace) ou d'un mélange des deux. La quantité d'eau liquide dans les nuages d'eau devient rapidement négligeable pour des températures inférieures à -20°C, alors que la quantité de glace dans les nuages mixtes sature à 100% pour des températures inférieures à -20°C. Cependant, la température doit être inférieure à -40°C pour assurer l'absence de phase liquide. Cependant, des gouttelettes d'eau surfondues ont été détectées au sommet de cumulonimbus, pour des températures descendant jusque -37°C. Ces mesures restent tout de même relativement rares et font figure d'exception.

Remarquons que ces deux paramètres, altitude et composition, sont corrélés : un nuage de basse altitude aura une température relativement chaude, et présentera donc davantage de chance d'y trouver des gouttelettes d'eau [6].

2.4 Classification

Famille	Genres (nom et symbole)	Propriétés caractéristiques		
Nuages supérieurs	Cirrus, Ci	non étalé en nappe, ni lenticulaire; structure fibreuse ou filamenteuse		Nuage totalement glacé; altitude de la base supérieure à 3 km
	Cirrocumulus, Cc	étalé en nappe ou lenticulaire	nappe fractionnée ou très accidenté, ou nuage lenticulaire	
	Cirrostratus, Cs		nappe continue et peu accidenté	
Nuages Moyens	Altocumulus, Ac	étalé en nappe ou lenticulaire; altitude du sommet supérieure à 2 km	nappe fractionnée ou très accidenté, ou nuage lenticulaire	
	Altostratus, As		nappe continue et peu accidenté	
	Nimbostratus, Ns			
Nuages Inférieurs	Stratocumulus, Sc	étalé en nappe ou lenticulaire; altitude du sommet inférieure à 2 km	nappe aux contours arrondis, fractionnée ou très accidentée ou nuage lenticulaire	Nuage non totalement glacé ou dont l'altitude de la base est inférieure à 2 km
	Stratus, St		nappe continue et peu accidentée, ou aux contours non arrondis	
Nuages à développement vertical	Cumulus, Cu	non étalé en nappe, ni lenticulaire	nuage totalement liquide	
	Cumulonimbus, Cb		nuage glacé à sa partie supérieure	

Tab.2.1. Classification internationale du nuage

2.5 Formation des cirrus

Le mode de formation des cirrus revêt une importance fondamentale. En effet, leurs propriétés microphysiques, dont dépendent fortement leurs propriétés optiques et radiatives, évoluent très lentement avec le temps, et sont donc fortement déterminées lors de la formation.

La condition principale nécessaire à la formation d'un cirrus est la quantité d'eau présente dans l'atmosphère.

En effet, si cette quantité devient suffisamment importante, il y a sursaturation de l'air par rapport à l'eau ou à la glace, ce qui mène alors respectivement à la formation de gouttes d'eau ou de cristaux de glace. Cette sursaturation peut être liée à trois types de phénomènes climatologiques:

- Les systèmes frontaux : les cirrus proviennent de la montée d'une masse d'air chaud au-dessus d'une masse d'air froid. Ces cirrus, appelés cirrus de front, sont par conséquent

situés à l'avant d'un front chaud ou à l'arrière d'un front froid. Ces conditions de formation mènent à la création de cirrostratus ou de cirrocumulus (optiquement assez épais et produisant des motifs de vagues).

- Un nuage convectif intense tel qu'un cumulonimbus : la convection conduit à la création d'un nuage très étendu verticalement. Lorsque le nuage atteint la tropopause, sa progression est stoppée et le sommet du nuage prend la forme d'une enclume. Après la dissipation de la partie basse du nuage par précipitation, la partie haute peut survivre par elle-même en tant que cirrus. Ce type de contexte mène à la formation de cirrostratus fibratus ou de cirrus spissatus.
- Les courants jets de la haute troposphère peuvent également mener à la création de cirrus, tel que des cirrus uncinus ou des cirrus fibratus [6].

2.6 Propriétés et climatologie des cirrus

Les cirrus sont donc des nuages de haute altitude, de température basse, et composés en majorité de cristaux de glace. Ils se présentent sous la forme de filaments blancs fibreux, d'aspect fin et délicat. Ils sont en général considérés comme non précipitants, cependant cette affirmation est à prendre avec précaution : il est possible que des particules de glace provenant de cirrus ensemencent des nuages situés en-dessous, d'altitude moins élevée (phénomène de précipitation de cristaux, ou virga). Par ailleurs, la glace présente dans les cirrus, en s'incorporant à des nuages à phase mixte, contrôle le déclenchement et la durée des précipitations. Elle a également un impact important sur les propriétés radiatives et la stabilité générale du nuage, sans oublier son rôle crucial dans les processus de transfert électrique et les processus chimiques.

Pour toutes ces raisons, de nombreuses études se sont penchées sur la distribution spatiale et temporelle de ces nuages. Leur couverture nuageuse moyenne varie entre 20 et 50%, suivant la saison et la position géographique. Leur étendue horizontale peut largement dépasser leur étendue verticale, ainsi il n'est pas rare qu'un cirrus d'une épaisseur de l'ordre du kilomètre s'étale sur un continent entier quasiment sans interruption [6].

2.5 Influence des cirrus sur le bilan radiatif (Propriétés radiatives des cirrus) :

La couverture nuageuse des cirrus est importante, que ce soit spatialement ou temporellement. Ce type de nuage a donc impact non négligeable sur le bilan radiatif terrestre dans son ensemble.

Cependant, leur contribution à ce bilan est encore mal connue, car les cirrus participent à l'établissement du bilan radiatif terrestre suivant deux effets antagonistes :

- Les cirrus réfléchissent la lumière du soleil. L'énergie ainsi détournée ne pénètre pas la troposphère terrestre, il s'agit donc d'une contribution négative au bilan énergétique (la troposphère se refroidit ou, plus précisément, ne se réchauffe pas).
- Les cirrus absorbent puis réémettent le rayonnement infrarouge terrestre. Une certaine quantité d'énergie reste donc prisonnière de la troposphère terrestre. Il s'agit par conséquent d'une contribution positive au bilan énergétique (la troposphère ne se refroidit pas) [6].

3. Les traînées de condensation

Les traînées de condensation sont créées par la condensation de la vapeur d'eau émise par les moteurs d'avion à très haute altitude. Elles se produisent généralement à 10 000 m d'altitude avec un taux d'humidité de plus de 68 %, une température à partir de -39° et sur des noyaux de congélations fournis en grande partie par les gaz de combustion

Elles s'estompent en général rapidement par sublimation mais peuvent se transformer, dans certaines conditions d'hygrométrie et de température, en nuages artificiels analogues à des cirrus allongés. On parle aussi de traînées de vapeur, traînées blanches.

La formation des traînées change l'albédo de l'atmosphère et l'augmentation du trafic aérien mondial produit ainsi un effet sur les échanges énergétiques de l'atmosphère, d'autant plus que le transport aérien tend à augmenter. Ces traînées par leurs impacts en termes d'effet de serre doubleraient la responsabilité du trafic aérien en termes de contribution au réchauffement, augmentant ainsi une part qu'on estimait autrefois faible par rapport à d'autres modes de transport.

3.1. Mécanisme de formation

Ces traînées ne se forment qu'à certaines conditions, qui ne se rencontrent pratiquement que dans la haute troposphère et un peu plus souvent en hiver :

- où l'air est à environ -40 °C
- où l'air est assez humide pour atteindre la saturation et former des cristaux de glace, spontanément ou si un élément supplémentaire déclenche le processus.
- où l'air contient des noyaux de congélation, capables de nucléé la vapeur d'eau qui forme alors des gouttelettes surfondues de condensation. Ces dernières peuvent persister jusqu'à -40 °C avant de se congeler sans l'intervention d'une particule d'aérosol servant de noyau

glacigène. Bien que l'atmosphère contienne de tels noyaux, ils sont en faibles concentration à haute altitude. Ce sont donc surtout les gaz éjectés par les moteurs et les particules qu'ils contiennent qui fournissent de tels éléments précipitant la formation des cristaux et l'apparition de la traînée. [5]

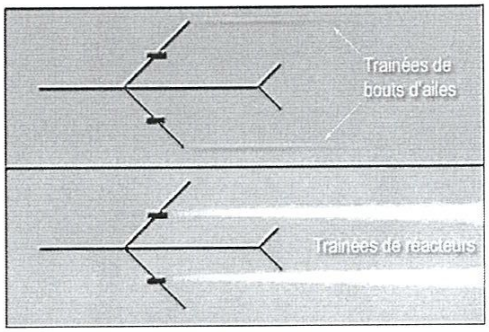


Fig.2.2. Les deux façons de former les traînées. En haut, celles prenant naissant au bout les ailes et en bas, celles provenant des réacteurs

3.1.1. Type

Il existe trois sources pouvant induire la formation de traînées visibles sur un avion: le moteur à hélice, la turbine de moteur à réaction et certains éléments de portance.

3.1.1.1. Traînées d'ailes

Quelques éléments de portance (ailerons ou extrémité d'aile) produisent un vortex tubulaire associé à une dépression sur le dessus de l'aile (ce qui permet à l'avion de voler). La chute de pression est la plus brusque en bout d'aile où elle entraîne une chute instantanée de température. Si l'avion vole dans une zone où l'humidité relative de l'air approche les 100 %, la baisse de température peut faire passer l'air au-delà de la saturation au bout des ailes.

Ces traînées sont rares et brèves car les cristaux de glaces sont rapidement sublimés en vapeur d'eau et l'humidité relative retombe sous 100 %. En effet, au contraire de ce qui se passe en ortie de réacteur ou de moteur

Fig.2.3. Exemple de condensation sur l'aile et de « trainée » se condensant en bout d'aile

3.1.1.2. Traînées de moteur à hélice

En sortie de moteur à hélice, les gaz chauds sont très humides, sont soumis à des phénomènes d'expansion/décompression quand ils sont pris dans le vortex de l'hélice et propulsés en arrière de l'avion. Si l'air est très humide et assez froid, ces gaz génèrent une traînée blanche, qui peut même prendre un aspect hélicoïdal. Cette trainée de condensation apparait quand la quantité d'humidité que peut contenir l'air est inférieure à celle ajoutée par les gaz d'échappement. Dès que l'air est en phase saturée, la vapeur se condense en microgouttelettes et éventuellement en cristaux de glace qui deviennent visibles. Selon les conditions de température et l'heure (jour nuit), ces traînées se dissipent en quelques minutes.

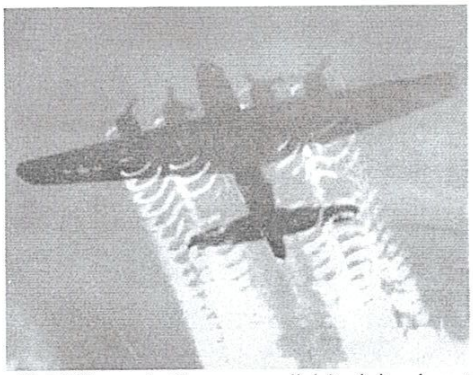


Fig.2.4. Cette forteresse volante B-17 (Seconde Guerre mondiale) génère deux types de trainées, l'une discrète, en bout d'aile droite, l'autre en sortie de moteur, avec une forme hélicoïdale

3.1.1.3. Traînées de moteur à réaction

En sortie de réacteur, les gaz d'échappement sont très chauds, très humides, riches en micro et nanoparticules, et subissent une brutale expansion/décompression qui les refroidit brutalement.

Chaque litre de carburant consommé produit environ un litre d'eau, qui va rapidement s'expansé en panache en vapeur, brutalement mise en contact avec l'air froid d'altitude.

Comme la quantité d'humidité que peut contenir l'air à ces altitudes est bien inférieure en général à celle venant du réacteur, l'air passe en phase saturée et la vapeur se condense alors en gouttelettes puis en cristaux de glace. [7]

3.2. Formation des traînées de condensation

Les traînées de condensation (en anglais contrail) sont des nuages artificiels analogues à des cirrus allongés créés, le plus souvent, par les réacteurs d'avion. Ils sont composés d'eau liquide et surtout de cristaux de glace.

Nous passerons sous silence les phénomènes qui se passent à toutes les altitudes de dépression sur le dessus de l'aile entraînant un abaissement de température et une condensation de l'humidité. La principale cause des traînées de condensation est l'éjection des gaz d'échappement très chauds par les moteurs et qui contiennent beaucoup de vapeur d'eau (on considère que pour un litre de carburant consommé il est produit un volume quasiment équivalent en eau). Dans l'air raréfié à faible pression et surtout très froid des hautes altitudes, la vapeur d'eau se condense alors en de minuscules cristaux de glace qui forment le sillage blanc de l'avion (dépassement du point de saturation). [8]

3.3. Comment les traînées de condensation deviennent visibles

Dans la stratosphère, le rayonnement solaire, qui n'est pas filtré par la couche d'ozone ou l'épaisseur de l'atmosphère est beaucoup plus énergétique qu'au sol. Ce rayonnement solaire sert de catalyseur à la condensation de la vapeur d'eau émise par les réacteurs en microgouttelettes qui gèlent instantanément en devenant alors visibles sous la forme d'une traînée blanche. Selon les conditions de pression, température, vent, etc., cette traînée évoluera en progressivement en nuage d'altitude en devenant plus ou moins épais, large et durable. Ils dériveront ensuite selon les vents d'altitudes. Elles peuvent conserver aussi une forme rectiligne, se briser ou prendre une forme de zigzag avant de disparaître et enfin devenir visuellement invisible rapidement par évaporation, sublimation et/ou dispersion [6].

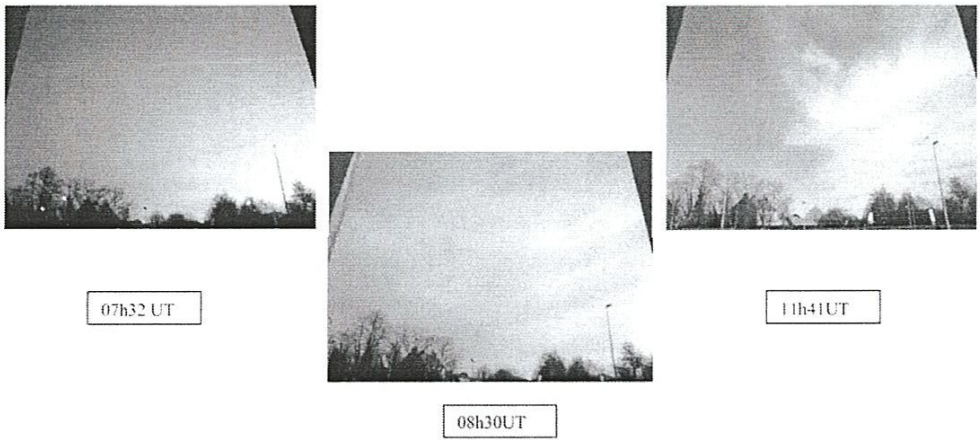


Fig.2.5. Traînées se transformant en nuage d'altitude

3.4. Traînées d'avion et influences climatiques

Les avions ont deux grands impacts connus sur le climat régional et planétaire, pour partie contradictoires :

- En émettant du CO_2 et de la vapeur d'eau (deux gaz à effet de serre), ils contribuent à moyen et long terme au réchauffement global;
- Par l'émission d'aérosols et la formation de traînées de condensation qui se transforment en larges cirrus, ils contribuent à court terme (surtout dans l'hémisphère nord où les vols sont plus nombreux) à un refroidissement.

À court terme, les premiers et plus importants impacts de l'aviation pourraient provenir des traînées de condensation et de l'induction des cirrus qu'elles provoquent. On n'a pris conscience de l'ampleur physique de ces impacts que dans les années 1990. En fait il y a eu de nombreuses études sur les contra ils à la fin des années 1950. Les États-Unis essayent de critiquer le premier avion à réaction commercial, conduit au royaume uni, en l'accusant de provoquer un refroidissement de l'atmosphère. Les scientifiques de la NASA en 1998 ont montré que les traînées de condensation produites sur la côte pacifique des États-Unis d'Amérique pouvaient s'étaler et fusionner pour produire un cirrus couvrant 3600 km^2. Des photographies satellite ont ensuite dévoilé des traînées de condensation produites par l'aviation commerciale au-dessus de la Nouvelle-Angleterre (formant un nuage qui a atteint 34000km^2). Divers travaux de recherche visent à mieux quantifier la part respective de ces deux

phénomènes, notamment en Europe de l'Ouest, dont les images satellitaires montrent qu'elle est l'une des zones les plus touchées au monde. [9]

Fig.2.6. Traînées de condensation au-dessus de la Nouvelle-Écosse, vues de satellite

3.5. Réchauffement climatique

La connaissance de l'impact sur le réchauffement climatique des traînées de condensation et cirrus induits par l'aviation ne fait pas l'objet du même niveau de compréhension scientifique que celui d'autres GES (Gaz à effet de serre,), en particulier parce que les effets des cirrus sont localisés et dépendent très directement des conditions atmosphériques rencontrées.

Le pouvoir radiatif et le temps de présence dans l'atmosphère des cotras (Les oxydes de soufre SOx sont partiellement adsorbés par les particules et les rendent hydrophiles, ce qui contribue à la formation de traînées de condensation) dépendent de leur localisation, de leur géométrie et des caractéristiques de l'atmosphère ambiante. C'est ce qui explique la très grande variabilité des valeurs de forçage radiatif que l'on peut associer au cotras et qu'il est très difficile d'y associer un indice comme le PRG (Potentiel de réchauffement global). [10]

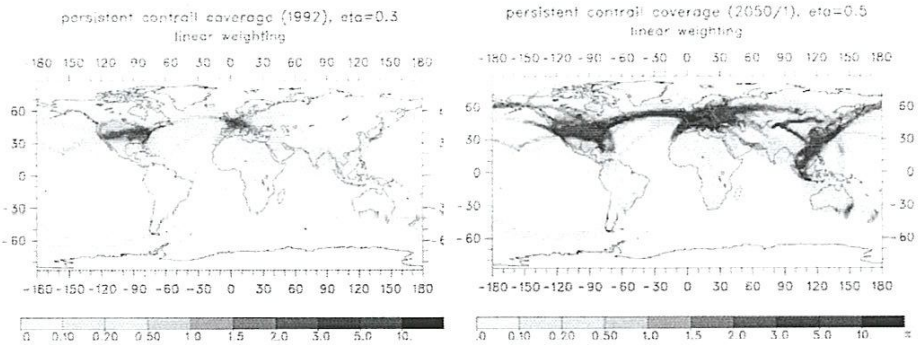


Fig.2.7. progression du réchauffement climatique

Conclusion

Dans ce chapitre nous avons caractérisé les nuages de type " cirrus " qui se forment dans la haute troposphère. Les nuages jouent un rôle important dans les mécanismes de réchauffement climatique. De plus, des cirrus artificiels sont des traînées d'avion et ajoutent une contribution non négligeable aux processus de pollution atmosphérique.

Chapitre 3

Les système multiagent et la plateforme JADE

1. Introduction

Dans ce chapitre, nous établirons tout d'abord une définition du concept d'agent autonome ainsi que des caractéristiques d'un tel agent. Nous verrons ensuite à travers un bref historique des travaux effectués dans le domaine, les défis auxquels les chercheurs ont eu à faire face dans le domaine de la technologie agent.

2. Les agents

Le concept d'agent a été l'objet d'études pour plusieurs décennies dans différentes disciplines. Il a été non seulement utilisé dans les systèmes à base de connaissances, la robotique, le langage naturel et d'autres domaines de l'intelligence artificielle, mais aussi dans des disciplines comme la philosophie et la psychologie. Aujourd'hui, avec l'avènement de nouvelles technologies et l'expansion de l'Internet, ce concept est encore associé à plusieurs nouvelles applications comme agent ressource, agent courtier, assistant personnel, agent interface, agent ontologique, etc. Dans la littérature, on trouve une multitude de définitions d'agents. Elles se ressemblent toutes, mais diffèrent selon le type d'application pour laquelle est conçu l'agent.

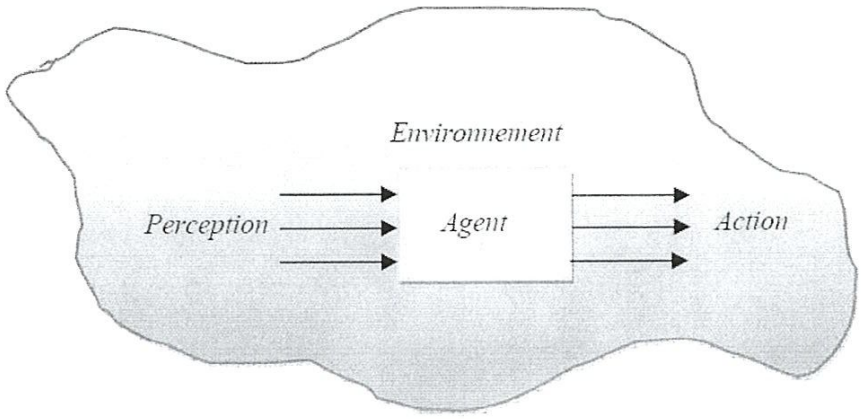


Fig.3.1. Aperçu externe et général d'un agent

2.1. Définition d'un agent

- **Selon Ferber :**

Un agent est une entité autonome, réelle ou abstraite, qui est capable d'agir sur elle-même et sur son environnement, qui, dans un univers multiagent, peut communiquer avec d'autres agents, et dont le comportement est la conséquence de ses observations, de ses connaissances et des interactions avec les autres agents. Il ressort de cette définition des propriétés clés comme l'autonomie, l'action, la perception et la communication. D'autres propriétés peuvent être

attribuées aux agents. Nous citons en particulier la réactivité, la rationalité, l'engagement et l'intention.

Il ressort de cette définition des propriétés clés comme l'autonomie, l'action, la perception et la communication. D'autres propriétés peuvent être attribuées aux agents. Nous citons en particulier la réactivité, la rationalité, l'engagement et l'intention [20].

- **Selon Jennings, Sycara et Wooldridge:**

Un agent est un système informatique, situé dans un environnement, et qui agit d'une façon autonome et flexible pour atteindre les objectifs pour lesquels il a été conçu.

Les notions "situé", "autonomie" et "flexible" sont définies comme suit :

- situé : l'agent est capable d'agir sur son environnement à partir des entrées sensorielles qu'il reçoit de ce même environnement. Exemples : systèmes de contrôle de processus, systèmes embarqués, etc.
- autonome : l'agent est capable d'agir sans l'intervention d'un tiers (humain ou agent) et contrôle ses propres actions ainsi que son état interne.
- flexible : l'agent dans ce cas est :
  - capable de répondre à temps : l'agent doit être capable de percevoir son environnement et élaborer une réponse dans les temps requis;
  - proactif : l'agent doit exhiber un comportement proactif et opportuniste, tout en étant capable de prendre l'initiative au "bon" moment;
  - social : l'agent doit être capable d'interagir avec les autres agents (logiciels et humains) quand la situation l'exige afin de compléter ses tâches ou aider ces agents à accomplir les leurs [12].

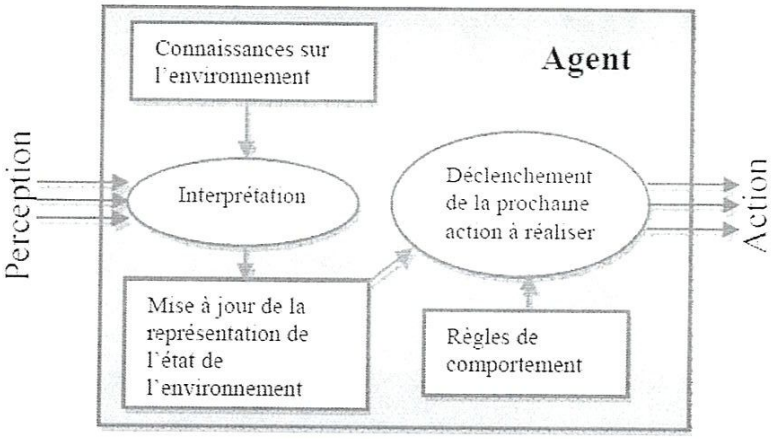


Fig.3.2. Exemple de décomposition d'un agent réactif

3. Les systèmes multiagents

Un système multiagent est un système distribué composé d'un ensemble d'agents. Contrairement aux systèmes d'IA, qui simulent dans une certaine mesure les capacités du raisonnement humain, les SMA sont conçus et implantés idéalement comme un ensemble d'agents interagissant, le plus souvent, selon des modes de coopération, de concurrence ou de coexistence [13].

Un SMA est généralement caractérisé par :

- chaque agent à des informations ou des capacités de résolution de problèmes limitées, ainsi chaque agent à un point de vue particl;
- il n'y a aucun contrôle global du système multiagent;
- les données sont décentralisées;
- le calcul est asynchrone.

Les SMA sont des systèmes idéaux pour représenter des problèmes possédant de multiples méthodes de résolution, de multiples perspectives et/ou de multiples résolveurs. Ces systèmes possèdent les avantages traditionnels de la résolution distribuée et concurrente de problèmes comme la modularité, la vitesse (avec le parallélisme), et la fiabilité (dûe à la redondance). Ils héritent aussi des bénéfices envisageables de l'Intelligence Artificielle comme le traitement symbolique (au niveau des connaissances), la facilité de maintenance, la réutilisation et la portabilité mais surtout, ils ont l'avantage de faire intervenir des schémas d'interaction sophistiqués. Les types courants d'interaction incluent la coopération (travailler ensemble à la résolution d'un but commun) ; la coordination (organiser la résolution d'un problème de telle sorte que les interactions nuisibles soient évitées ou que les interactions bénéfiques soient exploitées) ; et la négociation (parvenir à un accord acceptable pour toutes les parties concernées).

Les SMA sont à l'intersection de plusieurs domaines scientifiques : informatique répartie et génie logiciel, intelligence artificielle, vie artificielle. Ils s'inspirent également d'études issues d'autres disciplines connexes notamment la sociologie, la psychologie sociale, les sciences cognitives et bien d'autres. C'est ainsi qu'on les trouve parfois à la base des :

- systèmes distribués,
- interface personnes-machines,
- bases de données et bases de connaissances distribuées coopératives,
- systèmes pour la compréhension du langage naturel,
- protocoles de communication et réseaux de télécommunications,
- programmation orientée agents et génie logiciel,
- robotique cognitive et coopération entre robots,
- applications distribuées comme le web, l'Internet, le contrôle de trafic routier, le contrôle aérien, les réseaux d'énergie, etc.

Les recherches dans le domaine des systèmes multiagents poursuivent deux objectifs majeurs : Le premier concerne l'analyse théorique et expérimentale des mécanismes qui ont lieu lorsque plusieurs entités autonomes interagissent. Le second s'intéresse à la réalisation de programmes distribués capables d'accomplir des tâches complexes via la coopération et l'interaction. Leur position est donc double : d'un côté elles se placent au sein des sciences cognitives, des sciences sociales et des sciences naturelles pour à la fois modéliser, expliquer et simuler des phénomènes naturels, et susciter des modèles d'auto-organisation; de l'autre côté, elles se présentent comme une pratique, une technique qui vise la réalisation de systèmes informatiques complexes à partir des concepts d'agent, de communication, de coopération et de coordination d'actions.

Nous dressons maintenant un bref historique du domaine relatif aux SMA, en présentant des travaux précurseurs à ces systèmes ainsi que quelques exemples des premières applications développées.

3.1. Historique du SMA

En 1980, un groupe de chercheurs s'est réuni pour discuter des défis concernant la résolution "intelligente" de problèmes dans un système comportant plusieurs résolveurs de problèmes. Lors de cette réunion, il a été décidé que l'intelligence artificielle distribuée n'axerait pas ses travaux sur les détails de bas-niveau de la parallélisation ni sur comment paralléliser les algorithmes centralisés mais plutôt sur le fait de savoir comment un groupe de solutionneurs de problèmes pourrait coordonner ses efforts afin de résoudre des problèmes de manière efficace. On peut dire que les SMA ont vu le jour avec l'avènement de l'intelligence artificielle distribuée (IAD). A son début toutefois, l'IAD ne s'intéressait qu'à la coopération entre

résolveurs de problèmes afin de contribuer à résoudre un but commun. Pour y parvenir, on divisait en général, un problème en sous problèmes, et on allouait ces sous-problèmes à différents résolveurs qui sont appelés à coopérer pour élaborer des solutions partielles. Celles-ci sont finalement synthétiser en une réponse globale au problème de départ. Ainsi donc, l'IAD au départ privilégiait le "problème à résoudre" tout en mettant l'accent sur la résolution d'un tel problème par de multiples entités intelligentes. Dans les SMA d'aujourd'hui, les agents sont (entre autres) autonomes, possiblement préexistants et généralement hétérogènes. Dans ce cas, l'accent est plutôt mis sur le fait de savoir comment les agents vont coordonner leurs connaissances, buts et plans pour agir et résoudre des problèmes.

Le modèle des acteurs : Un des premiers modèles proposé à l'époque de l'IAD fut le modèle des Acteurs [14].Les acteurs sont des composantes autonomes d'un système qui communiquent par messages asynchrones. Ils sont composés d'un ensemble de primitives

- Create: créer un acteur à partir d'un ensemble de paramètres décrivant son comportement;
- Send: envoyer un message à un autre acteur;
- Become: changer l'état local d'un acteur.

Les acteurs s'avèrent être un modèle assez naturel pour le calcul parallèle. Cependant, les divers modèles d'acteurs, comme bien d'autres modèles de l'intelligence artificielle distribuée font face à un problème de cohérence. Leur granularité fine pose des problèmes de comportement dans des systèmes qui renferment plusieurs acteurs. Ils éprouvent également des difficultés à atteindre des buts globaux avec seulement des connaissances locales.

3.2. Interactions et coopération entre agents

Un système multiagent (SMA) se distingue d'une collection d'agents indépendants par le fait que les agents interagissent en vue de réaliser conjointement une tâche ou d'atteindre conjointement un but particulier. Les agents peuvent interagir en communiquant directement entre eux ou par l'intermédiaire d'un autre agent ou même en agissant sur leur environnement. Chaque agent peut être caractérisé par trois dimensions : ses buts, ses capacités à réaliser certaines tâches et les ressources dont il dispose. Les interactions des agents d'un SMA sont motivées par l'interdépendance des agents selon ces trois dimensions: leurs buts peuvent être compatibles ou non, les agents peuvent désirer des ressources que les autres possèdent; un agent X peut disposer d'une capacité nécessaire à un agent Y pour l'accomplissement d'un des plans d'action de Y.

3.2.1. Interactions entre agents coopératifs

La planification pour un seul agent, ne fait que construire une séquence d'actions en ne considérant que les buts de l'agent, ses capacités et les contraintes imposées par son environnement. Par contre, dans un environnement multiagent, on se doit de tenir compte des contraintes que les actions des autres agents placent sur le choix des actions de l'agent; des contraintes que les engagements de l'agent envers les autres imposent sur son propre choix d'action; ainsi que l'évolution imprévisible de l'environnement due à des agents qui ne sont pas modélisés.

3.2.2. Interactions entre agents égocentrés

Les interactions entre agents égocentrés se basent principalement sur la négociation. Cette dernière devient donc une méthode de coordination et de résolution de conflits. Elle a également été utilisée comme métaphore pour la communication de changements dans les plans, l'allocation de tâches ainsi que la résolution centralisée de violation de contraintes. On peut donc voir que la définition de la négociation est aussi imprécise que celle du concept d'agent. On arrive malgré tout à retrouver certaines caractéristiques importantes de la négociation :

- la présence d'une forme de conflit qui doit être résolu de façon décentralisée; par
- des agents égocentrés; ayant
- une rationalité limitée; et
- des informations incomplètes.

Comme le commerce électronique devient de plus en plus une réalité, le besoin de techniques de négociations qui tiennent compte des complexités du monde réel grandit. On se doit donc de tenir compte des informations incomplètes, des problèmes de négociations multiples, des échéances pour les négociations et la possibilité de briser des contrats. On a donc ajouté la possibilité d'établir des contrats moins rigides au sein du protocole Contract Net, des pénalités étant attribuées lorsqu'un contrat n'est pas rempli complètement [15].

3.3. Coordination entre agents

De nombreux exemples de coordination existent dans la vie quotidienne : deux déménageurs déplaçant un meuble lourd, deux jongleurs échangeant des balles avec lesquelles ils jonglent, des personnes qui parlent à tour de rôle en se passant un micro, etc. Malone [16] note que deux des composantes fondamentales de la coordination entre agents sont l'allocation de ressources rares et la communication de résultats intermédiaires. Dans ce contexte, les agents doivent être capables de communiquer entre eux de façon à pouvoir échanger les résultats intermédiaires.

Pour l'allocation des ressources partagées, les agents doivent être capables de faire des transferts de ressources. Ceci peut d'ailleurs imposer certains comportements à des agents particuliers.

Malone conclut qu'il peut être utile de distinguer les liens de contrôle comme une catégorie spéciale de liens de communication par lesquels certains agents transmettent des instructions que d'autres vont être motivés à suivre.

En étudiant les communautés humaines identifié trois processus fondamentaux de coordination : ajustement mutuel, supervision directe et coordination par standardisation. L'ajustement mutuel est la forme de coordination la plus simple qui se produit quand deux ou plusieurs agents s'accordent pour partager des ressources en vue d'atteindre un but commun. Habituellement les agents doivent échanger de nombreuses informations et faire plusieurs ajustements à leurs propres comportements en tenant compte des comportements des autres agents. Dans cette forme de coordination aucun agent n'a un contrôle sur les autres agents et le processus de décision est conjoint. La coordination dans les groupes de pairs (peer groups) et dans les marchés (market) est habituellement une forme d'ajustement mutuel. La supervision directe apparaît quand un ou plusieurs agents ont déjà établi une relation dans laquelle un des agents a un contrôle sur les autres. Cette relation est habituellement établie par ajustement mutuel comme par exemple dans le cas d'un employé ou d'un sous-contractant qui accepte de suivre les instructions du superviseur. Dans cette forme de coordination l'agent superviseur contrôle l'utilisation des ressources partagées (comme par exemple les ressources humaines, le temps de calcul ou l'argent) par les agents subordonnés. Il peut aussi imposer certains comportements. Dans les cas de coordination par standardisation le superviseur coordonne les activités en établissant des procédures que doivent suivre les subordonnés dans des situations identifiées. On trouve par exemple de telles procédures dans les entreprises, mais aussi dans les systèmes informatiques.

3.4. Négociation entre agents

La négociation joue un rôle fondamental dans les activités de coopération en permettant aux personnes de résoudre des conflits qui pourraient mettre en péril des comportements coopératifs. Durfee et ses collègues [17] définissent la négociation comme le processus d'améliorer les accords (en réduisant les inconsistances et l'incertitude) sur des points de vue communs ou des plans d'action grâce à l'échange structuré d'informations pertinentes. En général les chercheurs en IA distribuée utilisent la négociation comme un mécanisme pour coordonner un groupe d'agents.

3.5. Planification dans un environnement multiagent

La planification multiagent nécessite une forme ou une autre de synchronisation de plans qui peut être réalisée à divers moments : pendant la décomposition de plan, pendant la construction de plan ou après celle-ci. Les plans des agents peuvent être en conflits en raison d'incompatibilités d'états des systèmes, de l'ordre des activités ou de l'usage des ressources. De tels conflits peuvent être résolus par un agent en particulier (coordonnateur ou médiateur) ou une solution peut être obtenue par négociation.

3.6. Communication entre agents

Les agents peuvent interagir soit en accomplissant des actions linguistiques (en communiquant entre eux), soit en accomplissant des actions non-linguistiques qui modifient leur environnement. En communiquant, les agents peuvent échanger des informations et coordonner leurs activités. Dans les SMA deux stratégies principales ont été utilisées pour supporter la communication entre agents: les agents peuvent échanger des messages directement ou ils peuvent accéder à une base de données partagées (appelée tableau noir ou "blackboard") dans laquelle les informations sont postées. Les communications sont a la base des interactions et de l'organisation sociale d'un SMA.

3.6.1. Transfert de plans ou de messages

Dans l'approche de transfert de plans, un agent X communique son plan en totalité à un agent Y et Y communique son plan en totalité à X. Cette approche présente plusieurs inconvénients le transfert de plans est coûteux en ressources de communication et c'est une approche difficile à mettre en œuvre dans des applications réelles car il y a en général un fort degré d'incertitude au sujet des états actuel et futurs du monde. Dans des situations réelles en plus, il n'est pas possible de formuler à l'avance des plans en totalité. Par conséquent, on a besoin de communiquer des stratégies générales aux agents plutôt que des plans, et donc il est nécessaire que les agents puissent échanger des messages. De très nombreux travaux ont mis en oeuvre le transfert de message associé à l'usage de protocoles précis. Dans un SMA la complexité des communications dépend des caractéristiques des agents. Des agents réactifs utilisent un ensemble de règles de communication qui sont déclenchées quand des états spécifiques sont vérifiés ou quand des événements prédéterminés se produisent. Ces événements peuvent être des messages reçus ou des changements dans l'environnement. Les types de messages combinés avec les réponses attendues conduisent à la spécification de protocoles d'interaction.

3.6.2. Échange d'informations grâce à un tableau noir

En intelligence artificielle la technique du tableau noir ("blackboard") est très utilisée pour spécifier une mémoire partagée par divers systèmes [18]. Dans un SMA utilisant un tableau

noir, les agents peuvent écrire des messages, insérer des résultats partiels de leurs calculs et obtenir de l'information. Le tableau noir est en général partitionné en plusieurs niveaux qui sont spécifiques à l'application. Les agents qui travaillent sur un niveau particulier peuvent accéder aux informations contenues dans le niveau correspondant du tableau noir ainsi que dans des niveaux adjacents. Ainsi, les données peuvent être synthétisées à n'importe quel niveau et transférées aux niveaux supérieurs alors que les buts de haut niveau peuvent être filtrés et passés aux niveaux inférieurs pour diriger les agents qui œuvrent à ces niveaux. Le transfert de messages et les communications basées sur un tableau noir sont souvent combinés dans des systèmes complexes. Dans de tels systèmes, chaque agent est composé de plusieurs sous-systèmes (ou "sous agents") qui communiquent à travers un tableau noir. Les agents communiquent entre eux par échange de messages.

3.6.3. Actes de discours et conversations

Les philosophes du langage ont développé au cours des 25 dernières années la théorie des actes de discours qui considère que "dire quelque chose c'est en quelque sorte agir". Les paroles sont interprétées comme des actions appelées « actes » de discours. Cette théorie fournit un cadre qui identifie des types d'actes de discours primitifs et permet de dériver de ces types primitifs l'interprétation sous forme logique de n'importe quel verbe du langage qui exprime un acte de discours quelconque. Les chercheurs en intelligence artificielle ont très tôt pensé à étudier comment les actes de discours peuvent être utilisés pour influencer les états mentaux des agents.

3.6.4. Communication utilisant KQML, ACL et les conversations

Le langage KQML [19] a été proposé pour supporter la communication inter-agents. Ce langage définit un ensemble de types de messages (appelés abusivement "performatifs") et des règles qui définissent les comportements suggérés pour les agents qui reçoivent ces messages. Les types de messages de KQML sont de natures diverses : simples requêtes et assertions (ex: "ask", "tell"); instructions de routage de l'information ("forward" et "broadcast") commandes persistantes ("subscribe", "monitor"); commandes qui permettent aux agents consommateurs de demander à des agents intermédiaires de trouver les agents fournisseurs pertinents ("advertise", "recommend", "recruit" and "broker"). Ce langage a été développé de façon ad-hoc pour les besoins des développeurs d'agents logiciels : le terme "performatif" a été utilisé pour nommer diverses commandes qui ont une certaine ressemblance avec des verbes utilisés de façon performative dans le langage naturel; l'interprétation sémantique actuelle de ces commandes n'est pas satisfaisante.

Ces dernières années, KQML semble perdre du terrain au profit d'un autre langage plus riche sémantiquement ACL (pour Agent Communication Language). Un langage mis de l'avant par la FIPA qui s'occuppe de standardisé les communications entre agents. ACL est basé également sur la théorie du langage et a bénéficié grandement des résultats de recheche de KQML. Si toutefois, les deux langages se rapprochent au niveau des actes du langage, il n'en ait rien au niveau de la sémantique et il semble qu'un grand soin a été apporté au niveau de ACL tant au niveau de certains protocoles qui sont plus explites qu'au niveau de la sémantique des actes eux-mêmes.

4. la plateforme JADE

Le meilleur moyen pour construire un système multiagent(SMA) est d'utiliser une plate-forme multiagent. Une plate-forme multiagent est un ensemble d'outils nécessaire à la construction et à la mise en service d'agents au sein d'un environnement spécifique. Ces outils peuvent servir également à l'analyse et au test du SMA ainsi créé. Ces outils peuvent être sous la forme d'environnement de programmation (API) et d'applications permettant d'aider le développeur. Nous allons étudier dans cette partie la plate-forme JADE (Java Agent DEvelopment framework).

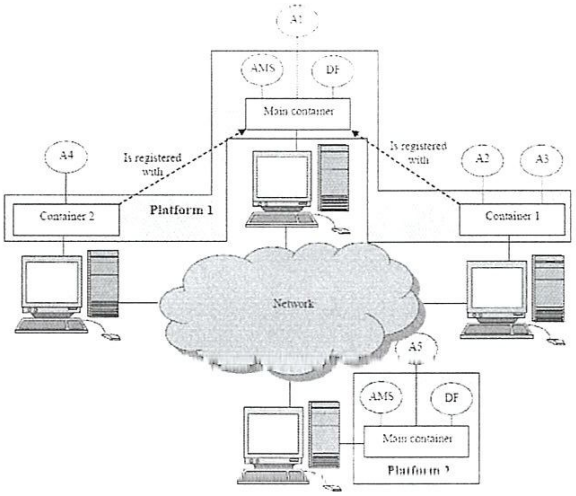


Fig.3.3. Illustration des concepts de base du Jade

4.1. Définition de JADE

JADE (Java Agent DEvelopement framework) est une plate-forme multi-agent créé par le laboratoire TILAB et décrite par Bellifemine et al. [21]. JADE permet le développement de

systèmes multiagents et d'applications conformes aux normes FIPA. Elle est implémentée en JAVA et fourni des classes qui implémentent « JESS » pour la définition du comportement des agents. JADE possède trois modules principaux (nécessaire aux normes FIPA).

- DF « Director Facilitor » fournit un service de « pages jaunes» à la plate-forme ;
- ACC «Agent Communication Channel » gère la communication entre les agents ;
- AMS « Agent Management System » supervise l'enregistrement des agents, leur authentification, leur accès et l'utilisation du système.

Ces trois modules sont activés à chaque démarrage de la plate-forme.

4.2. La norme FIPA

La FIPA (Foundation for Intelligent Physical Agents) est une organisation à but non lucratif fondée en 1996 dont l'objectif est de produire des standards pour l'interopération d'agents logiciels hétérogènes. Par la combinaison d'actes de langages, de logique des prédicats et d'ontologies publiques, la FIPA cherche à offrir des moyens standardisés permettant d'interpréter les communications entre agents de manière à respecter leur sens initial, ce qui est bien plus ambitieux que XML, qui ne standardise que la structure syntaxique des documents. Afin d'atteindre ce but, le FIPA émet des standards couvrant :

- Les applications (applications nomades, agent de voyage personnel, applications de diffusion audiovisuelles, gestion de réseaux, assistant personnel…) ;
- Les architectures abstraites, définissant d'une manière générale les architectures d'agents ;
- Les langages d'interaction (ACL), les langages de contenu (comme SL, CCL, KIF ou RDF) et les protocoles d'interaction ;
- La gestion des agents (nommage, cycle de vie, description, mobilité, configuration);
- Le transport des messages : représentation (textuelle, binaire ou XML) des messages ACL, transport (par IIOP, WAP ou HTTP) de ces messages.

4.3. Architecture de la plate-forme JADE

JADE reprend donc l'architecture de l'Agent Management Reference Model proposé par FIPA. Les différents modules présentés dans la figure suivante sont présentés sous forme de services.

Les services de base proposés sont le Directory Facilitator (DF) et l'Agent Management System (AMS). Il est possible de lui demander de tenir en plus le service de Message Transport Service (MTS) pour communiquer entre plusieurs plates-formes. Mais ce service sera chargé à la demande pour ne conserver par défaut que les fonctionnalités utiles à tout type d'utilisation.
L'agent est l'acteur fondamental de la plateforme, un "Agent Identifier" (AID) identifie un agent de manière unique. Le DF est un composant qui fait office d'annuaire. C'est un service de « pages jaunes » qui permet de mettre en relation les agents avec leurs compétences. Un agent peut enregistrer ses compétences dans le DF ou interroger le DF pour connaître les compétences proposées par les autres agents. L'AMS est un autre composant important car il contrôle l'accès et l'utilisation de la plate-forme et maintient un répertoire contenant les adresses de transport des agents de la plateforme. Ce service est plus un service de type «pages blanches» qui effectuent la correspondance entre l'agent et l'AID. Chaque agent doit s'enregistrer à un AMS pour avoir un AID. Il n'y a qu'un AMS par plate-forme. Le MTS est une méthode par défaut de communication entre agents de différentes plates-formes. Cela permet l'interconnexion entre systèmes hétérogènes ou tout au moins de système ne communiquant pas de la meme façon. L'Agent Platform (AP) constitue l'infrastructure physique sur laquelle se déploient les agents. Il contient le DF, l'AMS et le MTS. Lorsqu'on parle de AP, on inclut souvent le matériel électronique, l'OS, le software et les composants cités ci-dessus avec les agents. Enfin, l'Agent Identifier (AID) est un identifiant précis d'un agent. On lui donne plusieurs paramètres tels que l'adresse de transport, l'adresse de service de résolution de nom,... Un exemple : name@HAP (Home Agent Platform)

Dans la plate-forme JADE, deux méthodes sont fournies par la classe Agent afin d'obtenir l'identifiant de l'agent DF par défaut et celui de l'agent AMS : getDefaultDF() et respectivement getAMS(). Ces deux agents permettent de maintenir une liste des services et des adresses de tous les autres agents de la plate-forme. Le service DF propose quatre méthodes afin de pouvoir :

- Enregistrer un agent dans les pages jaunes (register).
- Supprimer un agent des pages jaunes (deregister).
- Modifier le nom d'un service fourni par un agent (modify).
- Recherché un service (search).

Le service AMS s'utilise généralement de manière transparente (chaque agent créé est automatiquement enregistré auprès de l'AMS et se voit attribué une adresse unique). Ces deux

services fournissent donc les annuaires qui permettent à n'importe quel agent de trouver un service ou un autre agent de la plate-forme.

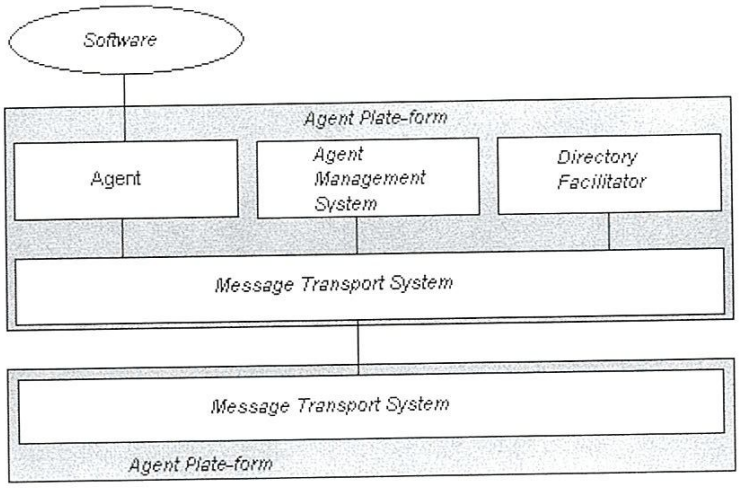


Fig.3.4. Architecture logiciel de La plate forme JADE

4.4. Langage de communication de la plate-forme JADE

Le langage de Communication de la plate-forme JADE est FIPA-ACL (Agent Communication language). La classe ACLMessage représente les messages qui peuvent être échangés par les agents. La communication de messages se fait en mode asynchrone. Lorsqu'un agent souhaite envoyer un message, il doit créer un nouvel objet ACLMessage, compléter ces champs avec des valeurs appropriées et enfin appeler la méthode send(). Lorsqu'un agent souhaite recevoir un message, il doit employer la méthode receive() ou la méthode blockingReceive().

Un message ACL dispose obligatoirement des champs suivants :

Performative :	type de l'acte de communication
Sender :	expéditeur du message
Receiver :	destinataire du message
reply-to :	participant de la communication
content :	contenu du message

language :	description du contenu
encoding :	description du contenu
ontology :	description du contenu
protocol :	contrôle de la communication
conversation-id :	contrôle de la communication
reply-with :	contrôle de la communication
in-reply-to :	contrôle de la communication
reply-by :	contrôle de la communication

Tab.3.1. table descriptive des champs de la class ACLMessege et leur signification

Tous les attributs de la classe ACLMessage peuvent être obtenus et modifiés par les méthodes set / get(). Le contenu des messages peut être aussi bien du texte que des objets car la sérialisation Java est supportée.

4.5. Comportements des agents dans la plate-forme JADE

Un agent doit être capable de gérer plusieurs tâches de manière concurrente en réponse à différents évènements extérieurs. Afin de rendre efficace cette gestion chaque agent de JADE est composé d'un seul thread et chaque comportement qui le compose est en fait un objet de type Behaviour. Des agents multi-thread peuvent être créés mais il n'existe pour l'heure actuelle aucun support fournis par la plate-forme (excepté la synchronisation de la file des messages ACL).

Afin d'implémenter un comportement, le développeur doit définir un ou plusieurs objets de la classe Behaviour, les instancier et les ajouter à la file des tâches « ready »de l'agent. Il est à noter qu'il est possible d'ajouter des comportements et sous-comportements à un agent ailleurs que dans la méthode setup()

Tout objet de type Behaviour dispose d'une méthode action() (qui constitue le traitement à effectuer par celui-ci) ainsi que d'une méthode done() (qui vérifie si le traitement est terminé). Dans les détails, l'ordonnanceur exécute la méthode action() de chaque objet Behaviour présent dans la file des tâches de l'agent. Une fois cette méthode terminée, la méthode done() est invoquée. Si la tâche a été complétée alors l'objet Behaviour est retiré de la file. L'ordonnanceur est non-préemptif et n'exécute qu'un seul comportement à la fois, on peut donc considérer la méthode action() comme étant atomique. Il est alors nécessaire de prendre certaines précautions

lors de l'implémentation de cette dernière, à savoir éviter des boucles infinies ou des opérations trop longues. La façon la plus classique de programmer un comportement consiste à le décrire comme une machine à états finis. L'état courant de l'agent étant conservé dans des variables locales.

Enfin, il existe également quelques méthodes supplémentaires afin de gérer les objets Behaviour :

- reset() qui permet de réinitialiser le comportement;
- onStart() qui définit des opérations à effectuer avant d'exécuter la méthode action();
- onEnd() qui finalise l'exécution de l'objet Behaviour avant qu'il ne soit retiré de la liste des comportements de l'agent;

La plate-forme JADE fournit sous forme de classes un ensemble de comportements ainsi que des sous-comportements prêt à l'emploi. Elle peut les exécuter selon un schéma prédéfini, par exemple la classe SequentialBehaviour est supportée et exécute des sous-comportements de manière séquentielle. Toutes les classes prédéfinies dans JADE héritent de la classe Abstraite Behaviour. On peut les citer :

- Classe SimpleBehaviour (abstraite): modélise un comportement simple. Sa méthode reset() n'effectue aucune opération.
- Classe CompositeBehaviour (abstraite) : modélise un comportement composé. Les actions effectuées par cette classe sont définies dans les comportements enfants.
- Classe FSMBehaviour : Cette classe hérite de CompositeBehaviour et éxecute des comportements enfants suivant un automate à états finis défini par l'utilisateur. Lorsqu'un comportement enfant termine, sa valeur de fin retournée par la fonction onEnd() indique le prochain état à atteindre. Le comportement correspondant à cet état sera exécuté à la prochaine exécution de la classe. Elle se termine lorsque qu'un comportement associé à un état final a été exécuté.
- Classe SenderBehaviour : elle étend la classe OneShotBehaviour et encapsule une unité atomique qui effectue une opération d'envoi de message.
- Classe ReceiverBehaviour : Elle encapsule une unité atomique qui effectue une opération de reception de message. Ce comportement s'arrête dès qu'un message a été reçu. S'il n'y a pas de message dans la file d'attente de l'agent ou que le message ne

correspond pas au MessageTemplate du constructeur de cette classe, alors il se met en attente.

- On peut aussi citer d'autres classes par exemples : Classe WakerBehaviour (abstraite), ParrallelBehaviour, SequentialBehaviour, CyclicBehaviour. (abstraite), OneShotBehaviour (abstraite).

4.6. Outils de débogage de JADE

Pour supporter la tâche difficile du débogage des applications multi-agents, des outils ont été développés dans la plate-forme JADE. Chaque outil est empaqueté comme un agent, obéissant aux mêmes règles, aux mêmes possibilités de communication et aux mêmes cycles de vie d'un agent générique (agentification de service).

4.6.1. Agent RMA Remote Management Agent

Le RMA permet de contrôler le cycle de vie de la plate-forme et tous les agents la composant. L'architecture répartie de JADE permet le contrôle à distance d'une autre plate-forme. Plusieurs RMA peuvent être lancés sur la même plate-forme du moment qu'ils ont des noms distincts.

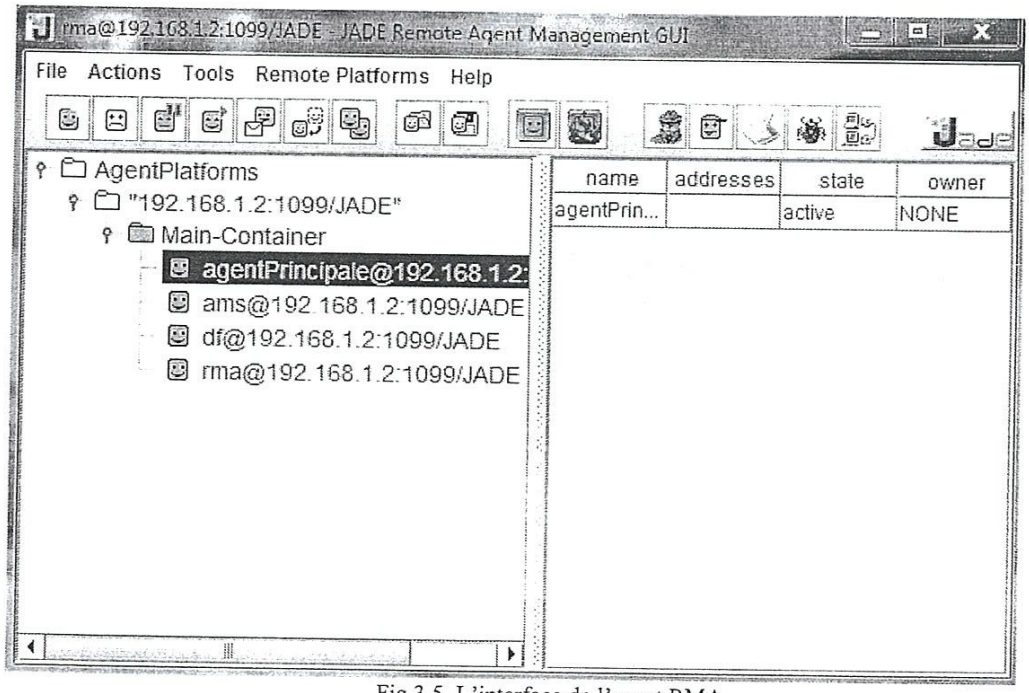


Fig.3.5. L'interface de l'agent RMA

4.6.2. Agent Direcory Facilitator

L'interface du DF peut être lancée à partir du menu du RMA .Cette action est en fait implantée par l'envoi d'un message ACL au DF lui demandant de charger son interface graphique. L'interface peut être juste vue sur l'hôte où la plate-forme est exécutée. En utilisant cette interface, l'utilisateur peut interagir avec le DF.

5. Conclusion

Nous avons vu, tout au long de ce chapitre que la technologie agent et multiagent n'est pas un concept voué à rester sur les tablettes des laboratoires de recherche puisque plusieurs exemples d'applications existent déjà.

Les personnes qui ont développé des SMA vous diront toutefois qu'il est difficile de concevoir et de bâtir un système multiagent. En effet, la construction de ce type de système comporte toutes les difficultés inhérentes aux systèmes répartis, auxquelles s'ajoute le caractère flexible et sophistiqué des interactions entre agents.

Dans le prochain chapitre on passera à la description des techniques utilisées dans notre application ainsi qu'on va expliquer tous les principes cités précédemment.

Chapitre 4

Conception & résultat

1. Introduction

Dans ce dernier chapitre, nous allons expliquer les déférentes phases d'implémentation de notre projet de fin d'étude. Nous allons en premier lieu expliquer la technique d'inversion de perspective, puis nous parleront sur le système géométrique utilisé dans notre application, ensuite nous allons parler de l'architecture de notre application et en fin on va faire une comparaison entre les résultats des déférents coefficients de corrélation.

2. Inversement de perspective

La photographie c'est la projection des rayons lumineuses reflète de la scène, sur un plan rétinal situé dans la caméra. La distance de la scène observée est très grande par rapport à la distance de l'objectif f. cependant, notre travail constitue d'observée une couche atmosphérique ou se trouve les trainés d'avion, ou l'altitude de cette couche est assez bien définie et dont l'épaisseur très faible par rapport à leur altitude (environ un dixième). Dans ce cas, il est possible d'adopter à priori une altitude moyenne de la couche, pour effectuer une inversion de perspective. En pratique, on introduit le concept d'une caméra virtuelle située à la verticale de la caméra réelle à une altitude L. cette caméra virtuelle à des propriétés identiques de celle de la caméra réelle. L'image fournisse par la projection de l'image original sur la caméra virtuel corresponde à la vue de la couche que pourrait observer un cosmonaute situé dans un vaisseau spatial à l'altitude L.

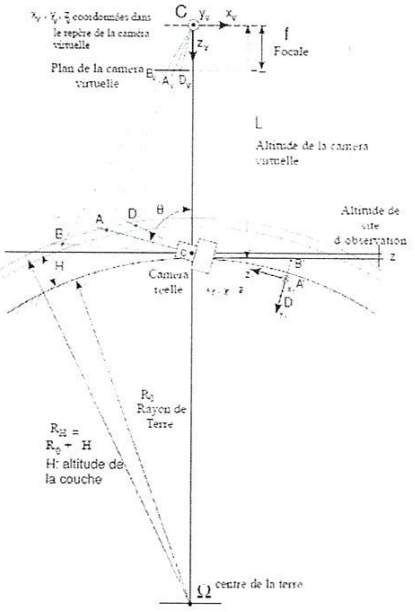


Fig.4.1 Inversion de perspective

La figure 4.1 illustre la technique d'inversion de perspective ou le point A de la couche a les coordonnées suivante : (X, Y, Z) par rapport au repère monde, (X_r, Y_r, Z_r) par rapport au repère caméra réelle, (X_v, Y_v, Z_v) par rapprot au repère caméra virtuel, (u_r, v_r) par rapport au repère image réelle et enfin (u_v, v_v) par rapport au repère image virtuel. C'est coordonné s'obtient à travers des transformations des repères. Nous allons expliquer comment ça se fait dans ce qui suit.

2.1. Transformation repère image caméra réelle – repère caméra réelle

La première transformation faite pour connaitre les coordonnées (X'_r, Y'_r, Z'_r) du point par rapport au repère caméra à partir des coordonnées image réelle (u_r, v_r) selon les formules suivant :

$$\begin{cases} X'_r = \dfrac{u_r - u_0}{\alpha_u} \\ Y'_r = \dfrac{v_r - v_0}{\alpha_v} \\ Z'_r = f \end{cases}$$

$u_u - k_u * f, u_v - k_v * f$, k_u est le facteur d'échelle horizontal , k_v le facteur d'échelle vertical ; u_0, v_0 les coordonnées du centre de projection dans le repère image ; f es la distance focale de la caméra ; k_u, k_v, u_0, v_0, f sont les paramètres intrinsèques de la caméra.

2.2. Transformation repère caméra réelle – repère monde

Après avoir les coordonnées du point A (X'_r, Y'_r, Z'_r) par rapport à la caméra réelle en peut déduire leur coordonnée (X, Y, Z) dans le repère monde à travers les formules suivante :

$$\begin{cases} X_r = C_t * X'_r \\ Y_r = C_t * Y'_r \\ Z_r = C_t * Z'_r \end{cases}$$

$$C_t = \frac{-R_H(Z'_r \sin\alpha_A) + \sqrt{R_H^2(Z'_r \sin\alpha_A)^2 + \left(X'^2_r + Y'^2_r + Z'^2_r\right) + (R_H^2 - R_h^2)}}{X'^2_r \mid Y'^2_r \mid Z'^2_r}$$

$R_H = R_t + H$, H est l'altitude de l'objet, R_t le rayon terrestre ; $R_h = R_t + h$, h est l'altitude du site d'observation.

α_A est la hauteur angulaire du point A par rapport au plan horizontal.

2.3. Transformation repère caméra réelle – repère caméra virtuelle – repère image virtuelle

Pour obtenir la projection du point A sur le plan de la caméra virtuelle, il faut déterminer ses coordonnées dans le repère de cette caméra(X_v, Y_v, Z_v). Pour passer du repère réel au repère virtuel il faut effectuer une rotation et une translation. Appelons α et θ la hauteur et l'azimut du point A. l'latitude de la caméra virtuelle est appelée L. les coordonnées (X_v, Y_v, Z_v) d'un point de l'image virtuelle s'expriment en fonction des coordonnées (X_r, Y_r, Z_r)du point correspondant de l'image réelle par la relation suivante :

$$\begin{cases} X_v = -\cos\theta . X_r - \sin\alpha\sin\theta . Y_r - \cos\alpha\cos\theta . Z_r \\ Y_v = -\sin\theta . X_r + \sin\alpha\cos\theta . Y_r + \cos\alpha\cos\theta . Z_r \\ Z_v = \cos\alpha . Y_r - \sin\alpha . Z_r + L \end{cases}$$

En ce qui concerne la projection sur le plan de l'image virtuelle, les coordonnées (X'_v, Y'_v, Z'_v) sont déterminées par les équations suivantes :

$$\begin{cases} X'_v = f * \dfrac{X_v}{Z_v} \\ Y'_v = f * \dfrac{Y_v}{Z_v} \\ Z'_v = f \end{cases}$$

Le dernier calcul est le passage des coordonnées dans le repère caméra virtuelle au repère image en appliquant la formule suivante :

$$\begin{pmatrix} su'_v \\ sv'_v \\ s \end{pmatrix} = \begin{pmatrix} k_u & 0 & 0 & u_0 \\ 0 & k_v & 0 & v_0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

2.4. Calcul de l'altitude de la caméra virtuelle

L'altitude de la caméra virtuelle se calcule comme suit :

$L = (f - y_{max} + f\frac{z_{max}}{Y_{max}})$ tel que :

$$y_{max} - \frac{R_h^2 - R_T^2}{R_h}\left(1 + \sqrt{\frac{R_h^2 - R_T^2}{R_H^2 - R_T^2}}\right)$$

$$z_{max} = \frac{R_T}{R_h}\sqrt{R_h^2 - R_T^2\left(1 + \sqrt{\frac{R_h^2 - R_T^2}{R_H^2 - R_T^2}}\right)}$$

Y_{max} =Taille d'un pixel * (Nombre de pixels/2)

H : Altitude de la couche atmosphérique étudiée, h : Altitude du site d'observation ;

$R_h = R_T + h; R_H = R_T + H;$ R_T: Rayon de la terre = 6371 Kilomètres [9].

3. Système géométrique utilisé

Nous allons utiliser dans notre système les repères suivants :

- Le repère monde (O, x, y, z) : l'origine de ce repère se situe sur le centre de la terre.
- Le repère de la caméra réelle (C_r, x, y, z) : C_r est le point central sur le plan optique. L'axe z est superposé sur l'axe optique.
- Le repère image réelle (u, v) : l'origine de ce repère, comme une matrice, se situe sur le point haut gauche. U et se sont les coordonnées des pixels, ou u est parallèle avec l'axe x de la caméra, et v parallèle avec l'axe y.
- Le repère de la caméra virtuel est identique de celle de la caméra réelle sauf que l'axe z (l'axe optique) est dirigé vers le centre de la terre.

Il nous faut aussi des informations sur les sites d'observation puisqu'ils interviennent dans la géométrie de notre système. Après avoir latitude et longitude des deux sites on peut calculer δlat et δlong.

δlat et δlong se sont respectivement les différences de latitude et longitude des deux sites calculer par les formules suivantes :

$$\delta lat = \text{asin}(\sin(latitude1) * \cos(latitude1) * \sin(latitude2) * \cos(longitude1))$$

$$\delta long = \text{asin}(\cos(latitude1) * \sin(longitude1)/\cos(\delta lat)$$

$$\beta = \delta long/2$$

Puis il faut calculer les centres des deux caméras réelles considèrent que la caméra 2 est la caméra de référence :

Centre optique C_1 de la caméra n°1 situé dans le repère monde (O, x, y, z) :

```
centre1.x = (Rt + L1) * Math.cos(dlat) * Math.sin(dlong2 / 2);
centre1.y = -(Rt + L1) * Math.sin(dlat);
centre1.z = -(Rt + L1) * Math.cos(dlat) * Math.cos(dlong2 / 2);
```

Centre optique C_2 de la caméra n°2 situé dans le repère monde (O, x, y, z) :

```
centre2.x = (Rt + L2) * Math.sin(-dlong2 / 2);
centre2.y = 0;
centre2.z = -(Rt + L2) * Math.cos(dlong2 / 2);
```

Pour avoir les coordonnées des centres des caméras virtuelles on peut employer les deux formules précédentes en remplaçant AltitudeSite par L_1 pour l'altitude de la première caméra et L_2 pour la deuxième.

```
L1 = projection1 * 1000. + hauteurCouche;
```

```
L2 = (L1 - hauteurCouche) * focale2 / focale1 + hauteurCouche;
```

On a focale1 = focale2 alors $L_1 = L_2$.

La prochaine phase est la projection des images sur les plans images des caméras virtuelle. Cette projection est une projection perspective. Considèrent un point $M(x, y, z)$, se projette sur le plan rétinal en un point $m(u, v)$. La matrice de projection est la suivante :

```
Pint.vec1.x = Alphau1;
Pint.vec1.y = 0;
Pint.vec1.z = u0;
Pint.vec2.x = 0;
Pint.vec2.y = Alphau1;
Pint.vec2.z = v0;
Pint.vec3.x = 0;
Pint.vec3.y = 0;
Pint.vec3.z = 1;
```

```
Alphau1 = focale1 / Taillepixel1;
```

```
u0 = (Ximg - 1) / 2.;
v0 = (Yimg - 1) / 2.;
```

X_{img}, Y_{img} sont les dimensions de l'image.

La transformation rigide qui passe du repère relatif de la caméra virtuelle au repère monde est appelée la matrice des paramètres extrinsèques noté P_{ext}.

$P_{ext} = \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix}$ ou R est un produit non commutatif de la matrice de rotation R_y^{β} suivant l'axe y et l'angle β par la matrice de rotation $R_x^{\delta lat}$ suivant l'axe x et l'angle δlat.

$R_1 = R_y^{\beta} * R_x^{\delta lat}$

```
Rx.vec1.x = 1;
Rx.vec1.y = 0;
Rx.vec1.z = 0;
Rx.vec2.x = 0;
Rx.vec2.y = Math.cos(dlat);
Rx.vec2.z = -Math.sin(dlat);
Rx.vec3.x = 0;
Rx.vec3.y = Math.sin(dlat);
Rx.vec3.z = Math.cos(dlat);

R1 = produitMatriciel(Rx, Ry);
```

On obtient ainsi la matrice $P = P_{int} * R_1$

```
P = produitMatriciel(Pint, R1);
```

Pour le deuxième repère de la caméra on applique les mêmes transformations et on utilise l'angle $-\beta$ pour la rotation par rapport à l'axe y et δlat pour la rotation par rapport à l'axe x. On obtient :

$R_2 = R_y^{-\beta} * R_x^{\delta lat}$ et $Q = P_{int} * R_2$

```
Rx.vec1.x = 1;
Rx.vec1.y = 0;
Rx.vec1.z = 0;
Rx.vec2.x = 0;
Rx.vec2.y = Math.cos(dlat);
Rx.vec2.z = Math.sin(-dlat);
Rx.vec3.x = 0;
Rx.vec3.y = Math.sin(dlat);
```

```
Rx.vec3.z = Math.cos(dlat);

Ry.vec1.x = Math.cos(-beta);
Ry.vec1.y = 0;
Ry.vec1.z = Math.sin(-beta);
Ry.vec2.x = 0;
Ry.vec2.y = 1;
Ry.vec2.z = 0;
Ry.vec3.x = -Math.sin(-beta);
Ry.vec3.y = 0;
Ry.vec3.z = Math.cos(-beta);

R2 = produitMatriciel(Ry, Rx);

Q = produitMatriciel(Pint, R2);
```

Les coordonnées (x, y, z) du point dans le repère caméra virtuelle sont égales au produit de la matrice des paramètres extrinsèque de la caméra par le vecteur des coordonnées (X, Y, Z) et le passage du repère caméra virtuelle au repère plan image est assuré par le produit des coordonnées (x, y, z) par la matrice des paramètres intrinsèques.

Après avoir calculé les coordonnées du point dans les repères des plans image virtuelle, nous obtient deux images projetés sur les caméras virtuelles. Pour extraire la carte de disparité (celle qui contienne les résultats de la mise en correspondance), on applique la géométrie épipolaire pour déterminé les couples homologues. Supposant qu'on a un point $W(X, Y, Z)$ qui a une projection m_1 sur le premier plan, pour déterminer la projection correspondant dans la deuxième image il faut calculer l'équation de la droite $(C_1 m_1)$ illustré dans la fig.1.2 relie le centre optique avec le centre du pixel courant. Cette droite sera utilisée dans le calcul du segment épipolaire qui

contient le point homologue m_2. Les extrémités du segment épipolaire sont définies par l'intervalle d'altitude $[Alt_{min}, Alt_{max}]$.

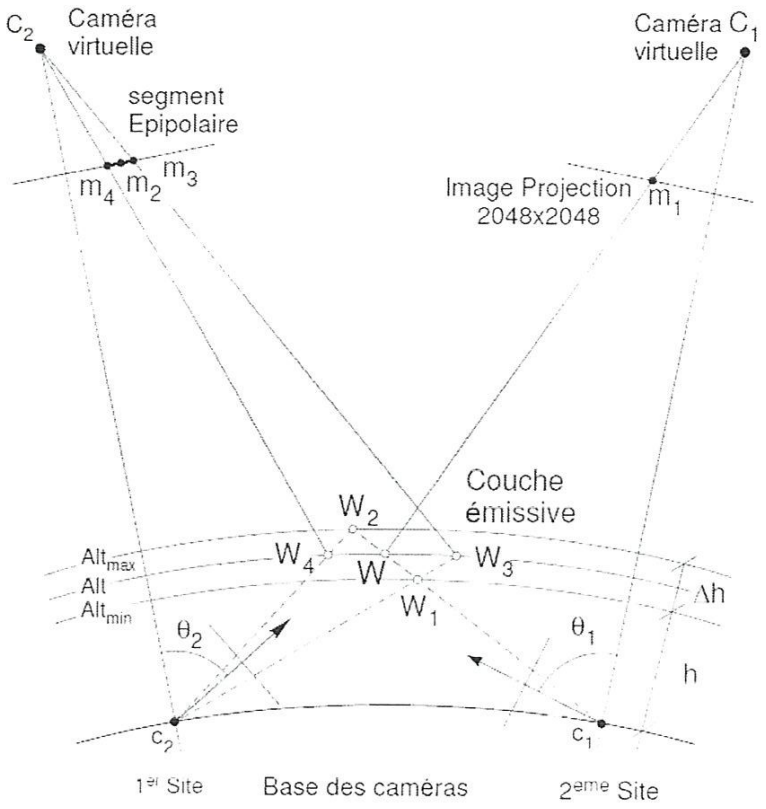


Fig.4.2 Géométrie du système stéréoscopique

Pour Alt_{min}

```
w = produitMatriceVecteur(InvP, m1);

w = additionVecteur(centreVirtuel1, w);

wtemp = produitMatriceVecteur(Q, w2);

temp = produitMatriceVecteur(Q, centreVirtuel2);

temp = produitVecteurScalaire(temp, -1);

wtemp = additionVecteur(wtemp, temp);
```

On calcule en suite les vecteurs directeurs (C_1O)et (C_1m_1)

```
module1 = moduleVecteur(centreVirtuel1);

vectdir1 = produitVecteurScalaire(centreVirtuel1, 1 / module1);
```

Et pour la deuxième

```
module1 = moduleVecteur(w);

vectdir2 = produitVecteurScalaire(w, 1 / module1);

vectdir2 = produitVecteurScalaire(vectdir2, -1);
```

On calcule l'altitude moyenne et les coordonnées du point de l'espace situé sur l'axe optique dans le repère du monde.

```
Wtemps = moduleVecteur(vectdir2);
Wtemps = (Rt + hauteurCouche) / Wtemps;
W = produitVecteurScalaire(vectdir2, 1 / Wtemps);
W = additionVecteur(centreVirtuel1, W);
Wtemps = moduleVecteur(W);
altitude = Wtemps - Rt;
```

On passe ensuite au calcul pour les points W_1 et W_2 qui correspondent à Alt_{min} et Alt_{max} telle que :

```
double alpha = (altitudemin - interval) / altitudemin;

W1.x = -alpha * W.x - (alpha - 1) * centreReal1.x;
W1.y = -alpha * W.y - (alpha - 1) * centreReal1.y;
W1.z = -alpha * W.z - (alpha - 1) * centreReal1.z;
```

Alors `double altitude_W1 = moduleVecteur(W1) - Rt;`

```
W2.x = -alpha * W.x - (alpha - 1) * centreReal1.x;
W2.y = -alpha * W.y - (alpha - 1) * centreReal1.y;
W2.z = -alpha * W.z - (alpha - 1) * centreReal1.z;

altitude_W2 = moduleVecteur(W2) - Rt;
```

On passe maintenant à la deuxième caméra pour trouver les points W_3 et W_4 :

```
alpha = (double) (hauteurCouche / 1000.)
                  / (double) (hauteurCouche / 1000. - interval);


W3.x = alpha * W1.x - (alpha - 1) * centreReal2.x;
W3.y = alpha * W1.y - (alpha - 1) * centreReal2.y;
```

```
W3.z = alpha * W1.z - (alpha - 1) * centreReal2.z;

altitude_W3 = moduleVecteur(W3) - Rt;

W4.x = alpha * W2.x - (alpha - 1) * centreReal2.x;
W4.y = alpha * W2.y - (alpha - 1) * centreReal2.y;
W4.z = alpha * W2.z - (alpha - 1) * centreReal2.z;

altitude_W4 = moduleVecteur(W4) - Rt;
```

On utilise les points (W_3W_1) et (W_4W_2) pour trouver les points exacts sur les droites (C_2W_1) respectivement (C_2W_2) qui ont une altitude Alt exacte.

$$t_1 = \frac{\text{AltitudeCouche} - \text{Alatitude}(W_1)}{\text{Alatitude}(W_3) - \text{Alatitude}(W_1)} \Rightarrow W_3 = t_1 * W_3 - (1 - t_1) * W_1$$

$$t_2 = \frac{\text{AltitudeCouche} - \text{Alatitude}(W_2)}{\text{Alatitude}(W_4) - \text{Alatitude}(W_2)} \Rightarrow W_4 = t_2 * W_4 - (1 - t_2) * W_2$$

On projette les deux nouveaux points W3, W4 sur le second plan image pour trouver les extrémités de notre segment épipolaire W_{min} et W_{max} :

```
wmin = produitVecteurScalaire(W4, -1);
wmin = additionVecteur(centreVirtuel2, wmin);
wmin = produitMatriceVecteur(Q, wmin);

wmax = produitVecteurScalaire(W3, -1);
wmax = additionVecteur(centreVirtuel2, wmax);
wmax = produitMatriceVecteur(Q, wmax);
```

Pour déterminer la position du point m2 sur le segment $[W_{min}, W_{max}]$ apparié sur la droite, on calcule les coefficients de la droite A, B tels que :

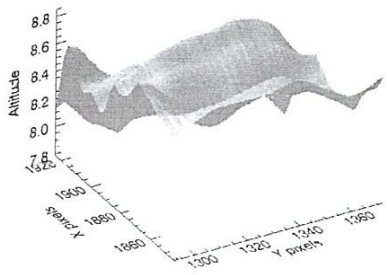
```
A = (wmax.y / wmax.z - wmin.y / wmin.z)
                    / (wmax.x / wmax.z - wmin.x / wmin.z);
B = wmax.y / wmax.z - A * wmax.x / wmax.z;

m2xmin = (int) (wmin.x / wmin.z + 0.5);
m2xmax = (int) (wmax.x / wmax.z + 0.5);
m2ymin = (int) (wmin.y / wmin.z + 0.5);
m2ymax = (int) (wmax.y / wmax.z + 0.5);
```

Pour le pixel $m_1(x, y)$ on trouve :

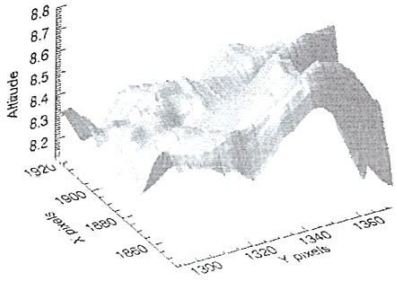
```
dxmax = m2xmax - x;
dxmin = m2xmin - x;
dymax = m2ymax - y;
dymin = m2ymin - y;
```

4. Présentation des résultats obtenue avec l'application précédente

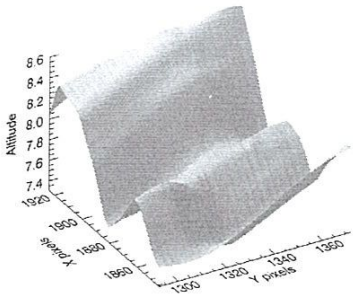
Notre travail est basé sur des observations faites dans deux sites qui se situent en France, le premier en Marney et le deuxième en Saint-Thiébaud. L'intérêt de cette observation c'est de faire une étude sur les cirrus naturel et artificiel (traîné d'avion). L'équipe d'observation a réalisé une application qui calcul l'altitude de cirrus observer utilisant la stéréovision. Dans cette section on va présenter les résultats produit avec l'ancienne application.



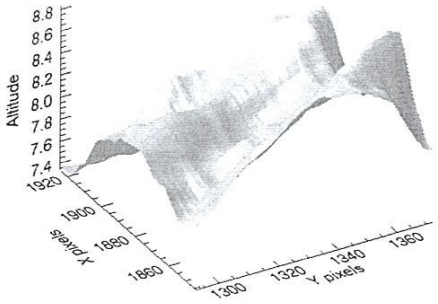
NCC 1 min 23 sec



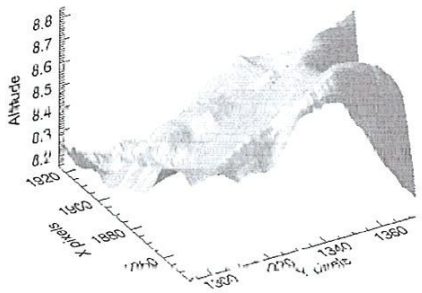
ZNNC 1 min 20 sec



SSD 2 min 29 sec



NZSSD 2 min 32 sec



MOR 1 min 24 sec

Dans ce qui suit nous allons présenter notre travail, après cela, nous allons faire une comparaison entre l'ancien résultat avec notre résultat.

5. Configuration et Architecture de notre application

5.1. Outils utilisé

- **Java**

Notre application est programmée avec Java. C'est un langage de programmation orienté objet, développé par Sun Microsystems. Il permet de créer des logiciels compatibles avec de nombreux systèmes d'exploitation (Windows, Linux, Macintosh, Solaris). Java donne aussi la possibilité de développer des programmes pour téléphones portables et assistants personnels. Enfin, ce langage

peut être utilisé sur internet pour des petites applications intégrées à la page web (applet) ou encore comme langage serveur (jsp).

- **Java Eclipse**

Eclipse IDE est un environnement de développement intégré libre (le terme Eclipse désigne également le projet correspondant, lancé par IBM) extensible, universel et polyvalent, permettant potentiellement de créer des projets de développement mettant en œuvre n'importe quel langage de programmation. Eclipse IDE est principalement écrit en Java (à l'aide de la bibliothèque graphique SWT, d'IBM), et ce langage, grâce à des bibliothèques spécifiques, est également utilisé pour écrire des extensions.

La spécificité d'Eclipse IDE vient du fait de son architecture totalement développée autour de la notion de plug-in (en conformité avec la norme OSGi) : toutes les fonctionnalités de cet atelier logiciel sont développées en tant que plug-in.

Plusieurs logiciels commerciaux sont basés sur ce logiciel libre, comme par exemple IBM Lotus Notes 8, IBM Symphony ou Websphere Studio Application Developer.

- **Plateforme Jade**

Notre système conçu sur le concept du système multiagent. Nous avons utilisé la plateforme Jade, puisque elle contienne tout ce qu'un développeur besoin dans les défirent phase de développement d'une application multiagent. On a déjà parlé de la plateforme Jade dans le chapitre précédent, mains on va faire un bref rappel.

Jade est un middleware qui facilite le développement des systèmes multiagents (SMA). JADE contient :

- Un runtime Environment : l'environnement ou les agents peuvent vivre. Ce runtime environment doit être activé pour pouvoir lancer les agents.
- Une librairie de classes : que les développeurs utilisent pour écrire leurs agents
- Une suite d'outils graphiques . qui facilitent la gestion et la supervision de la plateforme des agents

Chaque instance du JADE est appelée conteneur " container ", et peut contenir plusieurs agents. Un ensemble de conteneurs constituent une plateforme. Chaque plateforme doit contenir un conteneur spécial appelé main-container et tous les autres conteneurs s'enregistrent auprès de celui-là dès leur lancement

- **Environnement IDL**

IDL (**I**nteractive **D**ata **L**anguage) est un environnement de programmation pour la visualisation scientifique. En quelques instructions avec une syntaxe proche du Fortran il permet la réalisation d'applications graphiques. IDL est disponible sur de multiples systèmes : UNIX, PC, MAC.

5.2. Architecture de notre application

Notre application contient deux package. Le premier contient l'ensemble des agents et le deuxième contient les classes outils que nous sommes besoin.

Comme nous avons déjà-dit, cette application basée sur les agents. Elle contienne trois agents.

Un agent principal et deux autres agents. Au premier lieu, l'agent principal s'exécute et charge deux images, puis il donne l'ordre pour chaque agent pour faire une serie de traitement et de calcul. Les résultats de traitement des deux agents retournent à l'agent principal, et à base de ces résultats, l'agent principal peut commencer la phase de calcul de similarité. Le diagramme ci-dessous montre la circulation des données.

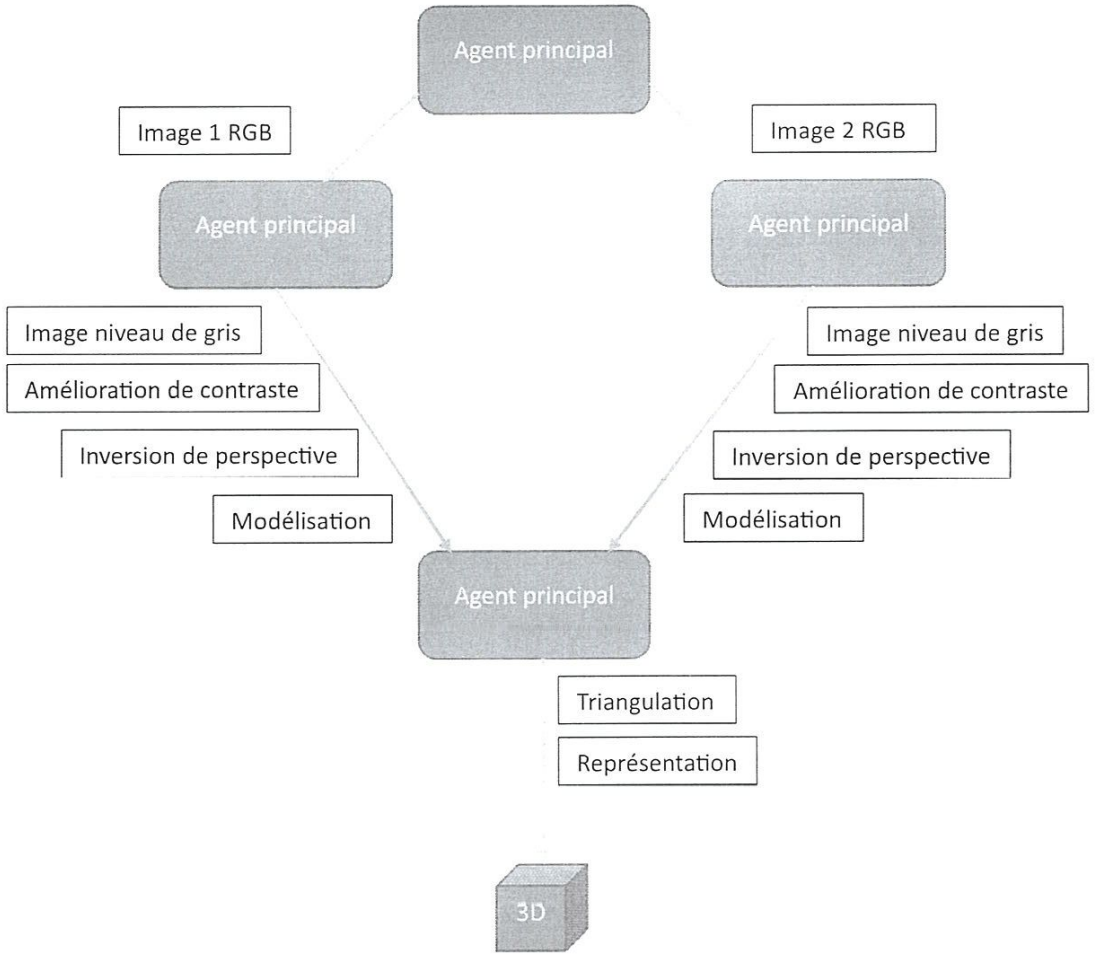


Fig.4.3. Schéma récapitulatif de notre application

Bon, maintenant on va expliquer comment l'exécution se déroule :

a. **Agent principal :** point de lancement de l'application responsable de la gestion de l'interface graphique et ses évènements.
b. **charger les deux images stéréoscopiques :** pour charger les deux images il faut aller au menu fichier > ouvrir, et sélectionné la premier image. Pour charger la deuxième cliqué une autre fois sur ouvrir un petit dialogue apparaitre pour choisir l'opération d'ouverture. Il y a deux cas :
    - si l'image 1 est déjà chargée, on obtient deux choix : charger la deuxième image et recharger la première

- Si l'image 2 est déjà chargée, on obtient deux choix : recharger la première image et recharger la deuxième image.

c. **Lancer le traitement :** après avoir chargé les deux images, il reste que lancer le traitement. Cette phase ce déroule comme suit :

- Transformation de l'espace colorimétrique RGB vers niveau de gris par la méthode RGB2Grey qui se trouve dans la classe Function. Voilà, la signature de la méthode ;

```
public static int[][] RGB2Grey(int[][] image, int largeur, int hauteur);
```

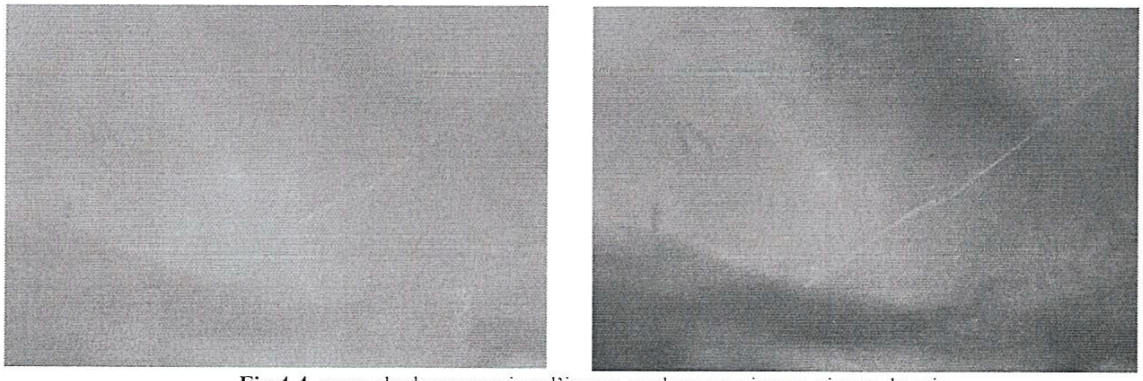


Fig.4.4. exemple de conversion d'image couleur vers image niveau de gris

Amélioration de contraste : nous traitons deux images de trainé d'avion, c'est pourquoi il faut séparer le trainé et le fond, et pour bien mettre en évidence les cirrus. Cette opération faite par la méthode contrasteAnhancement. On commence par une conversion vers le niveau de gris puis on calcule l'histogramme et l'histogramme cumulé puisque c'est une opération statistique.

```
public static int[][] contrasteAnhancement(int[][] image, int largeur,
            int hauteur);
```

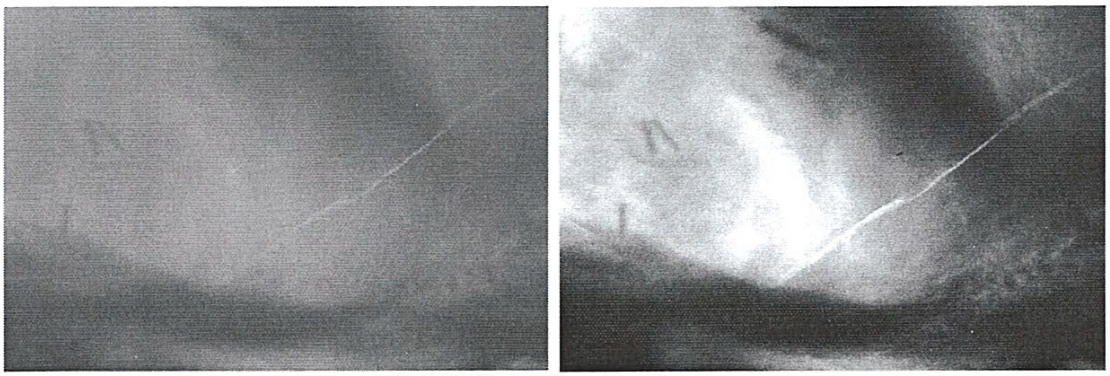


Fig.4.5. Résultat d'amélioration de contraste

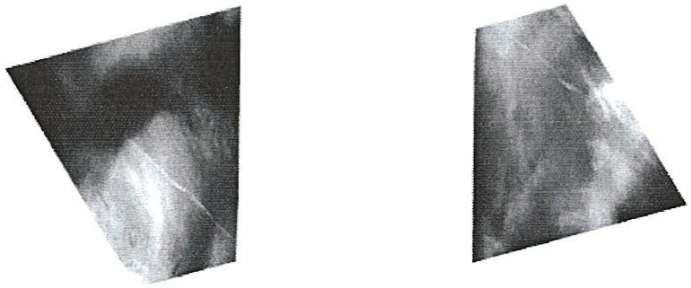
- **Projection** : cette étape est la dernière dans le processus de traitement d'image. Le résultat de cette étape est deux images projeté sur des plans virtuels.

  Comme on l'a vu précédemment, un certain nombre de paramètres sont nécessaires pour effectuer une inversion de perspective de type Satellite sur les deux images. Parmi eux certains sont liés aux appareils photographiques et aux objectifs (taille du pixel, distance focale), mais d'autres sont directement liés aux sites d'observations (altitude du site, hauteur de vue, latitude et longitude de chaque site, altitude choisie pour la caméra virtuelle L). Les paramètres sont les suivants :

Paramètre	**Marnay**	**Saint-Thiébaud**
Altitude de la couche	8 - 8.5 - 9 Km	8 - 8.5 - 9 Km
Altitude du site d'observation	275m	600 m
Azimut	165°	345°
Hauteur de visée	28-31	28-31
Altitude de la caméra virtuelle	150Km	150Km
Focale	35mm	35mm
Taille des pixels	5.7 µm	5.7 µm

Tab.4.1 – Les paramètres de projection.

La figure suivante montre deux projections obtenues pour deux images pris le 29 Avril 2010 à 22^h17 local (U.T.+2)



(a) (b)

Fig.4.6. Projection :(a) Site de St-Thiébaud, (b) Site de Marnay le 29/04/2010 à 20h17(U.T+2)

d. Triangulation

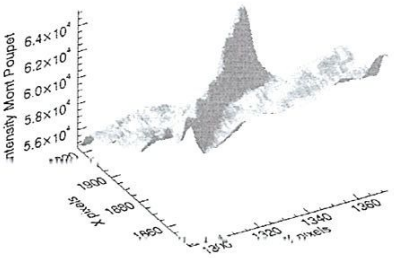
Dans notre travail l'objectif est d'établir une carte de disparité pour obtenir des informations sur la structure des cirrus. Nous avons utilisé une méthode pour calculer l'altitude moyenne du barycentre de la couche sur une zone géographique déterminée qui corresponde à un carré de 100*100 pixels. Une fois l'altitude moyenne déterminée, l'algorithme de triangulation permet d'obtenir la carte de disparité de la zone choisi.

Cette phase détermine les couples homologues entre les pixels des deux projections, par le biais de la mise en correspondance. Elle commence par le choix d'un des coefficients de corrélations qui se trouve dans le menu "Traitement". Après avoir choisi un coefficient, le programme demande de choisir une autre fois à entrer les deux projections. Ensuite, la fonction de triangulation calcule la ligne épipolaire, puis elle fait un appel à une des fonctions de calcule de disparité (dispariteZNNC, dispariteNNC, dispariteSSD, dispariteZNSSD, dispariteMOR). Enfin, la correspondance sera calculable, on ajoutant la disparité aux coordonnées de pixel courant. Les résultats (x1, y1, x2, y2, intensité1, intensité2, altitude, corrélation, facteur d'erreur sigma) seront enregistrés dans un fichier texte (resultat.txt).

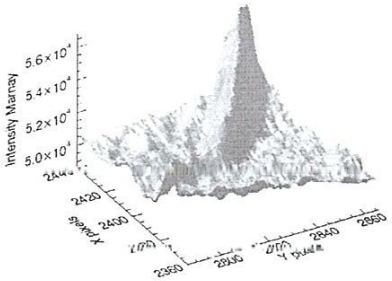
e. Afficher les résultats à l'aide d'IDL

❖ Résultat NCC

Nous calculons le coefficient de corrélation croisée (NCC) pour identifier les couples de points appariés dans les deux images prises au même instant dans les deux sites. Le NCC est un critère quantitatif bien défini qui permet de comparer les différents résultats obtenus lors de la phase d'identification des paires de points. La valeur maximum possible du NCC est 1, ce qui traduit une grande cohérence entre les structures observées dans les deux images projetées. La **Fig.4.14** montre le résultat d'application de NCC.



Intensité image gauche (Mont poupet)



Intensité image droite (Marney)

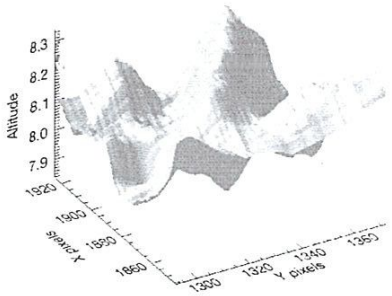
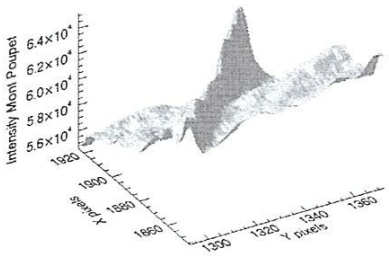
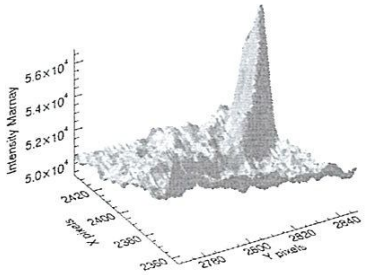


Fig.4.7. Application de NCC Altitude moyenne = 8 ± 0.5

❖ **Résultat ZNCC**



Intensité image gauche (Mont poupet)



Intensité image droite (Marney)

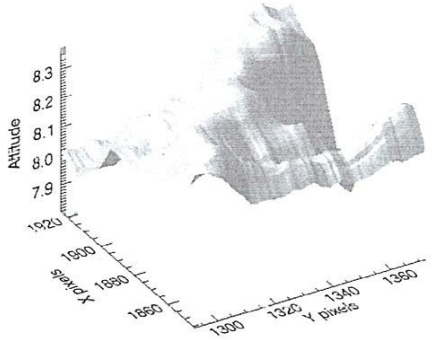
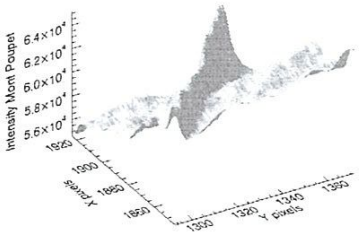
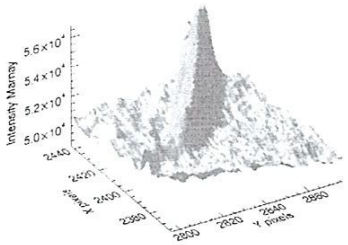


Fig.4.8. Application de ZNCC Altitude moyenne = 8 ± 0.5

❖ **Résultat SSD**



Intensité image gauche (Mont poupet)



Intensité image droite (Marney)

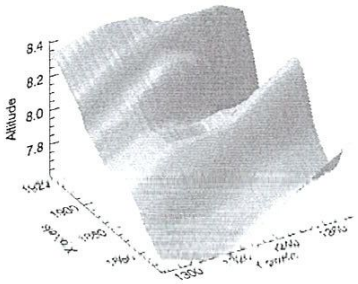
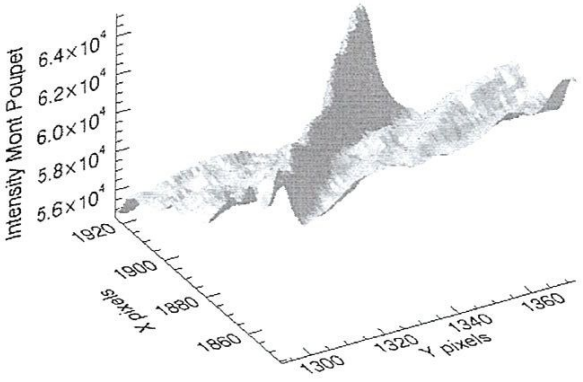
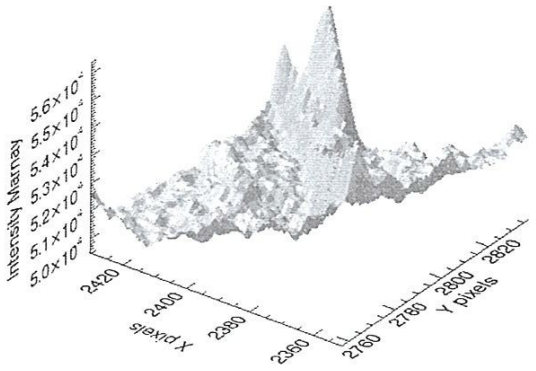


Fig.4.9. Application de SSD Altitude moyenne = 8 ± 0.5

❖ **Résultat ZNSSD**



Intensité image gauche (Mont poupet)



Intensité image droite (Marney)

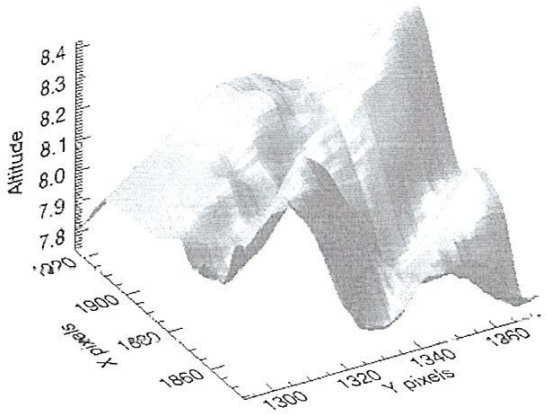
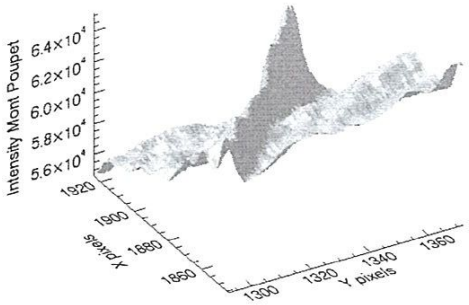
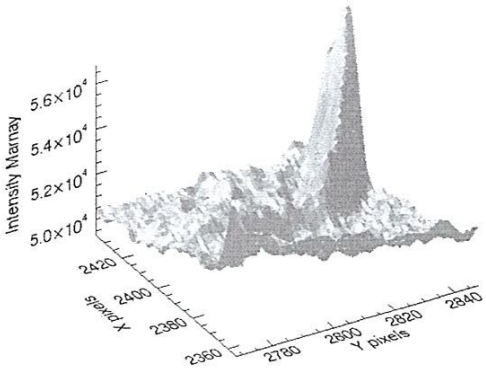


Fig.4.10. Application de ZNSSD Altitude moyenne = 8 ± 0.5

❖ **Résultat MOR**



Intensité image gauche (Mont poupet)



Intensité image droite (Marney)

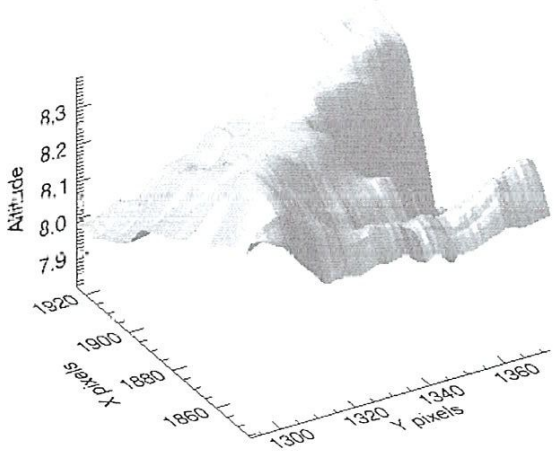


Fig.4.11. Application de ZNSSD Altitude moyenne = 8 ± 0.5

6. Discussion des résultats obtenus

Nous avons étudié quelque coefficient de corrélation, et nous avons obtenu les formes 3D précédemment présenté ainsi qu'un tableau qui montre le temps écoulé par chaque coefficient.

Coefficient de corrélation	Temps de calcul écoulé
NCC	6 sec, et 335 milliseconde(s)
ZNCC	3 sec, et 945 milliseconde(s)
SSD	10 sec, et 479 milliseconde(s)
NZSSD	15 sec, et 717 milliseconde(s)
MOR	4 sec, et 628 milliseconde(s)

Tab.4.2. table montrant le temps consacré par chaque coefficient

D'après les résultats obtenu en conclu que le NCC est le meilleur coefficient pour notre application, puisque il montre une forme d'altitude similaire a la forme des courbe d'intensité. En termes de temps de calcul en voir que le coefficient MOR est plus rapide, mais la forme est la première priorité, ce qui signifier que NCC est meilleur.

7. Conclusion

Le but essentiel de ce mémoire est de faire une comparaison judicieuse entre les défirent coefficient de corrélation. La comparaison faite sur deux critères essentiel : La forme 3D et le temps de calcul. Nous avons conclu que le NCC est le meilleur en termes de forme mais le ZNCC plus rapide en temps de calcul.

Conclusion générale

Dans ce mémoire nous avons abordé la mise en correspondance stéréoscopique avec ses aspects (géométrie épipolaire, corrélation des luminances, calcul de disparité…), ainsi que la reconstruction 3D en utilisant des méthodes géométriques d'un système stéréoscopique. Nous avons tenté de bâtir une application multiagent qui a pour but d'extraire la 3D à partir de deux images stéréo.

Nous avons utilisés deux appareils photographiques Canon 400D. Les images ont été prises en mode vis-à-vis en deux sites localisés en Franche-Comté avec une distance qui les sépare égale à 645 Km.

Les cirrus occupent une place privilégiée dans l'étude du climat. Leur présence a été observée sur l'intégralité du globe, et les moyens techniques d'aujourd'hui offrent enfin l'opportunité de pouvoir les étudier correctement. Par ailleurs, la complexité de leurs propriétés radiatives et optiques est unique parmi les nuages. Les principes de stéréoscopie et l'inversion de projection appliquée semblent concluants, mais peuvent être améliorés. En particulier, les méthodes d'appariement utilisées pour la restitution 3D doivent être étudiées afin de déterminer comment les appliquer aux objets diffus comme les structures des cirrus.

Référence bibliographique

[1] www.optique-ingenieur.org/fr/cours/OPI_fr_M04_C01/co/Contenu91.html

[2] Marr, D., Poggio, T. (1976). Cooperative computation of stereo disparity. Science.

[3] Kim, C., Lee, K., Choi, B., Lee, S. (2005). A dense stereo matching using two-pass dynamic programming with generalized ground control points. In IEEE Conference Proceedings of Computer Vision and Pattern Recognition, CVPR, vol. 2, pages 1075{1082, San Diego, Etats-Unis.

[4] Boukir, S., Bouthemy, P., Chaumette, F., Juvin, D. (1992). Mise en correspondance de segments dans une sequence d'images par une approche locale. Rapport de recherche 1792, Institut National de Recherche en Informatique et en Automatique, INRIA.

[5] Master OMR 2 Année 2004 /2005. Etude des propriétés optiques et radiatives des cirrus dans l'infrarouge thermique.

[6] Vincent NOEL. (2004) propriétés optiques et radiatives des cirrus par télédétection active apport des observations polarisée.

[7] fr.wikipedia.org/wiki/Tra%C3%AEn%C3%A9e_de_condensation#cite_note-LARC-30

[8] Alain LE GUE. (2001)Les traînées de condensations. Leurs impacts sur l'environnement et sur l'astronomie. Un essai de synthèse.

[9] Nadjib KOUAHLA. (2010) étude par stéréo imagerie des couches atmosphérique. Thèse de doctorat, université de Franche-comté, France.

[10] Chems CHKIOUA. (2009) aviation et Climat "Analyse des moyens d'action publique pour limiter l'impact de l'aviation sur le changement climatique".

[11] J. Ferber. (1995) Les systèmes multi-agents, vers une intelligence collective. InterEditions.

[12] N. R. Jennings, M. Wooldridge, and K. Sycara.(1998) A roadmap of agent research and development. Int Journal of Autonomous Agents and Multi-Agent Systems .

[13] B. Moulin and B. Chaib-draa.(1996) An overview of distributed artificial intelligence.

[14] G. Agha and C. Hewitt. Concurrent programming using actors. In A. Yonezawa and M. Tokoro, editors. (1988) Object-Oriented Concurrent Programming, Computer Systems Series, pages 37-53. MIT Press.

[15] T. Sandholm and V. Lesser. (1995) Issues in automated negotiation and electronic commerce: Extending the contract net framework. In Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-1), pages 328-335, San Francisco, CA.

[16] T. W. Malone. (1990) Organizing information processing systems: parallels between human organizations and computer systems. In W. W. Zachary and S. P. Robertson, editors, Cognition, Computation and Cooperation, pages 56-83. Ablex.

[17] E. H. Durfee and V. (1989)Lesser. Negotiating task decomposition and allocation using partial global planning. In L. Gasser and M. Huhns, editors, Distributed Artificial Intelligence Volume II, pages 229-244. Pitman Publishing: London and Morgan Kaufmann: San Mateo, CA.

[18] P. Nii, N. Aiello, and J. Rice. (1989) Experiments on Cage and Poligon: Measuring the performance of parallel blackboard systems. In L. Gasser and M. Huhns, editors, Distributed Artificial Intelligence, Volume II, pages 319-384. Pitman Publishing: London and Morgan Kaufmann : San Mateo, CA.

[19] T. Finin and R. Fritzson. (1994) KQML: a language and protocol for knowledge and information exchange. In Proceedings of the Thirteenth International Workshop on Distributed AArtificial Intelligence, pages 126-136, Lake Quinalt, WA.

[20] J. A. Alty, D. Griffiths, N. R. Jennings, E. H. Mamdani, A. Struthers, and M. E. Wiegand. ADEPT: (1994) Advanced decision environment for process tasks: Overview & architecture. In Proceedings of the 1994 BCS Expert Systems Conference (Applications Track, ISIP Theme), pages 359-371, Cambridge, UK.

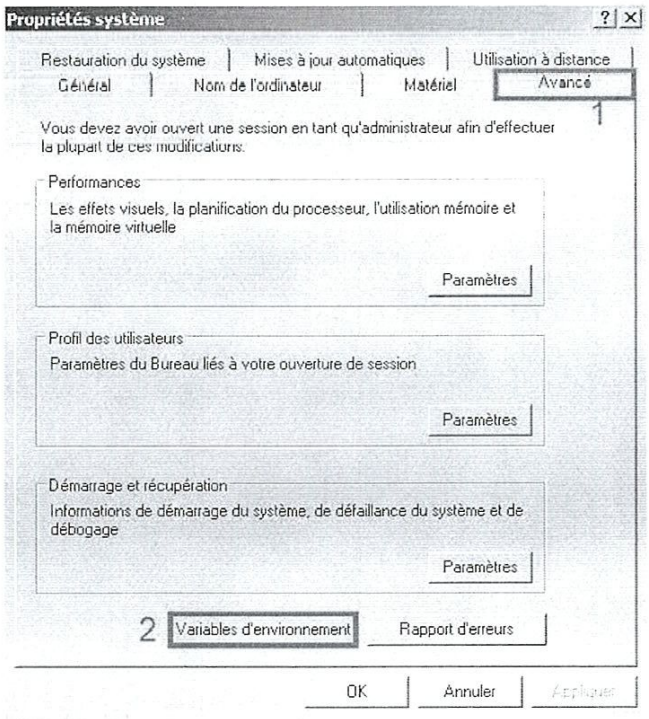
[21] Bellifemine F., Poggi A., Rimassa G. (1999) "JADE -- A FIPA-compliant agent framework", CSELT internal technical report. Part of this report has been also published in Proceedings of PAAM'99, London, pp.97-108.

Annexe

Configuration d'Eclipse avec Jade

Voici les étapes à suivre pour installer JADE :

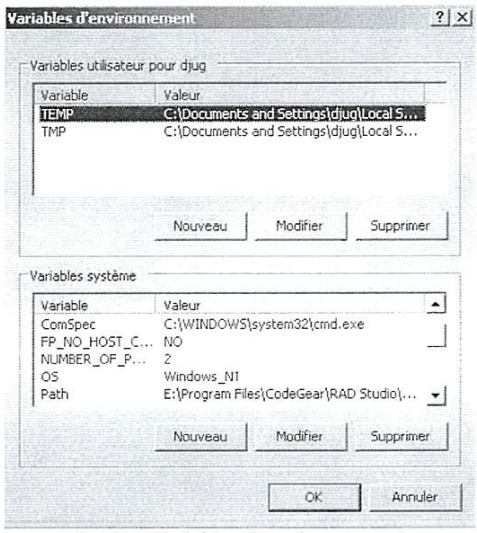
- Téléchargez le fichier JADE-all-3.6.zip de l'adresse suivante : http://jade.tilab.com/download.php
- décompressez le fichier (on va supposer tout au long de ce tutorial que le chemin du répertoire JADE-all-3.6 est le c:\ JADE-all-3.6). Apres avoir décompressé le fichier vous retrouvez quatre autres fichiers ZIP (JADE-bin-3.6.zip , JADE-doc-3.6.zip, JADE-examples-3.6.zip, JADE-src-3.6.zip). Décompressez ces 4 fichiers
- on doit maintenant mettre à jour la variable classpath (si elle n'existe pas encore il faut la créer) En faisant comme suit :



par Clic droit sur ordinateur, choisissez propriétés. La fenêtre propriétés système apparaît (fig.4.3, choisissez l'onglet Avancé Puis cliquez sur variables d'environnement

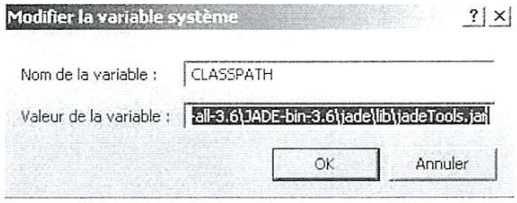
Fenêtre propriétés système

Une petite fenêtre intitulée " variables d'environnement " apparaît



Fenêtre Variables d'environnement

Dans la zone variable système, essayez de trouver la variable d'environnement qui porte le nom CLASSPATH. Si vous ne la trouvez pas, il faut la créer Maintenant que la variable est trouvée/créée on doit lui attribuer une valeur .cette valeur est la concaténation des chemins des quatre fichiers jar http.jar, iiop.jar, jade.jar, jadeTools.jar situés dans le chemin C:\jade\lib



Fenêtre Modifier la variable système

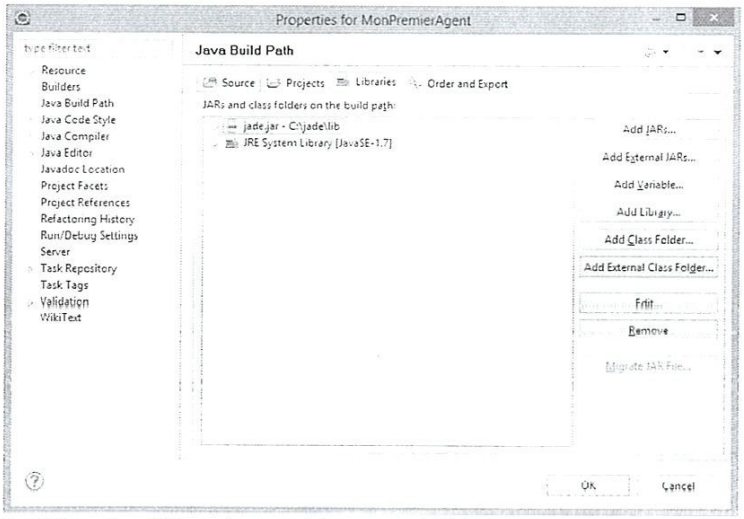
Création du premier agent avec JADE et ECLIPSE

Maintenant, nous allons créer notre premier agent. Ouvrir Eclipse et créer un nouveau projet (MonPremierAgent par exemple) ajouter un package (premierAgent) puis on crée une class appelée HelloWorldAgent, et on écrit le code suivant dedans :

```
package premierAgent;
import jade.core.Agent;
```

```
public class HelloWorldAgent extends Agent {

  protected void setup() {
        System.out.println("Hello World! My name is " + getLocalName());
        doDelete();
  }
}
```

On remarque qu'il existe beaucoup d'erreur, puisqu'on n'a pas ajouté la bibliothèque JADE. Pour résoudre ce petit problème,effectuez un clic droit sur le nom du projet Puis choisissez propriétés.



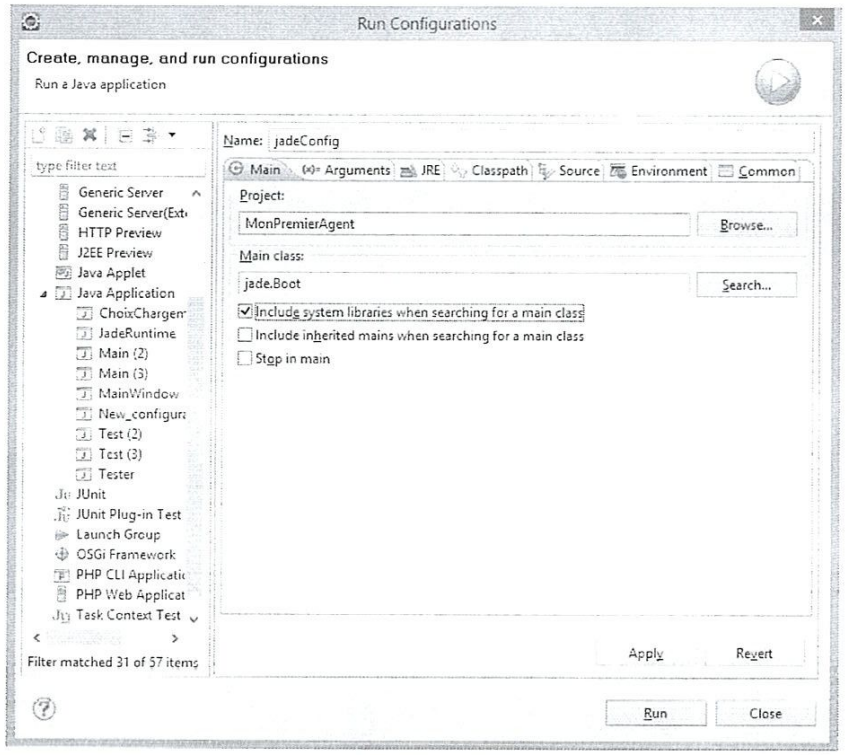
Cliquez sur java build path > Libraries > add external JARs

Configuration du projet MonPremierAgent dans Eclpise

Ajouter le fichier jade.jar situé dans "c:\jade\lib\jade.jar" puis cliquer OK.

Compiler et exécuter l'agent

Il reste à compiler et lancer l'agent pour cela : Allez dans run>>Run configuration puis double-cliquez sur java application Dans l'onglet " main ", dans le champ le plus haut Name modifier le nom de cette configuration (exemple jadeConfig) et dans la zone de saisie Main class, tapez le code suivant : jade.Boot puis cochez la case : " Include librairies when searching for a main class"



Fenêtre de configuration de création, de gestion et d'exécution : Arguments anglet : Main anglet

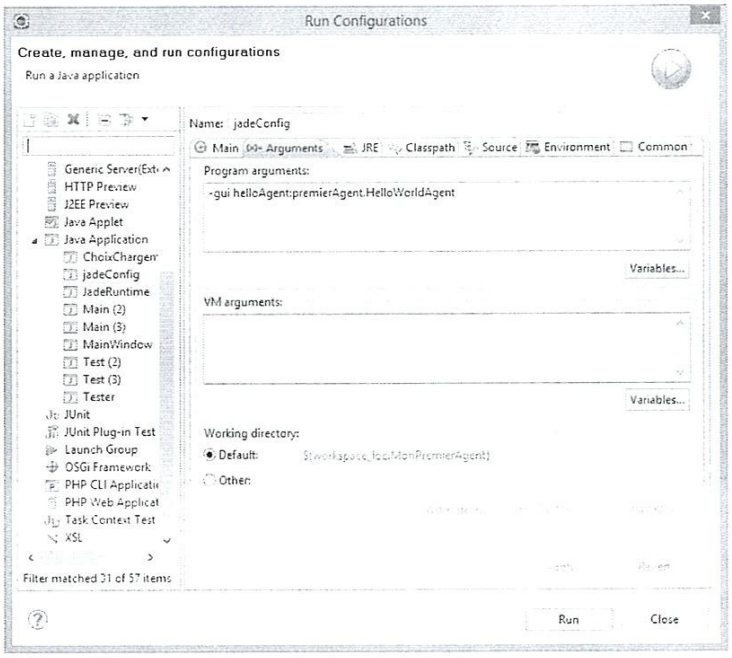
Et dans l'onglet arguments, tapez le code suivant :

```
-gui NomDuL'agent:LeNomDuPackage.LeNomDeLaClasse
```

Dans notre petit exemple on écrit :

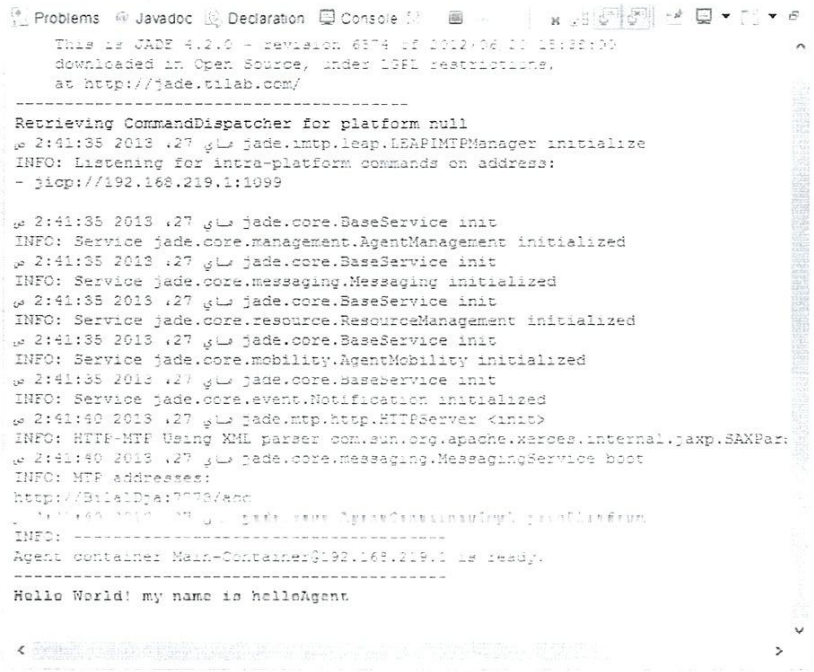
```
-gui helloAgent:premierAgent.HelloWorldAgent
```

Maintenant, cliquez sur "apply" pour ne pas refaire cette configuration plusieurs fois.



Fenêtre de configuration de création, de gestion et d'exécution : Arguments anglet

Cliquez sur run pour voir le résultat (affichage de « Hello World! My name is "helloAgent" » Dans la console et ouverture du la plateforme jade)



Résultat de l'exécution de notre petit exemple