Planification de trajectoires avion : approche par analogie lumineuse.

Nour Elhouda Dougui

THÈSE

En vue de l'obtention du
DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :
Université Toulouse III Paul Sabatier (UT3 Paul Sabatier)

Discipline ou spécialité :
Mathématiques appliquées et Informatique

Présentée et soutenue par
Nour Elhouda Dougui
le : jeudi 15 décembre 2011

Titre :
Aircraft trajectories planning : light propagation model

Jury :
Daniel Delahaye : Enseignant chercheur à l'École Nationale de l'Aviation Civile (Co-directeur)
Jean-Baptiste Hiriart-Urruty : Professeur à l'Université Paul Sabatier (Examinateur)
Marcel Mongeau : Maître de conférences HDR à l'Université Paul Sabatier (Co-directeur)
Adam Ouorou : Ingénieur R&D HDR à Orange Labs (Examinateur)

École doctorale :
École Doctorale Aéronautique Astronautique (EDAA)

Unité de recherche :
MAIAA : Laboratoire de Mathématiques Appliquées Informatique et Automatique pour l'Aérien (École Nationale de l'Aviation Civile) et IMT : Institut de Mathématiques de Toulouse UMR 5219

Directeurs de thèse :
Daniel Delahaye
Marcel Mongeau

Rapporteurs :
Eric Feron : Professeur à Georgia Institute of Technology (États-Unis)
Mohamed Haouari : Professeur à l'école Nationale d'Ingénieurs de Tunis (Tunisie)

AUTEUR : Nour Elhouda DOUGUI.

TITRE : Planification de trajectoires avion : approche par analogie lumineuse.

DIRECTEUR DE THESE : Marcel Mongeau et Daniel Delahaye.

LIEU ET DATE DE SOUTENANCE : Le 15 décembre 2011
à l'Ecole National de l'Aviation Civile (ENAC).

RESUME en français :

Dans le cadre du projet européen SESAR, la nécessité d'accroître la capacité du trafic aérien a motivé la planification de trajectoires avions 4D (espace + temps).

Afin de mettre en place une planification pré-tactique (évitement de zones avec une mauvaise météo ou congestionnées pour un avion) et de mettre en place une planification tactique (générer des ensembles de trajectoires 4D sans conflit), nous introduisons un nouvel algorithme : l'algorithme de propagation de la lumière (APL). Cet algorithme est basé sur une méthode de propagation de front d'onde qui s'inspire de l'analogie avec la propagation de la lumière et qui est adapté au problème de planification de trajectoires.

L'APL donne des résultats satisfaisant pour une journée de trafic réel sur la France tout en satisfaisant les contraintes spécifiques à la gestion du trafic aérien.

L'APL a ensuite été adapté pour prendre en compte les incertitudes qui concernent la vitesse réelle des avions. Ainsi adapté aux incertitude, l'APL a été testé sur la même journée de trafic avec mise en place de points RTA (Real Time Arrival). Les points RTA permettent de réduire l'incertitude dans le cas où l'APL n'arrive pas à résoudre les conflits. Les résultats obtenus sont très encourageants.

MOTS-CLES : Planification pré-tactique, Planification tactique Résolution de conflit, Propagation de la lumière, Gestion du trafic aérien, Branch and Bound, Propagation de front d'onde, Flight Management System (FMS), Gestion incertitude, Real time of arrival (RTA).

DISCIPLINE ADMINISTRATIVE : Mathématiques appliquées et Informatique

---

INTITULE ET ADRESSE DE L'UFR OU DU LABORATOIRE : MAIAA : Laboratoire de Mathématiques Appliquées Informatique et Automatique pour l'Aérien (École Nationale de l'Aviation Civile) et IMT : Institut de Mathématiques de Toulouse UMR 5219

Table des matières

Introduction

Dans ce chapitre introductif, on exposera la problématique générale traitée dans cette thèse, puis on présentera le contexte opérationnel, dans lequel elle s'inscrit, et on finira par le plan général de la thèse.

Problématique générale

C'est dans le cadre futuriste du projet européen SESAR, qu'on décrira par la suite, que cette thèse se situe. Son objectif est de produire un algorithme de planification pré-tactique (une ou deux heures à l'avance) et tactique (vingt minutes à l'avance) des trajectoires avion.

Dans la phase pré-tactique, on considère un espace aérien avec des zones critiques telles que des zones avec une mauvaise météo ou des zones congestionnées, c'est-à-dire des zones parcourues par un trafic non organisé, avec un nombre suffisamment important d'aéronefs. Le but de la planification pré-tactique est de rechercher, pour un avion donné dans cet espace, une trajectoire optimale évitant ces obstacles, en tenant compte d'une métrique donnée (temps, distance, consommation de carburant, etc.). Dans ce premier cadre, les obstacles seront considérés comme statiques.

Dans la phase tactique, on considère un ensemble d'avions en conflit [1], pour lesquels, on recherche de nouvelles trajectoires sans conflit. À l'inverse du cas pré-tactique, les obstacles à éviter (les autres avions) sont considérés dynamiques. De plus, ici la décision d'un avion influence la décision des autres.

Dans les deux cas, les modifications apportées aux trajectoires devront prendre en compte un ensemble de contraintes opérationnelles (respect des limites de vitesse, taux de virage limité, etc.).

On s'intéressera aussi dans cette étude, à la gestion de l'incertitude lors de la planification tactique des trajectoires. Les sources d'incertitudes sur les positions réelles des avions sont multiples (vent, température, etc.) et peuvent influencer fortement la détection et la résolution de conflit entre aéronefs en planification tactique. Dans le cadre de SESAR, afin de de gérer ces incertitudes, on verra les évolutions possibles du système actuel de gestion de vol, qui permet à l'avion de suivre sa trajectoire.

1. Rapprochement dangereux entre deux ou plusieurs aéronefs.

Contexte opérationnel

On exposera dans cette section, le contexte opérationnel dans lequel on va traiter les problématiques qu'on vient de décrire. On présentera le système actuel de gestion du trafic aérien, ses limites et les projets lancés en Europe et aux États-Unis pour pallier à ces limitations.

Contrôle du trafic aérien (ATM)

Tout au long de cette thèse, on utilisera l'acronyme ATM (*air traffic management*) pour faire référence au contrôle du trafic aérien. Lorsqu'un avion effectue un trajet entre deux aéroports, un plan de vol doit être déposé, afin d'en informer les services de la navigation aérienne. Ce dernier contient tous les éléments indicatifs décrivant le vol prévu, notamment :

- le type d'avion ;
- l'heure de départ ;
- le premier *niveau de vol* demandé pour la croisière. Le Niveau de vol est défini comme étant l'altitude lue sur un altimètre calé sur la surface isobare 1013 mb. Il est exprimé en centaines de pieds (ft [2]), ainsi une différence de 5000 pieds par rapport à cette surface produira un FL50 (*Flight Level*) ;
- l'équipement de bord ;
- la route prévue : elle est décrite par une série de balises.

Pour suivre son plan de vol, un avion utilise son système de gestion de vol (*Flight Management System*) ou FMS, dont on décrira le fonctionnement au paragraphe suivant.

Flight Management System (FMS)

Le FMS est un système embarqué de gestion de vol, qui intègre des informations sur la performance de l'avion et sur sa position, provenant de capteurs de navigation, du plan de vol stocké et des saisies manuelles. Son but est d'assister les pilotes, en fournissant des instructions de pilotage, ou de permettre un guidage automatique de l'avion sur sa trajectoire lorsqu'il est couplé avec un pilote automatique. Il assure ainsi un niveau de sécurité, de confort des passagers et de régularité des vols satisfaisant.

Afin de guider l'aéronef le long de son plan de vol, le FMS utilise différents capteurs pour déterminer la position de l'avion et la précision de cette position. Cette précision est définie comme le degré de conformité entre la position estimée, mesurée ou souhaitée et la position réelle de l'aéronef à un instant donné. Le FMS utilise un *filtre Kalman* [3] pour intégrer les positions provenant des différents capteurs en une seule position. Les capteurs les plus utilisés sont :

2. 1 ft = 0.3048m

3. Un filtre Kalman [61] est un outil statistique qui a pour but d'estimer l'état réel d'un système dynamique linéaire à partir de mesures observées contenant du bruit. Il s'agit d'un estimateur récursif : pour estimer l'état courant, seuls l'état précédent et les mesures actuelles sont nécessaires.

- Le récepteur GPS (*Global Positioning System*), système de navigation par satellite avec couverture mondiale, développé par le département de la défense des États-Unis. Ce récepteur est considéré comme le capteur principal, étant donné qu'il a la plus grande précision (moins de 100 m, 95% du temps) et la plus grande intégrité (c'est-à-dire la capacité du système à déterminer qu'il ne fonctionne pas correctement, et dans ce cas, à alerter l'utilisateur à temps).
- Les moyens de radionavigation (NAVAIDS), qui sont considérés comme les seconds meilleurs capteurs du point de vue précision et intégrité. Ils comprennent :
  - Le DME (*Distance Measuring Equipment*), qui est un radio-transpondeur (un dispositif électronique qui émet une réponse quand il reçoit une interrogation par radio) permettant de connaître la distance qui sépare un avion d'une station au sol en mesurant le temps que met une impulsion radioélectrique UHF (Ultra Haute Fréquence) pour faire un aller-retour. Le FMS vérifie les distances de l'avion par rapport à différentes stations simultanément afin de déterminer une position toutes les 10 secondes. La précision dépend du nombre de stations DME disponibles et de la configuration géographique par rapport à l'avion, mais elle est généralement inférieure 0,3 nautique (*Nautical miles*[4], noté Nm dans la suite).
  - Le récepteur VOR (*VHF omnidirectional radio range*) qui permet de déterminer un relèvement magnétique par rapport à une station au sol (balise émettrice VOR dont la position est connue), et donc le radial (position angulaire) sur lequel le récepteur (donc l'avion) est situé. La position de l'avion peut donc être déterminée avec uniquement deux stations VOR, mais la précision est limitée (environ 1Nm).
- La centrale inertielle (*Inertial reference systems*, IRS), qui utilise trois gyroscopes (un pour chaque axe), et trois accéléromètres, pour calculer la position de l'avion. Elle est très précise et indépendante de sources externes. Mais, les gyroscopes étant victimes d'une certaine dérive (2Nm/heure) sur de longues distances, on leur adjoint un système de guidage par GPS pour leur permettre de recaler leur positionnement.

Les avions sont en générale équipés avec plusieurs capteurs du même type afin d'augmenter la redondance de l'information.

Compte tenu du plan de vol et de la position de l'avion, le FMS calcule la route à suivre. Le FMS permet aussi de modifier le plan de vol, pendant le vol, pour diverses raisons comme un retard, de mauvaises conditions atmosphériques ou des contraintes ATM. Il permet ainsi d'insérer ou de modifier les procédures de départ (Standard Instrument Departure, SID) ou d'arrivée (Standard Terminal Arrival, STAR), insérer un point de report ou un circuit d'attente, changer de destination, etc. À partir des points du plan de vol, le FMS calcule une trajectoire latérale. À partir du niveau de croisière et des contraintes d'altitude, le FMS calcule un profil vertical. À partir des contraintes de vitesse et des vitesses optimisées de chaque phase de vol et en fonction du *cost index*[5] choisi par la compagnie aérienne, le FMS calcule un profil de vitesse.

Les étapes de calcul du profil de vitesse sont les suivantes :

4. 1Nm=1852m ou longueur d'une minute d'arc sur un grand cercle terrestre

5. le Cost Index (CI) est un paramètre du système de gestion de vol (FMS) qui représente le rapport entre le coût d'opération d'une heure de vol et le coût unitaire du carburant.

- pour le décollage (du lâcher des freins à l'altitude d'accélération) : calcul des vitesses d'exploitation (rentrée des becs et volets, etc),
- pour la montée (de l'altitude d'accélération à l'altitude de croisière) : calcul de la vitesse économique (CAS [6] puis Mach [7]) et du point de transition de la phase de montée vers la phase de croisière (*Top Of Climb*, TOC),
- pour la croisière (du TOC au point de début de descente (*Top Of Descent*, TOD)) : calcul de la vitesse économique (Mach) en fonction du niveau de croisière,
- pour la descente (du TOD au point de décélération) : calcul d'une vitesse économique (Mach puis CAS) correspondant à une poussée réduite et du profil d'altitude associé,
- pour l'approche (du point de décélération à l'atterrissage) : calcul du profil de décélération respectant les contraintes d'exploitation (suivi de la trajectoire, sortie des becs et volets) pour atteindre la vitesse d'approche finale.

Le pilote peut suivre la route calculée par le FMS manuellement, ou il peut régler le pilote automatique pour la suivre. Les modes de navigation du FMS sont : le mode de navigation latérale (LNAV) pour le plan de vol latéral, le mode de navigation verticale (VNAV) pour le plan de vol vertical et le mode en vitesse gérée (*Managed Speed Mode*) pour le suivi du profil de vitesse FMS.

Dans le mode de navigation latérale, le FMS calcule l'erreur sur la position qui est fonction de la précision des équipements utilisés, il calcule la vitesse sol (*Ground Speed*, GS) et les écarts par rapport à la route souhaitée (écart latéral et angulaire). Le FMS fournit alors des consignes de guidage en roulis et en lacet afin de minimiser ces écarts.

Dans le mode de navigation verticale (VNAV), les écarts par rapport à l'altitude souhaitée et à la pente souhaitée sont calculés. Le FMS fournit alors des consignes de guidage en tangage pour minimiser les écarts.

FIGURE 1 – Une partie d'un plan de vol affiché sur un écran de navigation

Dans le mode de navigation en vitesse gérée, le FMS fournit des consignes de poussée pour maintenir le profil de vitesse de l'avion. Dans ce mode, une exigence temporelle

6. Calibrated AirSpeed (CAS) est la vitesse indiquée par un indicateur conventionnelle de vitesse, après correction de l'erreur des instruments.

7. Le rapport de la vitesse de l'avion à la vitesse du son.

(*Required time of arrival*, RTA) peut être imposée à un point donné de la trajectoire de l'avion, par un contrôleur aérien. Dans ce cas, le FMS doit cibler ce point de la trajectoire et doit l'atteindre à l'instant RTA. Ceci est souvent utilisé dans la phase approche du vol, pour la planification des créneaux d'arrivée au niveau des aéroports. Le FMS régule alors la vitesse de croisière pour garantir la satisfaction du RTA.

Depuis le cockpit, le FMS est contrôlé par le pilote via un *Control Display Unit* (CDU). Il peut être utilisé pour modifier le plan de vol. Le FMS envoie également les informations du plan de vol à l'écran de navigation (*Navigation Display*, ND) ou à l'écran de vol primaire (*Primary Flight Display*, PFD) de la plateforme d'instruments de vol (*Electronic Flight Instrument System*, EFIS). Le plan de vol apparaît généralement sur le ND (voir figure 1) comme une ligne rouge, avec les aéroports, les moyens de radionavigation et les balises.

Tous les FMS contiennent une base de données de navigation (BDN). Elle est mise à jour tous les 28 jours, afin de s'assurer que son contenu est valide. La BDN contient des données, à partir desquelles le plan de vol est construit. Le format et la structure de ces données sont définies par la norme ARINC (*Aeronautical Radio, INC*) 424. Il s'agit des données suivantes :

- des données de circulation aérienne issues des publications de l'organisation de l'aviation civile internationale (OACI) ou des administrations nationales telles que les couloirs aériens (*Airways*), les moyens de radionavigation, les aéroports, les circuits d'attente, les procédures SID et STAR, etc.,
- des données de navigation propres à la compagnie aérienne : *routes compagnies* définies par un indicatif, l'aéroport de départ, l'aéroport de destination, la composition de la route (segments de montée, de croisière, de descente, d'approche), altitude de croisière, etc.,
- des données du constructeur : données aérodynamiques (trainée, portance et limites opérationnelles) et données moteur (consommation, poussée et vitesse économique).

Pour plus de détails sur le fonctionnement du FMS, on peut consulter plusieurs références [50, 93, 76].

Nous venons de voir comment on gère un plan de vol côté bord (avion). Dans les paragraphes suivants, nous verrons comment il est géré côté sol (les services de la navigation aérienne).

Planification

Les avions utilisent le plus souvent des routes pré-établies : les couloirs aériens. Ces couloirs sont des tubes de sections rectangulaires, entourant des segments de droites, aux intersections desquels sont situées des balises. C'est souvent autour de ces balises qu'apparaissent des risques de collisions appelés *conflits*. Un conflit est un événement dans lequel deux ou plusieurs avions perdent leurs séparations minimales. On définit pour cela une distance horizontale, exprimée en nautiques : *la séparation standard horizontale*, et une distance verticale, qui, elle, est exprimée en pieds : *la séparation standard verticale*. On dit que deux avions sont *séparés* quand la distance qui sépare leurs projections sur un plan horizontal est supérieure à la séparation standard horizontale *ou* quand la distance

qui sépare leurs projections sur un plan vertical est supérieure à la séparation standard verticale. Ainsi, il existe une zone (ou un volume) de protection autour de chaque avion dans lequel aucun autre avion ne doit pénétrer.

Le contrôle de la circulation aérienne organise les flux aériens afin d'assurer la sécurité des vols (en terme de risque de collision) et d'améliorer la capacité du réseau de routes sur lequel les avions se déplacent.

Suivant la nature du trafic, on distingue les trois types de contrôle suivants :

- *contrôle d'aérodrome* : gestion des phases de roulage, de décollage et d'atterrissage ;
- *contrôle d'approche* : gestion du trafic en étape préparatoire à l'atterrissage ou post-décollage dans une zone proche d'un aérodrome ;
- *contrôle en-Route* : il concerne essentiellement le trafic en croisière entre les aérodromes. Actuellement, l'OACI fixe la séparation standard horizontale à 5 Nm et la séparation standard verticale à 1000 ft pour le trafic en-Route.

À l'heure actuelle, on enregistre sur le territoire français environ 8000 mouvements par jour, ce qui représente une charge de contrôle impossible à gérer par un seul contrôleur. On répartit cette charge de travail en divisant l'espace aérien en plusieurs *secteurs*, pour chacun desquels on affecte une équipe de contrôleurs. Le nombre de secteurs est alors déterminé par la capacité d'un contrôleur à gérer N avions simultanément (dans la pratique la moyenne semble être $N = 10$ à 15 avions ; lorsque cette limite est atteinte on dit que le secteur est *saturé*). Un *centre de contrôle* regroupe un ensemble de secteurs sur une zone géographique donnée.

Les organismes du contrôle aérien sont responsables de l'écoulement du trafic dans l'espace dont ils ont la charge. Le service rendu aux usagers doit offrir une sécurité parfaite mais également le meilleur débit possible. À l'intérieur de chaque secteur, les contrôleurs maintiennent chaque avion séparé du reste du trafic en donnant aux pilotes des instructions que ces derniers sont tenus de suivre. Le *contrôleur organique* assure la coordination avec les secteurs voisins et fait de la pré-détection de conflits. Le *contrôleur radar* surveille le trafic, décide des mesures d'évitement nécessaires et communique avec les pilotes.

La planification de trajectoire est structurée suivant trois étapes principales :

1. *Planification stratégique.* Cette étape consiste à construire l'ensemble des trajectoires d'une journée de trafic à l'échelle d'un pays. Elle a pour but de structurer ces trajectoires, afin de satisfaire les contraintes de capacités secteurs tout en répondant au mieux aux demandes des compagnies aériennes. Cette planification est effectuée avant le décollage des avions et doit prendre en compte divers aléas (vent, piste utilisée à l'atterrissage, départ retardé, etc) et doit produire un schéma robuste.

2. *Planification pré-tactique.* Une fois que les avions ont décollé, cette étape permet de mettre à jour le résultat de la planifation stratégique en fonction des nouvelles informations disponibles (météo, congestion d'espace aérien). Elle a pour but de modifier les trajectoires des avions ainsi que leurs vitesses, afin d'éviter des zones à forte congestion ou avec une mauvaise météo. L'horizon d'action de cette planification se situe entre une à deux heures avant l'arrivée de l'avion dans une zone critique.

3. *Planification tactique.* Cette planification représente l'action du contrôleur sur son secteur. Le temps moyen passé par un avion dans un secteur est de l'ordre d'une vingtaine de minutes. La visibilité du contrôleur est légèrement supérieure puisqu'il dispose des plans de vol quelques minutes avant l'entrée de l'avion dans son secteur. Dans le cadre opérationnel actuel, la détection, et surtout la résolution des conflits, ne sont pas automatisées. Les contrôleurs sont donc entraînés à reconnaître des types de conflits et à appliquer à ces derniers des manœuvres typiques connues. Le contrôleur ne peut éviter les conflits qu'en jouant sur la trajectoire des avions. Pour cela, il dispose de quatre types d'ordre :

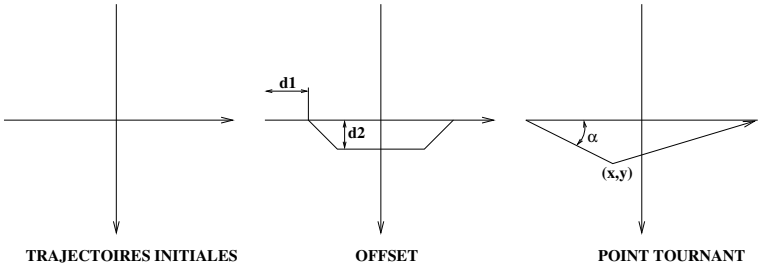


FIGURE 2 – Déviation de cap : offset et point tournant

   – des déviations de cap (*offset* ou *point tournant* illustrées par figure 2) ;
   – des paliers intermédiaires pour les avions en montée ou en descente ;
   – des changements de niveaux de vol ;
   – des mesures de régulation de vitesse (essentiellement sur des avions en descente).

   Dans le cadre du projet européen SESAR, décrit dans la suite, on envisage une résolution automatique de conflits, avec liaison de données air-sol. Dans ce contexte, il est possible d'envisager des manœuvres plus riches avec déformation continue de la trajectoire qu'un pilote humain ne pourrait mettre en œuvre.

Pour plus de détails sur les méthodes utilisées dans le contrôle du trafic aérien, on pourra consulter les références [105, 112, 71, 108, 23].

Limite du système actuel

Le système actuel de gestion du trafic aérien est très souvent en limite de capacité, dû à l'augmentation constante de la demande depuis de nombreuses années. Au fur et à mesure de l'augmentation du trafic, l'espace aérien a été divisé en secteurs de plus en plus petits, afin d'éviter la saturation de ces derniers. Malheureusement, ce principe de resectorisation présente une limite dans la mesure où l'on doit ménager un temps suffisant au contrôleur pour élaborer des stratégies de résolution des conflits entre aéronefs et l'espace nécessaire pour la mise en œuvre de ces stratégies. On doit donc générer des secteurs dont la taille permet de satisfaire cette contrainte. De plus, le contrôleur ne connaît que le trafic lié à son secteur et lorsqu'un avion passe d'un secteur à un autre, il s'opère un dialogue entre les contrôleurs et les pilotes, afin d'assurer la sécurité du vol lorsqu'il pénètre dans

le nouveau secteur. Ce dialogue induit une charge de travail supplémentaire pour les contrôleurs appelée *charge de coordination*. Ainsi, en augmentant le nombre des secteurs, on augmente aussi les charges de coordinations et donc la charge globale de contrôle. Lorsque l'on ne peut plus agir sur la capacité du système (en augmentant le nombre de secteurs ou en construisant de nouvelle infrastructures aéroportuaires), il est alors nécessaire de réguler la demande, afin que la charge de gestion induite reste toujours en deçà de la capacité. Dans le cadre du trafic aérien, ces régulations consistent à attribuer des retards au décollage ou à allouer de nouvelles routes aux avions, afin d'assurer le respect des capacités des secteurs. Dans certaines situations, les délais attribués peuvent être importants et les routes allouées peuvent être trop distantes des routes demandées par les compagnies aériennes. De plus, les études de prédiction de trafic prévoient un doublement du nombre d'avions au niveau européen à l'horizon de 2020 (18 millions de vols aux instruments [8] par an). Il est donc nécessaire d'envisager des améliorations majeures au niveau des concepts opérationnels, ainsi qu'au niveau des technologies. Dans ce contexte, le projet SESAR en Europe et le projet NextGen aux États-Unis ont été initiés, afin de répondre aux défis majeurs du futur système ATM. Ces deux projets sont l'objet des sections suivantes.

Le projet SESAR

Le programme SESAR (*Single European Sky ATM Research*) est l'un des programmes de recherche et de développement les plus ambitieux jamais lancé par la Commission européenne et Eurocontrol [9] (*European organisation for the safety of air navigation*).

La mission de SESAR est de développer un système modernisé de gestion du trafic aérien pour l'Europe. Ce futur système assurera la sécurité et la fluidité du transport aérien au cours des trente prochaines années. Il aura pour objectif d'augmenter la capacité du système ATM (à tous les niveaux), de réduire l'impact environnemental ainsi que les coûts induits par l'ATM. De plus, SESAR doit permettre l'augmentation de l'efficacité des opérations, afin de satisfaire les besoins et les priorités des utilisateurs ; il doit ainsi maximiser les capacités tout en minimisant les contraintes imposées aux utilisateurs (comme par exemple permettre aux avions de gérer eux même leurs vols sans l'intervention des contrôleurs) et les incertitudes dans le système.

D'autre part, ce projet a pour objectif la mise en œuvre du *ciel unique européen*, permettant d'unifier et d'harmoniser l'ensemble des systèmes ATM des états membres. Un aspect primordial de ce projet consiste à favoriser le partage de l'information entre les différents acteurs du système (compagnies aériennes, fournisseurs de services civils et militaires, etc.). La gestion de ce partage de l'information sera assurée par le système

8. Un vol aux instruments (*Instrument Flight Rules* ou IFR) respecte un certain nombre de règles lui permettant, avec l'aide de ses instruments de navigation et du contrôle aérien, de suivre sa trajectoire prévue.

9. Eurocontrol est une organisation intergouvernementale composée de 39 états membres de la communauté européenne. C'est une organisation civile et militaire qui a pour but de soutenir les améliorations apportées à l'ATM à travers l'Europe.

SWIM [10] qui permet la mise en œuvre de prises de décision collaboratives (CDM [11]). Le système SWIM sera basé sur des protocoles internet (liaisons de données entre le bord et les équipements sol). Dans ce contexte, les contrôleurs pourront transmettre des commandes plus complexes qu'un simple changement de cap ou changement de vitesse. Ainsi, des *trajectoires courbes* plus générales pourront être échangés entre le sol et le bord.

La trajectoire de référence (*Business Trajectory*, BT) [12], définie dans SESAR, vise à exécuter chaque vol aussi près que possible des intentions des compagnies aériennes. Les services de l'ATM assureront une gestion des BTs de façon sure et rentable tout en respectant les contraintes des infrastructures et de l'environnement. Le mécanisme de CDM assurera un changement minime des BTs sauf en cas de situation critique. Les BTs seront exprimées en 4D et seront suivies avec beaucoup plus de précision qu'aujourd'hui. Les besoins des militaires seront préservés dans ce nouveau système. Les utilisateurs ne seront plus astreints à rester sur des routes prédéfinies mais auront plus de libertés pour gérer leur navigation, sauf dans les espaces très congestionnés dans lesquels la notion de route structurée sera conservée.

En ce qui concerne la gestion des conflits, il est facile de démontrer que le risque de collisions entre aéronefs est proportionnel au carré du nombre d'avions présents dans une zone de l'espace aérien, quand on suppose une distribution aléatoire des positions et des orientations de trafic (flux non organisés). Or, si l'on souhaite doubler la demande d'ici 2020, il est important de développer de nouveaux outils de détection et de résolution de conflits afin d'assurer (voire d'augmenter) le niveau de sécurité actuel. Dans le cadre du projet SESAR, cette amélioration repose principalement sur l'augmentation des capacités de navigation et sur de nouvelles techniques de séparation. Pour cela, des modes d'opérations mixtes seront utilisés et il sera possible de prendre en compte plusieurs catégories d'avions en fonction de leur capacité de navigation. Les modes de séparation envisagés par SESAR, se répartissent en trois grandes catégories :

1. Les modes classiques tel qu'ils sont utilisés aujourd'hui, mais avec de meilleures données et de meilleurs outils afin d'améliorer les trajectoires et l'efficacité du réseau.
2. Les nouveaux modes au sol, qui comprendront :
   – Des *Precision Trajectory Clearances* (PTC) se basant sur les performances de navigation des avions, les contraintes de gestion, et le contrôle permanent de files d'attente. Une PTC est une autorisation donnée par un contrôleur à un avion, qui lui permet de passer sur une trajectoire 2D, 3D ou même 4D (3D+ temps). Une PTC-2D implique que l'avion devra suivre parfaitement sa trajectoire prévue dans le plan horizontal (sans exigence de précision sur le plan vertical). Une PTC-3D implique une précision dans les 3 dimensions. Une PTC-4D implique une précision sur la dimension supplémentaire du temps (ceci implique que la position

10. *System Wide Information Management* : un seul système d'information où toutes les parties prenantes sont reliées. Ce système contient toutes les données nécessaires pour permettre la prise de décision efficace basée sur des données temps réel.

11. *Collaborative Decision Making* : permet de partager les informations entre tous les acteurs ATM, avec des délais et une précision leur permettant de planifier les opérations en temps réel.

12. *Business Trajectory* : exprime les besoins spécifiques des utilisateurs de l'espace aérien, assure la sécurité des vols tout en minimisant les changements à apporter.

de l'avion soit connue à chaque instant). Ces autorisations sont soumises à un accord de l'équipage (pour des raisons de capacité de navigation de l'avion). Dans chaque PTC, l'avion devra maintenir sa trajectoire avec la précision convenue, permettant ainsi aux contrôleurs, aidés par des outils de prévision et de résolution de conflits ainsi que des moyens de surveillance des intentions, de gérer plus de trafic tout en gardant la charge de travail totale à des niveaux acceptables. La PTC a pour but d'assurer la résolution de conflit, tout en garantissant la flexibilité et la prédictibilité avec une intervention réduite des contrôleurs.
   - Un contrôle de vitesse. Il s'agit d'une méthode de résolution de conflit automatisée, qui impose un ajustement de la vitesse (horizontale et/ou verticale) dans une gamme limitée. Son but est de résoudre tactiquement des situations de trafic et de réduire la complexité et la charge de travail du contrôleur.
3. Les nouveaux modes de séparation à bord qui utilisent les applications ASAS [13] (*Airborne Separation Assistance Systems*) pour la séparation coopérative entre aéronefs. Dans ce cadre, la séparation est temporairement déléguée à l'équipage, pour assurer l'espacement avec les autres avions dans des circonstances spécifiques.

Les avantages de ces modes de séparation semblent être similaires. Dans les espaces aériens contrôlés, la séparation est assurée par les ANSP [14]. Toutefois, sous réserve de fournir un système sécurisé du point de vue ATM, l'auto-séparation sera autorisée. Ainsi, la séparation peut être déléguée au bord si des règles spécifiques sont respectées (un équipement avion approuvé et la qualification du pilote). L'un des objectifs du système ASAS est de permettre l'auto-séparation dans des modes d'opération mixtes : permettre aux vols auto-séparés et aux vols séparés par l'ANSP d'opérer dans le même espace aérien.

Il est de la responsabilité du pilote d'assurer la sécurité du vol dans un espace aérien non contrôlé, quand il demande à assurer la séparation ou quand il accepte d'effectuer une manœuvre ASAS. Par contre, lorsque l'avion pénètre dans un espace aérien où le service de séparation est fourni par l'ANSP ou lorsqu'il demande une séparation effectuée par le sol, le pilote doit négocier la BT avec l'ANSP.

Actuellement, la charge de travail du contrôleur est l'un des aspects limitant la capacité de l'ATM. Trois lignes d'action sont adoptées dans SESAR pour répondre à ce problème :

1. L'automatisation des tâches de routine des contrôleurs aidée par une meilleure méthode de saisie de données et une amélioration de leur gestion,
2. L'automatisation des supports de détection et de résolution de conflit ainsi que de la supervision (*monitoring*) qui consiste à vérifier que les avions suivent bien leurs trajectoires.
3. La réduction du besoin d'intervention tactique du contrôleur au travers de :
   - la réduction du nombre potentiel de conflits en utilisant des méthodes de planification de trajectoires à long terme,

13. ASAS est un système avion qui permet à l'équipage de maintenir la séparation de leur avion avec un ou plusieurs autres avions, et fournit les informations concernant le trafic environnant.

14. *Air Navigation Services Providers* : fournisseurs de services de navigation aérienne. En France, il s'agit de la Direction des Services de la navigation aérienne.

– La redistribution des tâches d'intervention tactique au pilote, quand cela est possible, via l'auto-séparation des avions.

Dans l'espace aérien contrôlé, en moyenne et haute densité, les modes de séparation ne seront pas tous déployés d'ici 2020. Les avions et les capacités des systèmes au sol nécessaires pour le contrat 4D (où l'avion prend la responsabilité de suivre sa trajectoire 4D) et l'auto-séparation ASAS ne devraient pas être disponibles pour la grande majorité des utilisateurs avant les années 2020-2025. Néanmoins, ils font partie de l'étude de faisabilité comme des moyens efficaces pour atteindre l'objectif de trois fois plus de capacité.

Dans les zones à faible densité, il est prévu que l'auto-séparation soit introduite avant 2020 dans les segments de vol où cela est possible et d'acquérir une expérience dans le cadre du processus de validation d'une plus large implémentation.

SESAR permettra le développement des systèmes ACAS (*Airborne Collision Avoidance System*) et STCA (*Short Term Conflict Alert*) pour que l'information partagée puisse être utilisée pour coordonner les avis de résolution et d'avertissement. Des logiques de détection totalement indépendantes doivent être présentes dans les différents systèmes, en utilisant des sources d'information distinctes mais le résultat des calculs est toujours partagé. Cela n'implique pas que les deux systèmes négocient les manœuvres de résolution.

Le projet SESAR permet aussi l'intégration totale des opérations aéroportuaires dans le processus de gestion des trajectoires en optimisant le roulage ainsi que le rendement des pistes. Les nuisances sonores ainsi que les émissions seront réduites par la mise en œuvre de profils continus de descente et de montée ainsi qu'une gestion optimale des files d'attentes au sol. En effet, pour le contrôle d'approche, le processus de gestion des arrivées assurera une séquence d'arrivée optimale, en imposant des temps d'arrivée contrôlés [15] (*Controlled Time of Arrival*, CTA) aux avions grâce à la fonctionnalité RTA des FMS, permettant ainsi une fusion organisée des flux d'arrivée avec des espacements optimisés.

La nouvelle architecture du système ATM sera modulaire avec une nouvelle approche de développement système. Tous les sous-systèmes ATM seront intégrés dans un même réseau de services. Pour plus de détails sur le projet SESAR on pourra consulter les références suivantes [97, 98, 99].

Le projet NextGen

Tout comme SESAR, mais dans le contexte américain, le projet NextGen a été mis en place pour faire face au doublement (voire triplement) du trafic aux États-Unis à l'horizon de 2020. Le système de transport aérien aux États-Unis se trouve aussi en limite de capacité surtout dans le Nord-Est du pays et ne pourra pas faire face, en l'état, à l'augmentation de la demande prévue dans les années 2020. Pour accomplir les objectifs du projet NextGen, des transformations devront donc avoir lieu. Ces transformations incluent la gestion du trafic aérien et des aéroports, afin d'assurer une meilleure efficacité et d'augmenter la sécurité. Les impacts environnementaux de l'aviation seront également pris en compte. De plus, l'augmentation de la capacité, de l'efficacité ainsi que la sécurité se feront en utilisant de nouvelles technologies. NextGen fournira donc plus de flexibilité

15. Une contrainte temporelle définie sur un point d'approche finale associé à une piste d'arrivée.

et plus d'information aux utilisateurs, tout en réduisant les interventions de l'ATM. Il permettra une liberté opérationnelle et adaptera les structures et les procédures gouvernementales aux besoins des utilisateurs en fournissant une large gamme de services. La capacité sera augmentée afin de satisfaire la demande, en investissant dans de nouvelles infrastructures, en mettant en œuvre des procédures plus efficaces et en minimisant les effets des contraintes opérationnelles comme la météo. Ainsi, NextGen anticipe les changements liés à l'augmentation de la demande en favorisant l'automatisation, la réduction des normes de séparation et la mise en œuvre de pistes supplémentaires. De nouvelles capacités comme les *tours virtuelles*[16] permettent l'expansion des services à un nombre plus important d'aéroports. NextGen sera donc un système flexible et réactif. D'autre part, dans NextGen une communication robuste des données remplace la communication vocale comme premier moyen de communication et le partage de l'information permet d'améliorer les prédictions de trajectoire en réduisant les incertitudes. Les utilisateurs du système deviennent des partenaires égaux avec les ANSPs dans la prise de décision opérationnelle. Les capacités des avions (incluant le système d'assistance à la séparation) sont exploitées pour maximiser le débit dans les aéroports et dans l'espace aérien. Afin d'assurer la sécurité dans un trafic dense, le processus de détection et de résolution de conflit sera automatisé dans NextGen. Cette automatisation se fera avec des algorithmes au sol ou à bord des avions. La réduction et la variabilité des normes de séparation introduites par NextGen impliquent l'automatisation des manœuvres de résolution mais le pilote conserve la responsabilité ultime d'assurer la sécurité de son appareil et doit donc approuver ces manœuvres. Afin d'assurer un débit aéroportuaire suffisant pour pouvoir contenir un trafic 2 à 3 fois plus important que celui d'aujourd'hui avec un effet environnemental minime, il y aura besoin de nouvelles procédures de séparation (par exemple des approches parallèles et convergentes permettant d'augmenter le débit des pistes proches les unes des autres). Une large catégorie d'opérations de séparation est déléguée par l'ANSP aux avions convenablement équipés. Ces opérations incluent les approches parallèles étroitement espacées, les couloirs de circulation, les opérations à haut débit faites dans la même direction, les trajectoires parallèles proches dans le trafic en-Route. L'auto-séparation sera aussi introduite dans NextGen. Parmi ses avantages, l'ANSP épargnera des ressources en ne fournissant pas les services de séparation ou de gestion de trajectoires associés à ce type de résolution et les utilisateurs obtiendront un routage aussi proche que possible de leur demande. Les premières mises en œuvre de l'auto-séparation seront effectuées dans les zones océaniques et dans les espaces aériens à très faible densité. La réduction des standards de séparations dépend de l'amélioration de la capacité à prédire et à communiquer les trajectoires. La délégation de séparation et l'auto-séparation vont requérir des preuves de sécurité incluant le hardware, le logiciel et l'interaction homme-machine dans des situations nominales et non-nominales. Les résultats des premières recherches sont encourageants mais les changements opérationnels et culturels majeurs requis constituent un défi d'envergure.

16. Les tours virtuelles sont utilisées dans les aéroports ne disposant pas de tour de contrôle. Le personnel y travaillant, rend des services pour plusieurs aéroports à partir d'un seul endroit physique. Ceci permet de réduire les coût liés à la construction de tours.

Les fonctionnalités qui seront assurées par NextGen sont les suivantes :
– L'accès à l'information par réseau,
– L'assimilation des données météo dans la prise de décision,
– Une sécurité adaptative et par couches,
– Une précision de navigation améliorée sur un vaste domaine,
– Des opérations basées sur les trajectoires 4D.

La capacité de travail des contrôleurs ne sera plus un facteur limitant, grâce à l'automatisation des outils qui fournissent des informations plus étendues et permettent l'amélioration de la prise de décision. D'autre part, le transfert de la responsabilité de séparation des contrôleurs aux membres d'équipage dans certaines zones de l'espace aérien, permettra aux contrôleurs de se concentrer sur la gestion des flux de trafic et non plus sur les vols individuels. Le système aéroportuaire est actuellement le point critique de la croissance du trafic aérien aux États Unis. Les nouvelles technologies et procédures vont améliorer l'accès aux aéroports permettant une meilleure utilisation des infrastructures existantes. Le composant de partage d'information *Net-Centricity* permettra d'assurer un transfert d'information robuste, efficace, en temps réel et en toute sécurité entre les différents utilisateurs du transport aérien. Une image météo commune et fiable sera fournie à une large gamme d'utilisateurs, afin d'aider à la prise de décision optimale. Le service de sécurité sera assuré par un système d'information des risques de sécurité. Les intérêts environnementaux seront assurés par la mise en œuvre d'un système de gestion environnemental intégré (*Environmental Management System*, EMS). Les technologies sont incorporées avant et pendant les opérations pour permettre la sélection de routes optimales par rapport au bruit, aux émissions chimiques, à la consommation de carburant, au coût et à l'efficacité de la route. Voir les références [5, 88] pour plus de précisions sur le projet NextGen.

Plan de la thèse

La thèse s'organise en quatre parties. Dans le chapitre 1, nous exposons de manière détaillée la problématique de planification de trajectoires avion. On présente ensuite un état de l'art des méthodes existantes pour traiter ce sujet, avec une discussion sur les avantages et les inconvénients de ces méthodes.

Dans le chapitre 2, nous proposons une modélisation mathématique du problème à traiter. Nous introduisons ensuite, un algorithme de résolution, que nous appelons algorithme de propagation de la lumière (APL), basé sur une analogie avec la théorie ondulatoire pour le calcul de géodésiques (plus court chemin en temps).

Dans le chapitre 3, nous appliquons l'algorithme APL, au contexte spécifique de la gestion de trafic aérien. Nous décrivons deux versions de l'APL, la première adaptée à la planification pré-tactique des trajectoires avion, et la deuxième adaptée à la planification tactique. Dans ce dernier cas, nous testons l'APL avec des données réelles d'une journée de trafic sur la France.

Dans le dernier chapitre de la thèse, nous nous intéressons à la prise en compte de l'incertitude sur la position des avions lors de la détection et de la résolution de conflits,

dans le cadre de la planification tactique. Dans le contexte du projet européen SESAR, différentes évolutions futures sont envisagées pour le FMS, afin de gérer l'incertitude en jeu. Nous modélisons donc cette incertitude et nous adaptons l'APL à cette modélisation.

Chapitre 1

Planification de trajectoires avion

Dans ce chapitre, nous présentons précisément le problème de planification de trajectoires avion, l'état de l'art des méthodes existantes pour traiter ce sujet, et nous finissons par une discussion sur les avantages et les inconvénients de ces méthodes.

1.1 Description du problème

L'objectif de cette thèse est de produire des algorithmes qui planifient des trajectoires d'avions dans les phases pré-tactique et tactique pour du trafic en-Route et de tester la viabilité pratique des approches développées.

Planification pré-tactique :

Dans cette phase, nous recherchons pour un objet mobile, dans notre cas un avion, le plus court chemin γ entre deux points (son point de départ, s et son point destination, d) dans $\mathbf{R}^3$ (l'espace aérien) où la *métrique* à optimiser peut être le temps, la distance, la consommation de carburant, etc. Dans cette thèse, la métrique utilisée sera le temps de parcours de la trajectoire.

De plus, la trajectoire recherchée doit satisfaire plusieurs contraintes. La première contrainte concerne la vitesse du mobile. Cette vitesse doit rester dans le domaine de vol de l'avion. Plus précisément, suivant les conditions météorologiques, température et pression, la vitesse de l'avion a une limite inférieure, en-deçà de laquelle il décroche. De plus, les contraintes aérodynamique auxquelles sont soumis les avions de ligne leur imposent une borne supérieure pour la vitesse (Mach 1), au-delà de laquelle l'avion devient instable. Une deuxième contrainte concerne les conditions initiale et finale de vitesse au niveau du point de départ et de destination. Par ailleurs, la courbure de la trajectoire recherchée doit être bornée, afin de garantir le confort des passagers. De fait, dans la pratique, les taux de changement de cap d'un avion sont limités. La dernière contrainte, que l'on va introduire, est spécifique à la planification pré-tactique. En effet, dans cette phase de planification, la trajectoire recherchée γ doit éviter, autant que possible, des zones où l'avion risque d'être ralenti, tel que les zones avec une mauvaise météo ou les zones avec un trafic congestionné. Dans cette thèse, on prendra comme hypothèse que ces zones sont statiques, relativement

au délai de planification imparti (entre une et deux heures à l'avance). Il s'agit donc de trouver une trajectoire γ qui évite un nombre k de sous-ensembles connexes prédéfinis de $\mathbf{R}^3$, notés Ω_i avec $i \in \{1, ..., k\}$. Cette dernière contrainte est une contrainte souple (qui pourrait être violée dans une certaine mesure). La trajectoire γ pourra ainsi traverser les sous-ensembles Ω_i au prix, par exemple, d'un terme de pénalité dans la fonction objectif. Une autre façon d'interpréter cette contrainte, est de considérer que les trajectoires sont plus longues à l'intérieur des sous-ensembles Ω_i, selon une métrique utilisée pour calculer la longueur du chemin. On définira donc dans la section 2.1 une telle métrique c de l'espace $\mathbf{R}^3$.

Soit $\gamma(s, d, u)$ la trajectoire d'un mobile, qui part du point s et atteint sa destination d, où u est l'abscisse curviligne avec

$$\gamma(s, d, u_s) = s \text{ et } \gamma(s, d, u_d) = d, \tag{1.1}$$

où u_s et u_d sont les abscisses curviligne de s et d. Le temps de parcours de la trajectoire, t, est donné par :

$$t(u) = t_s + u(t_d - t_s),$$

où t_s est le temps de départ du point s et t_d le temps auquel le mobile atteint la destination d.

La contrainte sur la vitesse associée à γ est modélisée par :

$$\|\frac{\partial\gamma(\vec{s}, \vec{d}, t(u))}{\partial t}\| \in]V_{min}, V_{max}[, \ \forall u \in [u_s, u_d], \tag{1.2}$$

où $\vec{V}_{min}$ est la vitesse de décrochage et $\vec{V}_{max}$ est Mach 1. Pour respecter les conditions initiale et finale sur la vitesse du mobile, les contraintes suivantes doivent être respectées :

$$\|\frac{\partial\gamma(s, d, t(u_s))}{\partial t}\| = V_s \text{ et } \|\frac{\partial\gamma(s, d, t(u_d))}{\partial t}\| = V_d, \tag{1.3}$$

où V_s et V_d sont les vitesses aux points s et d, respectivement.

De plus, la contrainte d'avoir une trajectoire lisse implique une contrainte sur la courbure, $K(\gamma, u)$, de la trajectoire γ à toute abscisse curviligne u. La courbure est définie par :

$$K(\gamma, u) = \frac{\|\gamma'(u) \wedge \gamma''(u)\|}{\|\gamma'(u)\|^3},$$

où $\wedge$ représente le produit vectoriel. On impose une borne supérieure K_{max} à cette courbure :

$$|K(\gamma, u)| \leq K_{max} \ \forall u \in [u_s, u_d]. \tag{1.4}$$

Le problème de planification pré-tactique consiste donc à trouver une trajectoire γ qui minimise la fonction objectif f donnée par :

$$f(\gamma) = \int_{u=u_s}^{u=u_d} c(\gamma(\vec{s}, \vec{d}, u))du, \tag{1.5}$$

où $c(\gamma(s,d,u))$ est le coût du parcours d'une portion élémentaire de l'espace du de la trajectoire γ selon la métrique c. De plus, γ doit satisfaire les contraintes (1.1), (1.2), (1.3) et (1.4). Les données du problème sont s, d, t_s, t_d, v_s, v_d, V_{min}, V_{max} et les zones Ω_i à éviter.

Planification tactique :

Dans la phase de planification tactique, il ne s'agit plus, pour un avion, de tenter d'éviter une zone indésirable, mais pour un ensemble d'avions d'éviter d'être en conflit, les uns avec les autres. L'horizon d'action de la planification tactique est de l'ordre d'une vingtaine de minutes. Pour tous les avions en vol pendant cet horizon temporel, la relation "est en conflit avec", c'est-à-dire perte de séparation minimale, définit une relation d'équivalence et les classes d'équivalences associées sont appelées des *clusters* d'avions en conflit.

Pour un cluster contenant n avions en conflit, il faut trouver, pour chaque avion a_j, appartenant au cluster, avec $j \in \{1,...,n\}$, une nouvelle trajectoire γ_j qui ne génère pas de conflit avec les autres trajectoires γ_k, $k \neq j$ et $k \in \{1,...,n\}$, sur l'horizon temporel considéré. Pour cela, à chaque instant t, l'avion a_j ne doit pas se trouver, à l'intérieur du *volume de protection* d'un autre avion. Le volume de protection d'un avion k est un cylindre qui entoure la position de l'avion, de hauteur deux fois la norme verticale de séparation et dont la base est horizontale et a un rayon égale à la norme horizontale de séparation, comme le montre la figure 1.1. Comme notre problème concerne du trafic en-Route, ces normes sont respectivement de 1000 ft et de 5 Nm. Contrairement au cas précédent, où la contrainte sur l'évitement des sous-ensembles Ω_i de $\mathbf{R}^3$, était une contrainte souple, dans ce second problème la contrainte d'évitement des volumes de protection des autres avions est une contrainte dure. Si on reprend l'expression mathématique du problème précédent, la fonction métrique c doit ici être définie de façon à prendre une valeur infinie, à l'intérieur de ces volumes de protection. De plus, comme les obstacles, c'est-à-dire les autres avions, sont dynamiques, c n'est plus définie sur $\mathbf{R}^3$, mais dépend aussi du temps et associe à chaque point de $\mathbf{R}^3 \times \mathbf{R}^+$ (espace $\times$ temps) une valeur coût.

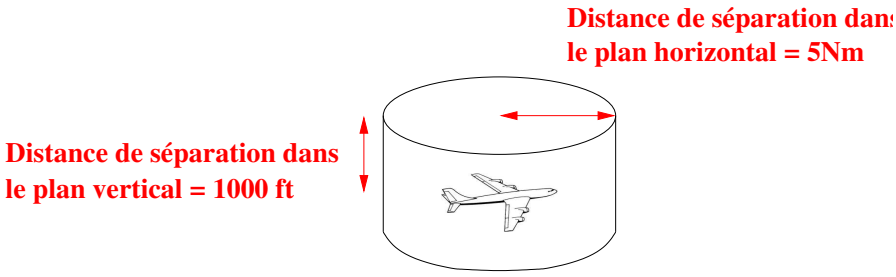


FIGURE 1.1 – Volume de protection d'un avion à un instant t.

Les autres contraintes opérationnelles, concernant la vitesse, la courbure de la trajectoire et les conditions initiales et finales, restent les mêmes que pour la planification pré-tactique. Les équations (1.1), (1.2), (1.3) et (1.4) restent donc valides, pour chaque trajectoire recherchée γ_j. La fonction objectif aussi reste valide, avec la nouvelle fonction

métrique c qui s'exprime dans $\mathbf{R}^3 \times \mathbf{R}^+$.

Ce second problème est beaucoup plus difficile que le premier, car outre le fait que les obstacles à éviter soient ici dynamique, cette dynamique n'est pas connue à l'avance. Effectivement, elle n'est pas indépendante de l'action du solveur, comme c'est le cas pour une zone avec une mauvaise météo dont la position est une entrée du problème. Ainsi, le choix de déplacement d'un avion influe sur le choix des autres, d'où l'apparition du problème de coordination d'actions entre les avions. Si l'objectif est la minimisation du coût global, ce problème est NP (à très forte combinatoire) comme montré dans [1].

Dans cette thèse, on se contentera de chercher une solution sous-optimale, en imposant à l'avance un ordre de priorité entre les avions. Nos résultats expérimentaux montreront que cette stratégie nous permet de trouver des solutions satisfaisantes à ce problème NP.

Gestion de l'incertitude :

Afin de traiter une situation réaliste de trafic, on doit prendre en compte l'incertitude sur les positions réelles des avions. On considère qu'avec les capacités actuelles du FMS, l'avion peut correctement suivre sa trajectoire 3D prévue. Mais une erreur sur la norme de la vitesse d'un avion de ± 15 Kt (*nœuds*[1]) doit être prise en compte. Cette erreur provient d'erreurs sur les données de vent et de température réelles. Une prise en compte de cette erreur de vitesse peut faire augmenter fortement le nombre de conflits détectés, par rapport au cas où on ignore cette incertitude. En conséquence, la résolution de tous les conflits supplémentaires risque d'être beaucoup plus difficile.

1.2 État de l'art

Dans cette section, nous présentons d'abord les grands projets historiques qui ont été lancés dans le but d'améliorer la gestion du trafic aérien. Ensuite, nous donnerons une classification des méthodes de détection et de résolution automatiques de conflits. Enfin, nous présenterons les principales approches existantes pour la décongestion (qui correspond à la planification pré-tactique) et pour la résolution de conflit (qui correspond à la planification tactique).

1.2.1 Historique

Durant ces dernières décennies, plusieurs projets ont été lancés pour tenter d'automatiser la détection et la résolution de conflits aériens. Nous résumons ici les cinq principales approches présentées dans [77, 47].

Automated En-Route Air Traffic Control (AERA)

Il s'agit d'un projet américain lancé par l'administration Reagan après la grève des contrôleurs de 1981. Ce projet a été financé par la FAA (*Federal Aviation Administration*) à hauteur de deux milliards de dollars mais n'a pas permis d'atteindre ses objectifs. Ce

1. 1 nœud = 1 Nm/heure = 1852m/heure.

projet traite le problème de planification tactique de trajectoires. Sa première phase, AERA 1, permet de prédire les trajectoires des avions en fonction des intentions du pilote et de détecter d'éventuelles violations des normes de séparation. AERA 1 ne propose donc pas de solution aux opérateurs. La deuxième phase, AERA 2, propose une liste de résolutions associées aux conflits détectés par AERA 1 ; néanmoins la responsabilité de séparer les avions revient au contrôleur. Dans AERA 3, la responsabilité de séparer les avions est laissée à un système automatique centralisé mais seuls les cas de résolution entre *paires* d'avions sont traités. Cette partie est assurée par un algorithme appelé *Gentle-Strict*. Cet algorithme ne résout les conflits que dans le plan horizontal et ne change pas les vitesses des avions ni leurs niveaux de vol. Il agit en utilisant des manœuvres latérales le plus tard possible afin d'éviter les manœuvres inutiles. Cependant, ce projet n'apporte pas de réponse au problème dès que le nombre d'avions impliqués dans un conflit dépasse trois. On pourra consulter les références [17, 86] pour plus d'information sur le projet AERA.

ARC2000

Le projet d'automatisation complète du contrôle en route européen lancé par Eurocontrol, ARC2000, se base sur la modélisation en tubes 4D des trajectoires avion dans l'espace-temps. Pour séparer les avions, il faut que les tubes 4D entourant la trajectoire de chaque avion ne s'intersectent pas. Pour cela, ARC2000 utilise un algorithme d'optimisation locale, type gradient, qui permet, étant donnés n tubes d'intersection vide, de construire un $n+1^{\text{ième}}$ tube, d'intersection vide avec tous les précédents, qui minimise le retard de l'avion $n+1$ correspondant. Un tube déjà affecté n'est pas remis en cause. Il s'agit donc d'un algorithme glouton où l'optimalité globale n'est pas recherchée. Initialement, le projet ARC2000 avait pour objectif de prévoir les trajectoires des avions dans leur totalité avec le minimum de conflits (en optimisant globalement), tout en surveillant les rapprochements potentiels. Les conflits sont éliminés 20 ou 30 minutes à l'avance. Ces ambitions ont été révisées à la baisse et abandonnées pour une gestion au niveau des *clusters* (ensemble d'avions en conflit). Dans la suite du projet, les critères utilisés pour la recherche de solutions optimales ont évolué en remplaçant le principe du "dernier arrivé, dernier servi" par une série de règles classant d'emblée les avions par ordre de priorité. On trouve une description plus détaillée du projet dans [39].

Le projet SAINTEX

Le projet SAINTEX [4] a été lancé par le Centre d'Études de la Navigation Aérienne (CENA) au début des années 1990. Dans ce projet, trois approches d'automatisation du contrôle en-Route ont été abordées :

– L'approche " Détection-Résolution " qui consiste à détecter les conflits en extrapolant les trajectoires avions 10 minutes dans le futur. Le conflit est ensuite classé et pour chaque classe de conflit, une manœuvre d'une classe prédéfinie de manœuvres est appliquée.

- Le scénario 4D : les trajectoires avions sont représentées par des tubes 4D dans l'espace-temps. Pour construire un tube admissible, diverses tentatives sont faites en partant de la trajectoire idéale (directe mais ignorant les conflits) vers des trajectoires de plus en plus pénalisées mais qui tiennent compte des conflits. L'algorithme utilisé ressemble fortement à celui utilisé dans ARC2000.
- Le scénario hybride dans lequel les avions *stables* en altitude sont gérés par le système détection-résolution et les avions *évolutifs*, c'est-à-dire soit en montée soit en descente, sont gérés par le système 4D.

Le projet SAINTEX ne s'intéresse qu'à la gestion d'espaces aériens peu saturés. Le problème de résolution de clusters d'avions n'est pas abordé.

Le projet FREER (*Free-Route Experimental Encounter Resolution*)

Ce projet est né en 1995 à Eurocontrol Brétigny. Il a connu plusieurs évolutions. Le modèle défini dans la phase 1 du projet FREER (FREER 1 [32]) considère que les avions sont totalement autonomes dans un espace à faible densité. L'espace concerné est le FFA (*Free Flight Airspace*) dans lequel on utilise les "*règles de l'air étendues*" [33], qui sont une extension des règles de l'air utilisées en vol à vue. Ces règles de l'air étendues permettent de prendre en compte toutes les configurations de conflit à deux avions d'une part, et d'autre part de définir un ordre total sur l'ensemble des avions. La prévision et la résolution de conflit se fait dans un horizon de 6 à 8 minutes mais les conflits ne doivent pas impliquer plus de quatre avions sinon les algorithmes embarqués ne garantissent plus la résolution de conflits. Ces algorithmes sont fondés sur HIPS [2] qui est un outil permettant de visualiser les *NoGoZones* (zones interdites) lorsque l'on fait évoluer les trajectoires des avions. Dans FREER 2 [31], on délègue partiellement la résolution de conflits élémentaires (impliquant deux avions) dans des espaces à trafic plus denses. Les avions doivent être ici équipés de systèmes ASAS (*Airborne Separation Assurance Systems*). Cependant, dans ce cas la délégation au système bord reste très partielle et les concepts introduits sont peu novateurs.

Le projet ERASMUS (*En Route Air Traffic Soft Management Ultimate System*)

Erasmus est un projet financé par l'Union Européenne dans la cadre du FP6 (*Sixth Framework Programme*) puis rattaché au programme SESAR. Son objectif est de déléguer une partie de la gestion des conflits à un système automatique d'une manière qui s'intègre bien dans la tâche du pilote et celle du contrôleur aérien. Il vise à proposer un concept opérationnel cohérent basé sur un nouveau mode de séparation (contrôle de la trajectoire par la régulation de vitesse) tout en tirant profit de la haute précision des trajectoires 4D.

2. Highly Interactive Problem Solver : cet outil se compose de deux éléments : un jeu d'écrans qui donnent une image complète de la situation de trafic aérien et un éditeur de trajectoires d'aéronefs. Le système n'est pas un dispositif autonome mais plutôt un instrument interactif d'aide à la décision à l'usage du contrôleur [40].

Pour la régulation en vitesse [6], le modèle d'incertitude adopté est basé sur les hypothèses suivantes :

- En croisière, l'altitude de l'avion est stable, entre deux balises le cap est stable, et enfin l'incertitude concerne surtout la vitesse horizontale suivant le cap de l'avion.
- Dans la phase de montée et de descente, l'incertitude concerne surtout la vitesse verticale qui dépend des conditions de vol et de la masse de l'avion.

Ces incertitudes peuvent être modélisées de façon relative, en affectant un pourcentage d'erreur e sur la vitesse nominale de l'avion (sa vitesse prévue). La vitesse réelle de l'avion se trouve alors incluse dans un intervalle d'incertitude : $[(1-e)v, (1+e)v]$ où v est la vitesse nominale de l'avion.

Le modèle considéré sera donc le suivant (figure 1.2). Dans le plan horizontal, l'avion est représenté par un segment dont les extrémités sont les positions déterminées par la vitesse minimale et la vitesse maximale. Ces deux positions extrêmes suivent le plan de vol, et au niveau d'une balise l'avion peut être représenté par un ou plusieurs segments, puisque l'avion peut changer de cap au niveau d'une balise et donc suivre un nouveau segment. Dans le plan vertical, l'avion est représenté par une altitude maximale et une altitude minimale déduite de l'altitude initiale et de l'intervalle d'incertitude pour la vitesse verticale.

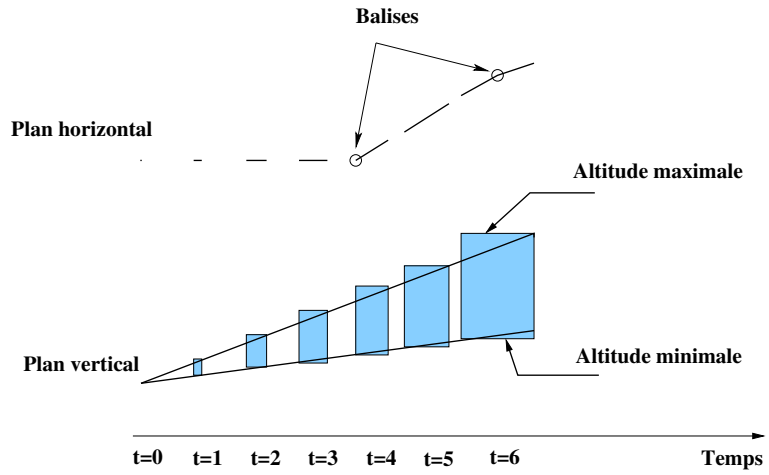


FIGURE 1.2 – Représentation de l'incertitude sur la position de l'avion dans le plan horizontal et vertical

L'évaluation de l'influence de l'incertitude sur la vitesse a été effectuée avec un simulateur arithmétique (CATS) [3]. Les résultats sont fortement dépendants de l'amplitude de l'erreur relative e utilisée. En effet, 1% d'erreur sur la vitesse (horizontale et verticale) conduit à plus de 30% de fausses alarmes (détection de faux conflits), mais avec 20%

3. Complete Air Traffic Simulator développé à la Direction des Services de la Navigation Aérienne : Direction de la Technique et de l'Innovation (DSNA/DTI). Il s'agit d'un simulateur de trafic basé sur un modèle tabulé de performances avion utilisant des données de plans de vol réels.

d'erreur sur la vitesse verticale et 5% d'erreur sur la vitesse horizontale, on obtient plus de 200% de fausses alarmes soit deux fois plus de fausses alarmes que de vraies.

Pour la résolution de conflits, CATS envisage 3 types de manœuvres qui correspondent à des manœuvres utilisées par les contrôleurs aériens :

- Des manœuvres de changement de cap, dites *offset* (figure 2), dans le plan horizontal,
- Dans le plan vertical, l'avion peut descendre à un niveau de vol moins élevé à l'instant t_0, pour ensuite commencer à remonter à un instant t_1,
- Des manœuvres de régulation en vitesse qui peuvent être modélisées avec deux temps : t_0 (temps de début de manœuvre) et t_1 (temps de fin de manœuvre) ainsi que l'amplitude de la manœuvre qui peut être une augmentation ou une réduction de la vitesse.

Les contraintes du simulateur CATS sont les suivantes. Une manœuvre peut être exécutée, soit dans le plan horizontal, soit dans le plan vertical, mais pas dans les deux. Un avion ne peut pas commencer une manœuvre, avant la fin de l'exécution de la manœuvre précédente.

CATS utilise deux solveurs de conflits :

- un solveur basé sur un algorithme génétique [2]. Le processus de résolution est ici appliqué individuellement à chaque cluster de conflit. Dans cette implémentation particulière de l'algorithme génétique, un individu ou chromosome représente l'ensemble des manœuvres appliquées à chaque avion dans le cluster. Pour mieux comprendre le fonctionnement des algorithmes génétiques, on peut consulter l'Annexe A.
- un solveur séquentiel distribué [52] qui consiste à rechercher l'optimalité dans une arborescence et qui est basé sur un ordonnancement a priori des avions. L'ordonnancement est obtenu par une stratégie de répartition de jetons utilisant une relation d'ordre total de priorité entre les différents avions impliqués dans le cluster de conflits (cet ordre peut être obtenu avec le code transpondeur des avions). Chaque avion reçoit un jeton de la part de tous les avions qui sont en conflit avec lui et qui ont une priorité supérieure à la sienne. Chaque avion impliqué dans un conflit qui n'a pas de jeton, effectue une résolution sans prendre en compte les avions qui ont un jeton ou plus. Une fois l'ordre de résolution établi, le problème revient à minimiser la longueur de la trajectoire d'un avion qui évite les n trajectoires déjà existantes. Ce problème est résolu à l'aide de l'algorithme A* [85] qui cherche la solution en parcourant un arbre de recherche à l'aide d'une heuristique. Chaque branche de cet arbre représente une trajectoire potentielle pour l'avion en cours.

Le premier solveur effectue une optimisation globale non déterministe et le second optimise localement et de façon déterministe.

L'intérêt des manœuvres par changement de vitesse est qu'elles ont peu d'impact sur les conditions du vol. En effet, elles ne modifient pas la trajectoire 3D et n'agissent donc que très peu sur l'image globale du trafic. De plus, ces manœuvres sont extrêmement efficaces dans les conflits de dépassement et leur exécution est facilement déléguée au FMS.

À travers ce parcours historique des principaux projets lancés dans le but d'améliorer la gestion du trafic aérien, on a vu que diverses méthodes ont été proposées, pour la détection et la résolution de conflits. Dans la section suivante, nous allons tenter de classifier ces méthodes en se basant sur les traveaux de Kuchar et Yang [67] et les traveaux de Chaloulos et Lygeros [19].

1.2.2 Classification des méthodes de détection et de résolution de conflits (CD&R)

Comme nous l'avons déjà vu, il y a un besoin de développement de nouvelles méthodes de détection et de résolution de conflit aérien, afin d'assister l'opérateur humain dans la gestion de la charge de trafic et l'amélioration de l'écoulement des flux. Le but des systèmes CD&R (Conflict Detection and Resolution) est de prédire l'occurrence d'un conflit, de communiquer la détection à un opérateur humain (ou à un système de bord) et, dans certains cas, de l'assister dans la résolution. Ces systèmes commencent par collecter des informations sur l'état courant de la situation de trafic, au travers des capteurs dont l'imperfection induit des incertitudes. Puis, un modèle dynamique de trajectoire est requis pour prédire l'état du système dans le futur, afin de détecter l'occurrence des conflits. La détection de conflits peut être vue comme le processus visant à déterminer "où" et "quand" une action doit être prise et la résolution consiste à déterminer "comment" une action doit être réalisée. En pratique, la séparation entre la détection et la résolution de conflit n'est cependant pas claire, car décider quand une action doit avoir lieu peut dépendre du type d'action qui sera mise en œuvre.

La différence entre les diverses approches de détection de conflits est basée sur la façon dont l'état courant est projeté dans le futur. Trois méthodes d'extrapolation ont été définies : nominale, pire cas et probabiliste.

La méthode nominale consiste à extrapoler la position de l'avion en se basant sur son vecteur vitesse courant. Cette méthode peut être assez précise quand la projection se fait sur quelques secondes dans le futur.

L'autre extrême de la modélisation dynamique est la projection du pire cas. Ici, on considère qu'un avion peut effectuer toute une gamme de manœuvres. Cette méthode est pessimiste et peut réduire considérablement la capacité du trafic à cause du grand nombre de fausses alarmes qu'elle engendre.

Dans la méthode probabiliste, les incertitudes sont modélisées pour décrire les variations potentielles de la trajectoire future. Cette approche consiste à développer un ensemble complet de trajectoires futures possibles, chacune ayant une probabilité d'occurrence (en utilisant des fonctions de densité de probabilité). On détermine ensuite les probabilités de conflit associées. L'avantage de cette méthode est que la décision de résolution peut être prise en se basant sur la probabilité d'un conflit, ainsi la sécurité et le taux d'alarmes peuvent être considérés directement. Par contre, on peut rencontrer des difficultés pour modéliser la probabilité avec laquelle les trajectoires futures seront suivies
.

Il existe différentes approches pour la résolution de conflits : prescrite, optimisée,

champs de force et manuelle.

Les manœuvres de résolution prescrites sont fixées pendant la phase de conception du système, basées sur un ensemble de procédures prédéfinies. Les opérateurs peuvent alors s'entraîner sur ces manœuvres types, ce qui réduit le temps de réponse en cas de conflit, mais elles sont moins efficaces que des manœuvres sur mesure calculées en temps réel puisqu'il n'y a pas de possibilité d'adapter la résolution.

L'approche par optimisation combine un modèle cinématique et un ensemble de métriques de coût. Une stratégie de résolution optimale induit alors un coût minimum. Par exemple, le TCAS, système de résolution de conflit dont on détaillera le fonctionnement par la suite, détermine la manœuvre la moins coûteuse qui assure une protection adéquate, à partir d'un ensemble prédéfini de manœuvres de montée ou de descente. Les systèmes de résolution de conflit par optimisation utilisent des techniques basées sur la théorie des jeux [107, 25], les algorithmes génétiques, les systèmes experts [60], le contrôle flou [48], le contrôle optimal [104, 14], etc.

L'approche par champs de forces considère chaque avion comme une particule chargée et utilise des équations électromagnétiques pour générer des manœuvres de résolution. Ces méthodes sont intéressantes, car elles garantissent en permanence une solution au problème mais font l'hypothèse que l'avion peut continuellement faire des manœuvres ou qu'il peut modifier sa vitesse sur un large intervalle, ce qui ne correspond pas à la réalité, surtout en altitude.

D'autres modèles, appelés modèles manuels, permettent à l'utilisateur de proposer des solutions et le modèle lui donne un compte-rendu lui indiquant si l'essai est acceptable. Ces modèles sont basés sur l'intuition humaine.

Pour gérer un cluster de conflits, on peut : soit considérer séquentiellement des paires d'avions, soit avoir une approche globale. Dans l'approche par paires d'avions, une solution de résolution de conflit induisant un nouveau conflit doit être modifiée jusqu'à l'obtention d'une situation sans conflit. Cette approche est suivie par le système TCAS. Elle est efficace mais peut être incapable de trouver une solution dans le cas de la résolution tactique d'un cluster. Tout modèle doit donc être examiné dans un contexte de conflits multiples, afin de vérifier sa robustesse face à ce genre de situation.

L'approche fondamentale utilisée dans la conception de systèmes CD&R se base sur un modèle dynamique et déterministe de trajectoire ainsi que sur un ensemble de métriques et de seuils d'alerte. Le système est ensuite exposé à un large ensemble de situations de trafic et le nombre de fausses alarmes résultant ainsi que le nombre de pertes de séparation sont enregistrés. Des incertitudes peuvent être modélisées et injectées dans la conception du système, afin de vérifier sa robustesse. Les paramètres du système sont ajustés, par une procédure d'essais et erreurs, aux situations auxquelles il est exposé. Idéalement, le système devrait adapter de façon continue son modèle dynamique ainsi que ses seuils d'alerte aux situations spécifiques auxquelles il est confronté. Une approche plus directe consisterait à utiliser les informations contenues dans les simulations ainsi que dans les scénarios d'évaluation, afin de construire un modèle de trajectoire probabiliste pour le système CD&R. Le but de cette approche est de disposer des meilleures informations décrivant l'environnement dans lequel le système va opérer. Une telle approche requiert, par

contre, une estimation en temps réel des probabilités et doit utiliser un modèle probabiliste de trajectoire viable. L'estimation des probabilités peut être faite de façon analytique ou en utilisant une simulation de Monte Carlo [74].

D'autre part, les méthodes de détection et de résolution de conflit peuvent être classées en deux catégories : centralisées (au sol) ou décentralisées (à bord). Les différences majeures entre ces approches sont les suivantes : le système sol a l'autorité de décider quel avion va manœuvrer et comment, alors qu'un système embarqué va manœuvrer uniquement son propre avion, de façon indépendante. Un système embarqué cherchera à optimiser le vol courant et un système sol cherchera à atteindre un optimum global (prenant en compte les différents vols impliqués). De plus, le système embarqué a accès aux informations les plus récentes de la situation locale (vent local, température, trafic local), alors que le sol a accès aux informations globales : météo à grande échelle (mais imprécise) et trafic dans sa totalité.

Les manœuvres typiques utilisées dans les méthodes actuelles de résolution de conflit (CR) interviennent dans le plan horizontal ou dans le plan vertical, elles peuvent impliquer des changements de vitesse ou encore elles peuvent mettre en œuvre des actions 3D.

On dit que les méthodes CR sont *coopératives* si les avions en conflit coordonnent leurs manœuvres ou *non-coopératives* si chaque avion résout le problème indépendamment des autres avions. Les méthodes non-coopératives entraînent en général des manœuvres excessives (inutilement coûteuses).

Les méthodes CD&R se placent dans un schéma plus large de méthodes de planification de trajectoire qui se différencient par leurs horizons temporels d'action. Certaines méthodes agissent à *long terme*, avec un horizon temporel de quelques heures. Elles correspondent à la planification pré-tactique des trajectoires et sont basées sur une description macroscopique des vols (plans de vol) et de l'environnement (capacités aéroports et secteurs). Il s'agit ici, plus de décongestion que de résolution de conflit à proprement parler. Elles peuvent aussi agir à *moyen terme*, c'est-à-dire sur un horizon temporel de 15 à 30 minutes, ce qui correspond à la planification tactique des trajectoires. Dans ce cas, elles sont basées sur des informations plus actualisées. Finalement, elles peuvent travailler à *court terme*, en utilisant une extrapolation des données courantes sur un horizon de 1 à 3 minutes. Il s'agit là, de méthodes CD&R d'urgence, utilisées en dernier recours quand toutes les planifications précédentes ont échoué dans la résolution des conflits. Dans les paragraphes suivants, nous décrivons plus en détails chacune de ces catégories de méthodes CD&R.

Les méthodes de décongestion à long terme

Ces méthodes, appelées aussi Air Traffic Flow Management (ATFM), se réfèrent à la construction de plans de vol (routes et heures de départ) de façon à ce que la probabilité d'occurrence de conflit dans les heures suivantes soit minimisée. Comme le niveau d'incertitude est assez élevé pour de tels horizons temporels, il ne s'agit pas vraiment de résoudre les conflits mais de gérer des problèmes de congestion. La congestion survient chaque fois que la capacité des aéroports ou des secteurs est dépassée. On recherche alors, pour un ensemble d'avions (à l'échelle d'un pays), un horaire de départ et une route permettant

de minimiser la congestion sur l'ensemble de l'espace aérien. Ces problèmes dits de *bi-allocation* (routes et heures de départ) sont des problèmes fortement combinatoires. Ainsi, dans la littérature, ces problèmes ne sont souvent traités que partiellement. La plupart des approches exactes consistent à ne traiter que l'allocation de délais [4], sans prendre en compte conjointement l'optimisation des routes.

Plusieurs modélisations de systèmes ATFM existent : les modèles *stochastiques* (respectivement *déterministes*) considèrent les capacités systèmes (aéroport ou secteurs) comme probabilistes (respectivement déterministes). On parle de modèles *dynamiques* ou *statiques* selon que l'on considère que les solutions sont adaptées dynamiquement dans le temps ou non. Plusieurs modélisations co-existent pour les problèmes de TFM généralisés. Il y a, d'une part, les modèles qui tentent de trouver une solution optimale au problème comme l'approche de Bertsimas et Stock Pattersson [13], le modèle d'allocation temporelle [69], le réseau spatio-temporel [53], les inéquations variationnelles [45] et le contrôle optimal [46]. Il y a, d'autre part, le modélisation visant des solutions approximatives comme le programmateur multi-aéroports [38] et le *Computer Assisted Slot Allocation* (CASA) [87].

Dans la section 1.2.3, on verra plus en détails quelques méthodes de décongestion à long terme.

La résolution de conflit à moyen terme

Dans le cadre de la planification tactique, et en se basant sur des informations liées aux intentions des avions et sur un modèle de prédiction de trajectoire, une procédure de détection de conflit est exécutée afin de détecter les conflits imminents. Un mécanisme d'alerte en informe ensuite le système de contrôle de trafic aérien (ATC). L'horizon temporel considéré est de l'ordre d'une quinzaine à une trentaine de minutes. Les modèles de prédiction de trajectoire sont : soit déterministes, soit stochastiques, soit de pire cas. Le calcul de la prédiction de trajectoire est basé sur des estimations de la position et de la vitesse de l'avion, sur des informations météorologiques et sur un modèle de performance avion comme le modèle BADA [5]. La position et la vitesse de l'avion peuvent être estimées, à partir de mesures radar, ou peuvent être diffusées par l'avion lui-même, (cf. *radar mode-S* [20] ou *système ADS-B* [101]). Les informations sur les intentions incluent des instructions du contrôleur, ainsi que les procédures opérationnelles (par exemple, comment une descente est exécutée). Les informations météorologiques incluent les vents prévus et les profils de température. Le modèle de performance décrit le comportement dynamique de l'avion. Les avions commerciaux, par exemple, peuvent être modélisés par un simple modèle cinématique. La prise en compte de l'incertitude sur la position avion est effectuée en superposant une répartition d'erreurs de position sur la trajectoire nominale prévue. La forme de la distribution des erreurs de position est estimée sur la base des enregistrements des traces radar.

4. Ces délais évitent que les avions rentre dans un secteur ou une région de contrôle terminale d'un aéroport (*Terminal Manoeuvring Area*, TMA) lorsque les capacités de ces derniers sont saturées. Ainsi, au lieu de faire attendre l'avion en vol il est préférable de le retenir au sol en retardant son décollage.

5. *Base of Aircraft DAta* : un modèle de performance avions mis au point par le Centre expérimental d'Eurocontrol [80].

Au niveau de la résolution de conflit, beaucoup de travaux ont été effectués dans le cadre de conflits à *deux* avions. Dans le cadre de la présente étude néanmoins, il est plus intéressant d'avoir une vue globale du problème. L'approche générale adoptée pour résoudre les conflits, est de choisir certaines manœuvres prédéfinies à exécuter, qui permettent de minimiser une fonction coût, tout en respectant certaines contraintes de séparation.

L'une des approches d'optimisation est celle proposée dans [90]. Dans cette article, on utilise des modèles avions complexes (les avions évoluent selon une dynamiques non-linéaires). Cette approche utilise des variables continues pour représenter la zone de sécurité cylindrique, et convertit le problème de contrôle optimal sous-jacent en un problème d'optimisation non-linéaire de dimension finie. Des algorithmes de point intérieur [6](approche par barrières) sont employés pour effectuer l'optimisation [73].

Le problème de résolution des conflits en trois dimensions est également abordée dans [56]. Les manœuvres utilisées comprennent le changement d'altitude, de cap et de vitesse. La manœuvre qui minimise une fonction d'énergie donnée est choisie parmi toutes les manœuvres qui résolvent les conflits. Une construction géométrique est proposée, obtenue par un algorithme numérique, pour déterminer la manœuvre optimale, dans la cas d'un conflit à deux avions. Pour un conflit multiple, une approximation est utilisée pour calculer une solution sous-optimale.

Dans [66], les manœuvres de résolution utilisées sont à la fois dans le plan horizontal et dans le plan vertical. La solution est obtenue par l'application des équations d'Euler-Lagrange du contrôle optimal [104]. Cette approche analyse aussi la mise en œuvre d'un système de résolution et de détection de conflits avec identification d'éventuelles sources d'erreur et d'incertitudes.

Dans la section 1.2.4, on verra plus en détails quelques méthodes la résolution de conflit à moyen terme.

La résolution de conflit à court terme

La résolution de conflit à court terme est mise en œuvre lorsque les procédés précédents n'ont pas apporté de solution (parfois par manque de temps pour répondre au problème). La résolution à court terme est donc uniquement utilisée dans les cas d'urgence. L'horizon temporel des résolutions à court terme est de l'ordre de quelques minutes. Dans la pratique, l'ATC est responsable de la détection (système STCA : *Short Term Conflict Alert*) et de la résolution des conflits. La responsabilité de la résolution de conflits prévus dans moins d'une minute incombe, quant à elle, à l'équipage de bord qui est assisté par les systèmes comme le TCAS. Dans les paragraphes qui suivent, les systèmes TCAS et STCA sont présentés.

Traffic Alert and Collision Avoidance System (TCAS)

Le TCAS dans sa forme actuelle, est utilisé comme un système supplémentaire qui informe l'équipage des collisions éventuelles et instruit des manœuvres d'évitement uni-

6. Les méthodes de point intérieur [110] forment une classe d'algorithmes qui permettent de résoudre des problèmes d'optimisation convexe (linéaires ou non).

quement dans le plan vertical. Comme le TCAS ne contrôle pas directement l'appareil, c'est au pilote de mettre en œuvre les instructions du TCAS. La détection de conflits par le TCAS est réalisée en faisant des requêtes aux avions proches en utilisant les transpondeurs Mode S. Le transpondeur retourne l'altitude de l'avion questionné, alors que la distance est estimée à partir du temps de réponse. Quand un intrus potentiel, c'est-à-dire un avion en collision potentielle est détecté, un avis de circulation est émis pour informer l'équipage de la présence de l'intrus et le préparer pour une manœuvre possible. Si une collision est imminente, un avis de résolution dans le plan vertical est émis ("monter", "descendre", "ne pas monter", "ne pas descendre", suivant des gammes de vitesse prédéfinies). Lorsque les deux avions sont équipés de TCAS, la coordination entre eux est faite de telle sorte qu'ils réagissent de la meilleure façon possible. Une présentation détaillée du système est donnée par la FAA dans [81].

Short Term Conflict Alert (STCA)

Le système de détection de conflit centralisé (STCA) est un système sol de détection de conflits. Plus précisément, il est utilisé sur les stations ATC dans les zones congestionnées, afin de donner des alertes et des informations aux contrôleurs sur des conflits éventuels. Comme tout système opérant au niveau de l'ATC, le système STCA est centralisé. En effet, il surveille en permanence tous les avions dans un secteur de contrôle et identifie tous les conflits potentiels. Le STCA est plus un concept basé sur un ensemble de spécifications et d'exigences opérationnelles, qu'un système explicitement défini. Sa mise en œuvre peut varier en fonction du fournisseur ATC. Les exigences relatives aux STCA sont décrites dans [7]. L'idée générale derrière le STCA est d'attirer l'attention du contrôleur sur les conflits imminents, pour qu'il puisse les résoudre à temps. Comme chaque contrôleur surveille un grand nombre d'avions, le niveau de nuisance doit être faible pour que le contrôleur ne soit pas surchargé d'alertes inutiles.

Afin de prédire la trajectoire future de l'appareil et donc toutes les pertes éventuelles de séparation, des données de surveillance sont utilisées, en particulier la vitesse sol de l'avion, ainsi que sa position et son taux de montée. Par contre, aucune information sur l'intention de l'équipage n'est exploitée. Afin de minimiser le nombre de fausses alertes, les probabilités de perte d'espacement vertical et latéral sont combinées de telle sorte que la probabilité de conflit soit estimée et utilisée pour déclencher une alarme. L'horizon d'action du STCA a été fixé à deux minutes ce qui est un bon compromis entre une prédiction fiable et la minimisation de fausses alarmes.

Nous venons de voir les différents critères qui différencient les méthodes de détection et de résolution de conflit. Dans les sections suivantes, nous allons aborder plus en détails, certaines les méthodes parmi les plus étudiées, pour la décongestion et pour la résolution de conflits.

1.2.3 Méthodes de décongestion

Dans cette section, on présente quelques méthodes, pour la planification pré-tactique, qui cherchent à réduire la congestion globale du système.

Algorithme génétique pour le problème de bi-allocation

L'approche développée par Oussedik et Delahaye [83], traite le problème complet de bi-allocation (allocation de routes et d'heures de départ), par une approche stochastique originale. Cette approche utilise une mesure de la congestion des secteurs qui représente les principaux indicateurs de charge de contrôle. Pour réduire la congestion dans un espace aérien, on peut soit changer les créneaux horaires de départ des avions (*séparation temporelle*), appelés *slots*, soit changer leurs routes (*séparation spatiale*). Les changements de slots doivent rester dans un certain domaine limité et la modification des routes ne doit pas impliquer un rallongement supérieur à 10% de la longueur initiale. Afin d'assurer l'équité entre les compagnies aériennes, une taxe pourra être imposée aux vols non déviés et non retardés après optimisation. Un algorithme génétique est utilisé pour optimiser le trafic, en utilisant des routes alternatives issues du monde réel. Une approche dynamique permet d'effectuer cette optimisation sur une fenêtre temporelle glissante, afin de disposer d'information fiable par rapport aux incertitudes sur les trajectoires. En effet, il est inutile de planifier trop longtemps à l'avance car le niveau d'incertitude associée aux trajectoires remet en cause les décisions qui sont prises dans le passé (l'incertitude ne fait que croître avec le temps).

Le modèle proposé par [83] est le suivant. À chaque vol i, on associe un couple de variables de décision (δ_i, r_i) où δ_i est l'avance ou le retard par rapport au temps initial de départ et r_i la nouvelle route proposée. Ces variables seront choisies parmi deux ensembles finis et discrets, Δ_i pour les slots et R_i pour les routes, prédéfinis pour chaque vol i.

La charge de contrôle pour un secteur S_k pendant une période t est définie par

$$W^t_{S_k} = \omega W_{mo_{S_k}}(t) + \psi W_{co_{S_k}}(t),$$

où $W_{mo_{S_k}}(t)$ est la charge de monitoring (surveillance des différentes trajectoires des avions dans un secteur), $W_{co_{S_k}}(t)$ est la charge de coordination (échange des informations avec les contrôleurs des secteurs adjacents) et les paramètres de pondération ω et ψ, fixés par l'utilisateur, appartenant à $[0, 1]$ permettent d'accorder plus ou moins d'importance aux deux indicateurs de congestion.

Après avoir obtenu les charges instantanées de chaque secteur, Oussedik et Delahaye définissent deux mesures importantes :

- $\sum_{t=0}^{t=T} W^t_{S_k}$: la surface de congestion relevée pendant la journée ; où T est le nombre de période temporelles.
- $\max_{t\in[0,T]} W^t_{S_k}$: le maximum de congestion relevée sur la journée.

Ces deux indicateurs de congestion d'un secteur sont conjointement utilisés dans la fonction-objectif qu'il faudra minimiser :

$$W := \sum_{k=1}^{k=P} ((\sum_{t=0}^{t=T} W^t_{S_k})^{\phi} \times (\max_{t\in T} W^t_{S_k})^{\varphi})$$

ϕ et φ sont des paramètres de pondération fixés par l'utilisateur et P est le nombre total de secteurs.

Le problème traité a été prouvé NP-difficile [12] avec un espace d'état non séparable de fait de l'interaction entre vols (conflits entre vols, capacité des secteurs et aéroports et

connexions entre vols). Il a donc été abordé par optimisation stochastique (algorithmes génétiques). Le codage d'une solution (ou *chromosome*), représentant un individu d'un ensemble de M solutions appelé population, est effectué sous la forme d'une matrice qui représente la planification des N vols pour une journée de trafic. Chaque vol i possède un décalage δ_i^j où j est l'indice du décalage choisi dans l'ensemble Δ_i et une route r_i^k où k est l'indice de la route choisie dans l'ensemble R_i comme le montre la figure 1.3.

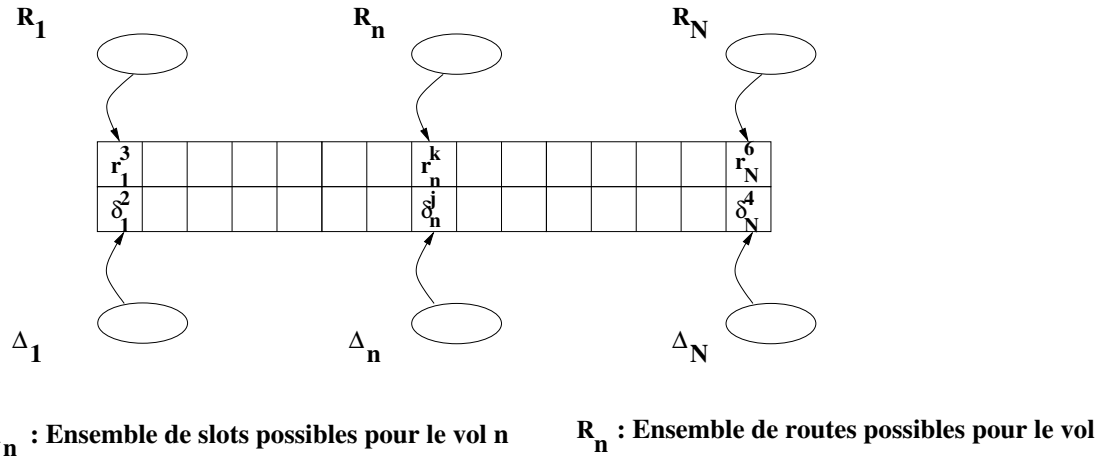


FIGURE 1.3 – Exemple de codage d'une solution

La *fitness* d'une solution, qui représente sa qualité par rapport à la fonction-objectif est alors donnée par :

$$\text{fitness}(solution) = \frac{W(ref)}{W(solution)}$$

où W est la fonction-objectif et où ref représente la distribution initiale des plans de vol (solution de départ ou de référence). Ainsi, une $fitness > 1$ traduit une situation qui réduit la congestion, par rapport à la situation de référence.

Pour chaque vol i d'une solution (ou chromosome) p, Oussedik et Delahaye définissent une fonction C_i^p qui représente la congestion rencontrée par le vol i dans la configuration de la solution p. L'opérateur de *croisement* utilisé ici, qui tente de générer un chromosome avec les meilleurs caractéristiques de ses deux *parents*, garde pour chaque vol i le *gène* (δ_i^j, r_i^k) du parent (1 ou 2) qui induit la meilleure performance c'est-à-dire, celui qui vérifie la condition $C_i^1 < lC_i^2$ avec $l \in [0.7, 0.95]$. Si cette condition n'est pas vérifiée, les parents sont dits équivalents et le gène correspondant au vol i de l'un des deux est choisi au hasard avec une probabilité de 0.5. L'opérateur de mutation est utilisé uniquement sur les vols impliqués dans les *pics de congestion*. Il permet de déterminer si un vol va être avancé ou retardé.

Cet algorithme a été testé sur une journée de trafic de l'espace aérien français. Les routes alternatives utilisées ont été obtenues par filtrage des différentes routes possibles pour chaque couple origine-destination sur une semaine de plans de vol. Cette méthode

permet alors de réduire d'un facteur 2 la congestion sur l'ensemble de l'espace aérien français en fournissant une solution très robuste aux incertitudes.

Programmation par contraintes pour le problème d'allocation de délais

La modélisation consistant à allouer uniquement des délais aux avions, a été utilisée pour tenter de résoudre directement les conflits à long terme (travaux de Barnier et Allignol [10]). Cette approche est basée sur l'estimation du coût direct de la résolution de tous les conflits au-dessus d'un certain niveau de vol sous la contrainte que les avions puissent suivre scrupuleusement leurs trajectoires dans les 4 dimensions. Actuellement, la CFMU [7] (*Central Flow Management Unit*) est en charge de l'optimisation du trafic. Pour cela, il retarde les slots de départ des avions impliqués dans des secteurs en-Route surchargés. Le but du CFMU est d'assurer que le nombre d'avions par heure reste inférieur aux capacités secteurs fournies par les centres de contrôle en-Route. L'un des inconvénients de ce modèle de régulation, est que la charge de travail du contrôleur n'est pas directement proportionnelle à la capacité des secteurs.

Sachant cela, au lieu de tenter de satisfaire la capacité en-Route, Barnier et Allignol ont utilisé une modélisation qui consiste à résoudre tous les conflits potentiels entre les trajectoires avec un ajustement du temps de départ. Ainsi, à chaque avion, un délai est associé tel que tous les conflits (au-dessus d'un certain niveau de vol) sont évités. L'algorithme de résolution utilisé est un algorithme de *programmation par contrainte* (PPC) [8]. Cette modélisation de granularité fine génère beaucoup plus de contraintes que la modélisation macroscopique utilisant les capacités secteurs, mais elle a l'avantage de garantir des trajectoires sans conflits. L'approche adoptée de programmation par contrainte permet d'obtenir des bornes au niveau des délais maximaux nécessaires pour résoudre les conflits, ce qui permet de déduire des résultats sur la faisabilité d'une telle régulation. Imposer des délais aux avions est une mesure contraignante pour les compagnies aériennes, car des correspondances peuvent alors être perdues. De fait, la valeur des délais attribués doit être aussi faible que possible. Le problème résultant est intrinsèquement disjonctif (combinatoire), puisque pour chaque conflit entre un avion i et un avion j, de deux choses l'une : i doit précéder j ou inversement. Les trajectoires sur lesquelles l'algorithme travaille, sont générées par le simulateur CATS qui produit des échantillons toutes les 15 secondes. Les trajectoires sont ensuite comparées deux à deux, afin de détecter les pertes de séparation. Chaque fois qu'une situation ne possède pas de solution, elle doit être modifiée, avec une valeur plus grande de délai. De plus, les vols issus des pays hors de la zone Eurocontrol ne peuvent pas être retardés. Leurs variables de délais doivent donc être fixées à zéro.

Le problème d'allocation de délais est modélisé de la façon suivante. Pour chaque vol

7. La CFMU est l'unité opérationnelle d'EUROCONTROL, située à Bruxelles. Sa mission est d'améliorer la sécurité grâce à une gestion coordonnée du trafic aérien en Europe. Il a aussi pour but de réduire la congestion dans l'air et d'assurer l'utilisation efficace de la capacité disponible.

8. En programmation par contraintes [89], on sépare la partie modélisation à l'aide de problème de satisfaction de contraintes (CSP), de la partie résolution dont la particularité réside dans l'utilisation active des contraintes du problème pour réduire la taille de l'espace des solutions à parcourir (on parle de propagation de contraintes).

i on note :

- $\{p_i^k\}$ est la séquence ordonnée chronologiquement des points 3D de la trajectoire de l'avion i, k étant l'indice du $k^{ème}$ point de cette séquence,
- t_i^k est l'instant auquel l'avion i sera au point p_i^k s'il n'est pas retardé,
- Un ensemble de variables de décision $D = \{\delta_i, \forall i \in \{1, ..., n\}\}$ où δ_i est le délai associé à l'avion i tel que $0 \leq \delta_i \leq$ délai_max.
- $\theta_i^k = t_i^k + \delta_i$ l'instant auquel l'avion sera au point p_i^k s'il est retardé de δ_i,
- $d_{ij} = \delta_j - \delta_i$ la différence de délais entre les deux avions i et j.

Pour tous points p_i^k et p_j^l en conflit géométrique (dans l'espace 3D), la séparation est assurée si :

$$\theta_i^k \neq \theta_j^l,$$

ce qui peut être réécrit avec la différence de variable d_{ij}

$$d_{ij} \neq t_i^k - t_j^l.$$

Pour un point p_i^k en conflit avec un vol j, on prend en compte tout le segment de la trajectoire de j en conflit avec p_i^k :

$$\{p_j^l, \forall l \in \{a, ..., b\}\}$$

avec a (respectivement b) les indices du premier (respectivement dernier) point de ce segment. En d'autres termes, la séparation est assurée dès que :

$$d_{ij} \notin \{t_i^k - t_j^l, \forall l \in \{a, ..., b\}\}, \tag{1.6}$$

$$d_{ij} \notin [l_{ij}^k, u_{ij}^k],$$

avec l_{ij}^k et u_{ij}^k les bornes inférieure et supérieure de l'ensemble de la contrainte 1.6 (on notera l^k et u^k ces bornes dans ce qui suit pour alléger la notation). Si le point suivant de la trajectoire de l'avion i, p_i^{k+1}, est en conflit avec un autre point de la trajectoire de l'avion j, on obtient un autre segment interdit :

$$d_{ij} \notin [l^{k+1}, u^{k+1}]$$

Si les deux segments se chevauchent, la séparation est assurée si :

$$d_{ij} \notin [\min(l^k, l^{k+1}), \min(u^k, u^{k+1})]$$

et plus généralement, on obtient

$$d_{ij} \notin [\underline{lb_1}, \overline{ub_1}] \tag{1.7}$$

avec $\underline{lb_1} = \min\{lb^{k+u}, u \in [0, s]\}$ et $\overline{ub_1} = \max\{ub^{k+u}, u \in [0, s]\}$ avec s tel que $\{p_i^k, p_i^{k+1}, ..., p_i^{k+s}\}$ représente les points successifs pour lesquels l'avion i est en conflit avec la trajectoire de l'avion j. Si les deux vols i et j sont en conflit σ fois avec des intervalles disjoints, la séparation est assurée si :

$$d_{ij} \notin [\underline{lb_1}, \overline{ub_1}] \cup ... \cup [\underline{lb_\sigma}, \overline{ub_\sigma}]$$

Enfin, la solution résultante aura pour coût :

$$\text{coût} = \max\{\delta_i, i = 1, 2, ..., n\}$$

Dans cette modélisation PPC, le décollage et l'atterrissage des avions sont tronqués autour des aéroports (avec un rayons de 10NM), comme les trafic est géré avec des procédures spécifiques au contrôle en zones terminales (TMA). Après vérification des contraintes, les trajectoires sont à nouveaux échantillonnées avec un pas de temps de 1 min (plus raisonnable que les 15 secondes utilisées lors de la détection de conflit). Afin d'améliorer la robustesse des solutions face aux incertitudes sur le temps de départ, un paramètre *ext*, représentant une quantité fixe de temps, est introduit pour rallonger les intervalles de conflit. Une telle extension de *ext* minutes qui étire chaque bout de l'intervalle de conflit permet de gérer une incertitude de $\pm\frac{ext}{2}$ minutes sur les slots de départ, au prix de l'augmentation du coût des solutions. La stratégie de recherche adoptée est inspirée des techniques d'ordonnancement standards où l'on retrouve dans les contraintes du type la nature disjonctive du problème. Comme pour certains schémas de recherche utilisés dans l'ordonnancement, on doit satisfaire une des deux contraintes : $d_{ij} < \underline{lb}$ ou $d_{ij} > \overline{ub}$. La recherche commence avec la variable d_{ij} qui a la plus faible *densité*, c'est-à-dire dont le quotient entre la taille du domaine et la différence entre les bornes du domaine est le plus faible. Pour compenser le fait que le coût ne prend en compte que le délai maximal et non pas le total des délais, la recherche commence par l'intervalle d_{ij} qui correspondant à l'augmentation potentielle minimale des variables δ_i et δ_j. Quand tous les conflits sont ordonnés, on procède à l'étiquetage des variables de décision δ_i avec une heuristique de sélection, les valeurs les plus proches de 0 étant testées en premier, de façon à réduire le retard total.

Le modèle PPC utilisé a été implémenté avec la librairie FaCiLe[11]. Il a été testé sur différentes journées de trafic en 2007 avec un maximum de 9500 vols. Cet algorithme permet d'obtenir des solutions optimales (de coût minimal) de façon systématique pour toutes les instances du problème qui n'épuisent pas la mémoire (jusqu'à 600 000 paires de trajectoires en conflit). Le total des délais (figure 1.5), ainsi que le pourcentage d'avions retardés (figure 1.4) augmentent de façon linéaire, puis de façon quasi-exponentielle en fonction du nombre total de vols considérés. Les résultats sur ces figures sont identifiés par la date de la journée ainsi que par les types de routes utilisées : standards (c'est-à-dire des routes qui suivent les balises) ou directes. Par exemple, "070123s" correspond à la journée du 23 janvier 2007 avec des routes standard.

Ce modèle a été également testé sur des routes directes qui sont les trajectoires idéales pour les compagnies aériennes du point de vue coût opérationnel, mais qui généreraient un trafic incontrôlable par un opérateur humain. Cependant, les routes directes génèrent un *graphe de contraintes*[9] plus réduit que celui des routes standards. De plus, le total des

9. Dans un problème de satisfaction de contraintes (CSP), une contrainte correspond à une relation entre plusieurs variables qui limitent l'ensemble des valeurs que peuvent prendre ces variables simultanément. Un graphe de contraintes est un graphe qui représente la structure du problème CSP, avec les sommets qui représentent les variables du problème et une arête existe entre deux sommets, s'il existe une contrainte entre les variables correspondantes.

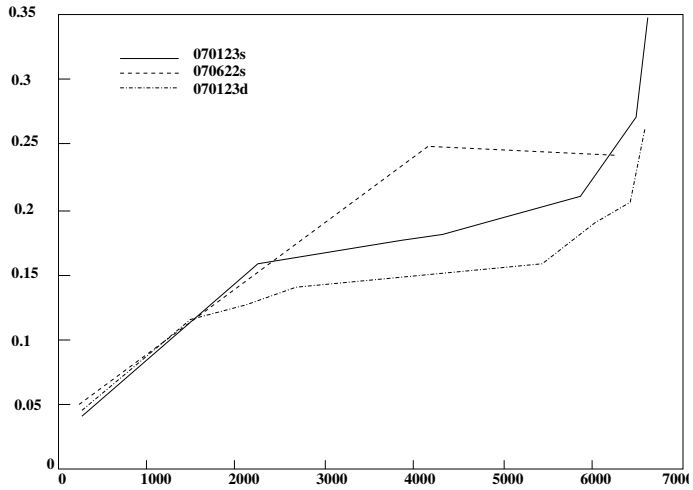


FIGURE 1.4 – Pourcentage d'avions retardés en fonction du nombre d'avions considérés

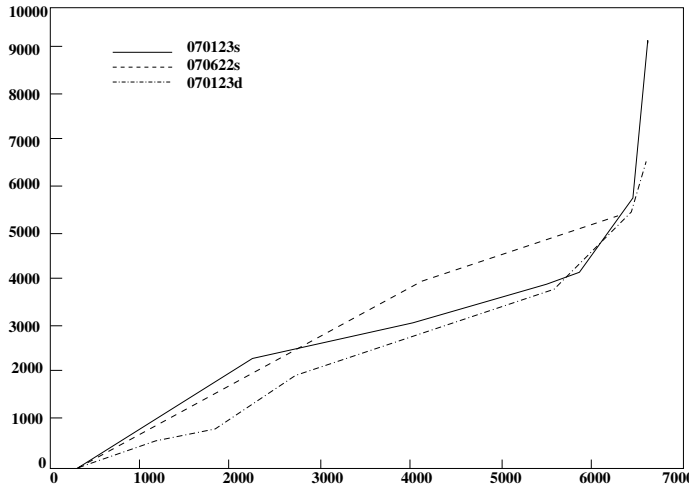


FIGURE 1.5 – Total des délais en minutes en fonction du nombre d'avions considérés

délais est plus faible avec les routes directes, par contre, en fonction du jour de trafic, leurs coût maximales peut être plus grand.

Bien que le problème soit de très grande taille, l'algorithme a réussi à déterminer les solutions optimales (de coût minimal), pour tous conflits (au-dessus d'un certain niveau de vol). Les résultats en terme de délais maximaux, de temps total d'attente et de proportion d'avions retardés sont comparables à ceux alloués par le CFMU. Cependant, la résolution de conflits pendant des journées très congestionnées peut être très coûteuse. En ce qui concerne la gestion des délais, une incertitudes sur les retards au départ de ± 2 minutes peut générer une très grande quantité de délais, bien plus importante que les performances requises par SESAR.

1.2.4 Méthodes de résolution de conflits

Dans cette section, nous présentons les méthodes de résolution de conflits les plus étudiées dans la littérature. Ces méthodes correspondent à la planification tactique des trajectoires.

Résolution de conflits par algorithme génétique

Le problème de résolution de conflits aériens est un problème fortement combinatoire. En effet, si on se restreint au plan horizontal, dans le cas d'un cluster comprenant n avions, il y a $\frac{n(n-1)}{2}$ conflits potentiels. De plus, la résolution de conflits nécessite d'avoir recours à un simulateur pour calculer les positions futures d'un avion, ce qui rend impossible la recherche analytique d'une solution, ainsi que l'utilisation de méthodes classiques d'optimisation (dérivées non disponibles). C'est pourquoi, le problème de résolution de conflits a largement été abordé par des approches stochastiques. L'algorithme génétique (AG), qui est un cadre classique pour la résolution des problèmes à forte combinatoire, est adapté, dans les paragraphes suivants, à ce problème spécifique.

Offset, point tournant et AG

Nous présentons ici, l'approche proposé par Durand dans [34] pour la résolution des conflits en tactique (sur un horizon temporelle de moins de 12 minutes), en considérant des trajectoires 4D (3D + temps), avec prise en compte des incertitudes sur la position des avions.

Pour résoudre les conflits, Durand propose d'appliquer des manœuvres opérationnelles aux trajectoires impliquées et il modélise le problème de résolution de conflit sous la forme d'un problème d'optimisation. Les critères à optimiser sont les suivants : la minimisation du nombre total de manœuvres, la minimisation du retard induit pour chaque avion et, bien que la séparation entre aéronefs soit une contrainte dans le problème initial, dans la modélisation de Durand il s'agit d'un critère d'optimisation (on cherche à minimiser le nombre d'avions en conflit et à terme on souhaite l'annuler).

La méthode d'optimisation utilisée, est l'évolution artificielle (ou algorithmes génétiques). Les algorithmes génétiques sont préférables au recuit simulé, car bien que le recuit simulé converge généralement plus rapidement que les algorithmes génétiques comme le montrent les résultats de Ingber et Rosen dans [59], il ne donne qu'une solution dans le cas d'un problème multimodal.

Pour résoudre les conflits en temps réel, Durand utilise un simulateur qui effectue, toutes les δ minutes (en pratique $\delta = 2$ ou 3 minutes) une détection des paires de conflits sur une fenêtre temporelle Δ (en pratique Δ, est entre 8 et 12 minutes). Ces paires sont ensuite regroupées dans des clusters, par fermeture transitive de la relation "est en conflit avec". Les conflits de chacun de ces clusters sont résolus par un algorithme génétique. L'algorithme propose des solutions au simulateur qui vérifie que les nouvelles trajectoires, ainsi obtenues, ne créent pas de conflits avec les trajectoires d'un autre cluster. Si c'est le cas, les clusters sont regroupés et une nouvelle résolution est effectuée.

Décrivons maintenant l'ensemble des manœuvres qui peuvent être considérées par

cette modélisation. Tout d'abord, les changements de vitesse ne sont pas considérés dans le plan horizontal, car la modification de la vitesse ne se révèle efficace que dans la phase de descente. Un conflit est d'autant plus difficile à résoudre que l'angle d'incidence des trajectoires en conflit est faible et que les avions ont des vitesses proches. La plupart des trajectoires optimales peuvent être approchées par des trajectoires rectilignes par morceaux.

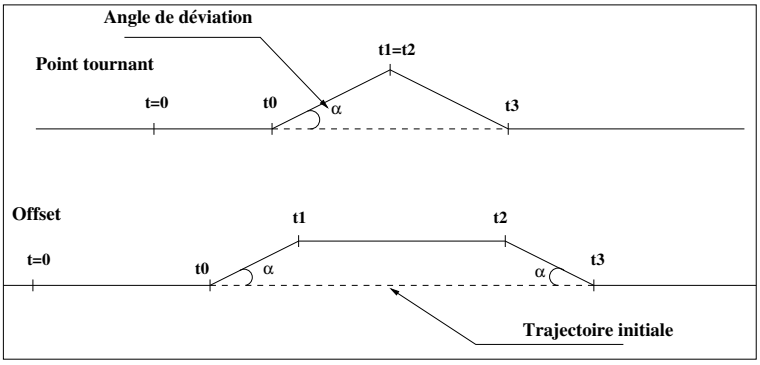


FIGURE 1.6 – Manœuvre d'évitement par offset et par point tournant dans le plan horizontal

Comme cette approche veut garder l'humain dans la boucle, elle génère des trajectoires qu'un pilote peut mettre en œuvre. D'où l'utilisation de manœuvres satisfaisant les contraintes opérationnelles. Dans le plan horizontal, ces manœuvres sont les suivantes :

- le *point tournant* qui consiste à modifier le cap d'un avion et de le ramener ensuite sur sa trajectoire (figure 1.6).
- l'offset qui induit un décalage latéral par rapport à la modélisation précédente (figure 1.6). Cette modélisation est indispensable pour un conflit de *rattrapage* (un avion rattrape l'autre sur une même trajectoire).

Ces deux manœuvres sont modélisées par quatre variables de décision.

- l'altération de cap, α, dont la valeur peut être 10, 20 ou 30 degrés à droite ou à gauche,
- la date de début de la manœuvre d'éloignement t_0,
- la date de la fin de la manœuvre d'éloignement t_1,
- la date du début du virage de retour sur la trajectoire initiale t_2.

Un cluster à n avions engendre donc, un problème d'optimisation à $4n$ variables.

Dans la plan vertical (figure 1.7), on peut interrompre la phase de montée au temps t_0 pour la reprendre à l'instant t_1. Pendant la phase de croisière, on peut attribuer à l'avion, un niveau immédiatement inférieur à t_0 pour reprendre le niveau initial à t_1. Environ 50 Nm avant le début de la descente, celle-ci peut être anticipée à t_0. Un pallier est marqué à t_1 en attendant de rattraper le plan de descente. En phase de descente, aucune manœuvre n'est possible, mais l'avion peut réduire sa vitesse de 10%, 20% ou 30%. Le modèle prévoit de donner une seule manœuvre à la fois par avion. Ainsi, un avion ne peut, par exemple,

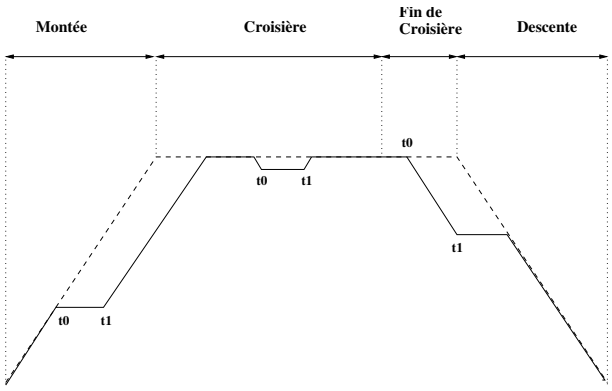


FIGURE 1.7 – Manœuvres dans le plan vertical (altitude en fonction du temps).

être mis en descente anticipée alors qu'il est déjà sous le coup d'une déviation dans le plan horizontal. De plus, une manœuvre commencée ne peut être remise en cause.

Les incertitudes sur la position avion sont modélisées, dans le plan horizontal, par un polyèdre, qui délimite les positions possibles de l'avion. Dans le plan vertical, ces incertitudes sont représentées par un rectangle avec une altitude maximale et minimale (figure 1.8).

Pour la mise en œuvre de l'algorithme génétique, Durand [59] utilise une *population* (ensemble des solutions considérées à chaque itération) dont la taille est proportionnelle à n, le nombre d'avions dans le cluster considéré. La proportion retenue par Durand est de 10 individus par avion. De plus, une fonction *fitness*, qui prend à la fois en compte les contraintes de séparation et de la qualité de la résolution lorsque celle-ci est réalisée, doit être définie. Pour prendre en compte les différents critères à optimiser, Durand remplace la fonction fitness unique par une matrice triangulaire inférieur F de taille $(n \times n)$ dont les éléments sont les suivants pour une solution donnée :

- $F_{i,i}$ permet de mesurer l'allongement de la trajectoire de l'avion i ainsi que le nombre de manœuvres nécessaires pour obtenir cette trajectoire,
- $F_{i,j}$ avec $i < j$ mesure la violation de la séparation entre l'avion i et j. Il est nul si les deux avions ne sont pas en conflit

Pour les besoins de l'opérateur de sélection, une fitness scalaire f est définie, de telle sorte que les individus (c'est-à-dire les solution), pour lesquels il reste des conflits, aient des fitness inférieures à 0.5 et que les individus, où il n'y a plus de conflit aient des fitness supérieures à 0.5

$$f = \begin{cases} \frac{1}{2} + \frac{1}{1+\Sigma_i F_{i,i}} \text{ si } F_{i,j} = 0 \ \forall i \neq j, \\ \frac{1}{2+\Sigma_{i \neq j} F_{i,j}} \text{ sinon.} \end{cases}$$

Pour les besoins des opérateurs de croisement et de mutation, une fitness locale F_i, pour un avion i d'un individu donné, est définie comme suit

$$F_i = \sum_{j=1}^{n} F_{i,j}$$

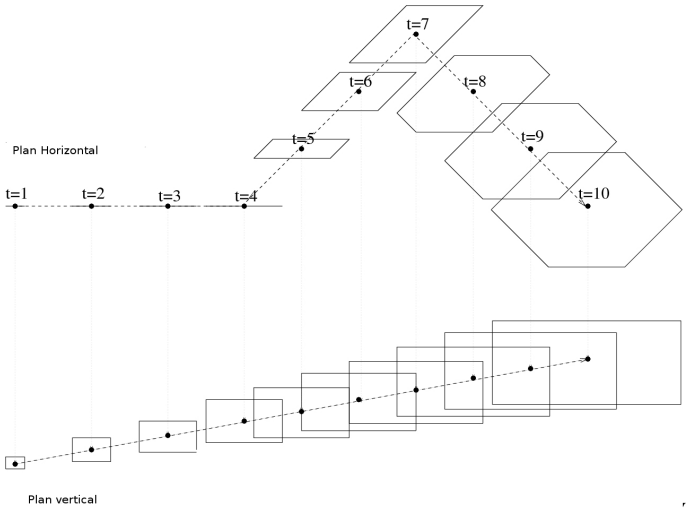


FIGURE 1.8 – Modélisation des incertitudes de position.

L'opérateur de croisement, appliqué à 50% de la population, est défini comme suit. Après avoir choisi deux parents A et B, on compare les fitness locales de leurs avions (notées A_i et B_i pour l'avion i). Si $A_i \leq B_i - \delta$ (Durand fixe δ à n fois la distance standard de séparation horizontale) les deux enfants héritent de l'avion i du parent A. Si $B_i \leq A_i - \delta$, ils héritent de l'avion i du parent B. Sinon l'opérateur de croisement barycentrique [10] est appliqué sur les avions i des parents A et B. L'opérateur de mutation, appliqué sur 15% de la population, est défini comme suit. On choisit au hasard un avion dont la fitness locale est inférieure à un seuil ε donné. L'une des variables qui modélisent l'avion choisi est modifiée soit d'un pas de temps de 12 secondes, soit d'un angle de 10 degrés.

L'optimisation s'arrête au bout d'un nombre fixe de générations (d'itérations). Comme le solveur n'a qu'une vision à court terme des trajectoires, il peut avoir tendance à repousser le conflit au-delà de la fenêtre temporelle de résolution au lieu de résoudre les conflits du cluster considéré. Pour contrer cet effet horizon, la fonction coût est pénalisée dans un tel cas.

L'algorithme a été testé par Durand pour résoudre des clusters mettant en jeu n avions en conflit, convergeant vers un même point. Ces avions volent à 400 Kt en considérant 3% d'incertitude sur la vitesse horizontale. Pour cette application, le nombre de générations est fixé à 20 et la taille de la population à 50. L'algorithme a réussi à résoudre des clusters comprenant jusqu'à 20 avions en conflit et il a nécessité 750 appels à la fonction d'évaluation.

AG et programmation linéaire

10. À partir de deux parents P_1 et P_2, deux gènes $P_1(i)$ et $P_2(i)$ sont sélectionnés à la même position i et deux nouveaux gènes $C_1(i)$ et $C_2(i)$ sont crées par pondération sur la droite qui relie ces derniers. $C_1(i) = aP_1(i)+(1-a)P_2(i)$ et $C_2(i) = (1-a)P_1(i)+aP_2(i)$ avec a un coefficient aléatoire de pondération qui n'appartient pas forcément à $[0, 1]$.

Cette approche a été proposée par Médioni et al. dans [78]. Comme précédemment, on cherche à résoudre des cluster à n avions en conflit, mais les hypothèses retenues ici sont plus restrictives puisque les trajectoires des avions sont des droites contenues dans un plan horizontal et il n'y a aucune prise en compte de l'incertitude sur la position des avions.

Une condition nécessaire est suffisante pour que les n avions soient séparés est qu'ils soient séparés deux à deux. On s'intéresse donc au cas de deux avions i et j, ayant pour origine respectivement O_i et O_j et pour destination respectivement D_i et D_j et dont les trajectoires (initiales) forment un angle ϕ_{ij}. La condition de séparation entre ces deux avions s'écrit ainsi

$$v_i^2 v_j^2 \sin(\phi_{ij})^2 (t_{ij}^i - t_{ij}^j)^2 \geq d^2(v_i^2 + v_j^2 - 2v_i v_j \cos(\phi_{ij})), \tag{1.8}$$

avec v_i et v_j les vitesses respectives des avions i et j ; t_{ij}^i (resp. t_{ij}^j) l'instant auquel la trajectoire non déviée de l'avion i (resp. j) coupe celle de l'avion j (resp. i) et d est la norme de séparation horizontale choisie.

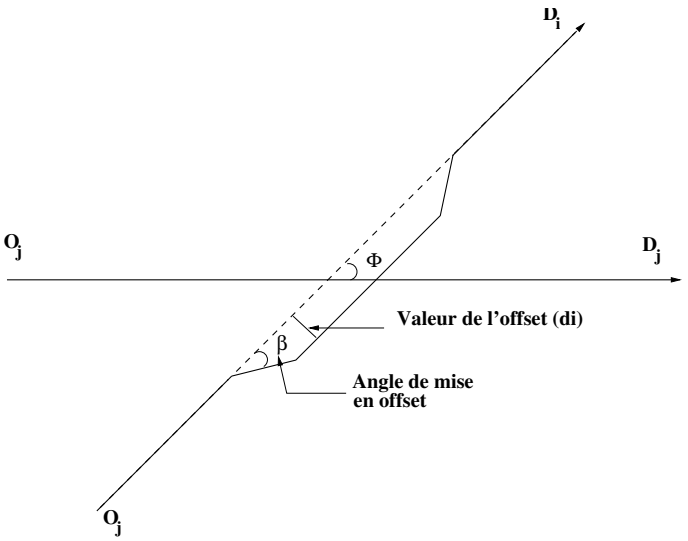


FIGURE 1.9 – Conflit à deux avions, un seul avion étant dévié

Si cette condition n'est pas respectée (donc dans le cas d'un conflit), la modélisation, utilisée par Médioni et al., consiste à mettre en offset l'un ou les deux avions i et j impliqués dans ce conflit, comme le montre la figure 1.9. L'offset est ici modélisé à l'aide des variables d'optimisation suivantes : l'angle de mise en offset β (fixé par l'utilisateur), le sens de l'offset (à gauche ou à droite), et la valeur de l'offset (distance entre la trajectoire initiale de l'avion et la partie de la manœuvre d'offset parallèle à cette trajectoire), noté d_i (resp. d_j) pour l'avion i (resp. j) . Cette mise en offset a pour effet, d'introduire un décalage d'une trajectoire par rapport à l'autre et, par conséquent, de modifier l'instant auquel la trajectoire de chacun des deux avions coupe celle de l'autre.

Suivant l'ordre de passage des avion i et j au point d'intersection de leurs trajectoires (avion i avant j ou inversement), on peut déduire deux conditions suffisantes linéaires en t_{ij}^i et en t_{ij}^j, à partir de la condition (1.8). En choisissant a priori l'ordre de passage des avions et le sens de l'offset. On obtient un sous-problème qui nous conduira à une solution sous-optimale mais qui aura l'avantage de se ramener à un programme linéaire. En effet, en exprimant t_{ij}^i et t_{ij}^j en fonction de d_i et d_j, la condition (1.8) se ramène à une inéquation linéaire en d_i et d_j. Par exemple, si les offset des deux avions sont vers l'extérieur et si l'avion i passe derrière l'avion j on obtient

$$(t_{ij}^i + \frac{di\tan(\frac{\beta}{2})}{v_i} + \frac{di\cot(\phi_{ij})}{v_i} + \frac{dj}{v_i\cot(\phi_{ij})} - t_{ij}^j - \frac{dj\tan(\frac{\beta}{2})}{v_j} - \frac{dj\cot(\phi_{ij})}{v_j} - \frac{di}{v_j\cot(\phi_{ij})})v_iv_j\sin(\phi_{ij}) \geq d\sqrt{v_i^2+v_j^2-2v_iv_j\cos(\phi_{ij})}.$$

On a donc obtenu des contraintes linéaires rendant compte des séparations des avions deux à deux, ce qui donne pour un conflit à n avions $\frac{n(n-1)}{2}$ contraintes, auxquelles sont ajoutées les contraintes $0 \leq d_i \leq d_{borne}$ avec $i \in \{1, ..., n\}$ et d_{borne} une borne imposée par l'utilisateur aux valeurs d'offset pour ne retenir que des solutions "raisonnables". On se retrouve donc avec $\frac{n(n+1)}{2}$ contraintes.

La fonction à minimiser est la somme des retards des avions :

$$S(d1, ..., dn) = 2\sum_{i=1}^{n} \frac{di\tan(\frac{\beta}{2})}{v_i},$$

avec β l'angle de mise en offset et di la valeur de l'offset de l'avion i.

Il s'agit donc de résoudre un problème d'optimisation linéaire à n inconnues (les valeurs des offsets des n avions), et à $\frac{n(n+1)}{2}$ contraintes. Ce problème peut se résoudre à l'aide de l'algorithme du simplexe. Cependant, comme nous l'avons mentionné plus haut, on obtiendra alors un optimum local (par rapport au problème initial permettant tout ordre de passage et tout sens d'offset) dépendant de la configuration initiale choisie (sens de l'offset et ordre de passage des avions). Si on veut tester toutes les configurations initiales possibles on doit tester $2^{\frac{n(n+1)}{2}}$ combinaisons possibles. Ce problème étant fortement combinatoire, un algorithme génétique est chargé de déterminer une bonne configuration initiale et la méthode du simplexe sert alors à résoudre le problème linéaire, pour chaque configuration du problème.

Cet algorithme a été testé sur un cluster de n avions en conflit qui volent à 400 Kt dont les trajectoires convergent vers un même point. La norme de séparation considéré est de 7 Nm, mais aucune incertitude n'est prise en compte sur la vitesse des avions.

Pour une instance du problème avec $n = 5$ avions, il est encore possible de traiter le problème avec programmation linéaire de façon déterministe : toutes les combinaisons de sens de déviation et d'ordre de passage, il y en a 32 768, peuvent être comparées, afin d'obtenir la solution optimale.

Lorsqu'on teste la stratégie AG pour chercher une bonne configuration initiale avec une population de 150 individus et moins d'une centaine de générations, on retrouve toujours la solution optimale. Le test a été effectué 50 fois.

Pour une instance du problème avec $n = 6$ avions, la solution optimale est trouvée 39 fois seulement sur les 50 essais. Parcourir toutes les combinaisons possibles nécessiterait 2 097 152 applications du programme linéaire.

Cette modélisation ne permet pas d'aborder des problèmes de très grande taille (d'une vingtaine d'avions), car le coût d'évaluation d'une configuration est trop important. En fait, elle fut abandonnée pour d'autres raisons :

- Cette modélisation ne permet pas de prendre en compte l'incertitude sur la position des avions.
- La modélisation des trajectoires d'avion par des droites n'est pas réalistes dans les phases de montée ou de descente.
- Cette modélisation requiert qu'aucun conflit n'ait lieu pendant les phases de mise en offset ou de retour sur trajectoire, ce qui rend le problème beaucoup plus difficile dans les zones denses.

Résolution de conflits par fonctions de navigation et extensions

La deuxième grande classe de méthodes de résolution de conflit dans le trafic aérien, est basée sur les fonctions de navigation (FN). Les FN ont été initialement utilisées en robotique, pour la résolution des conflits entre les véhicules terrestres (robots). Elles ont été proposées pour pallier aux défauts des fonctions de potentiel classiques.

Fonctions de potentiel classiques

Une fonction de potentiel [21] est une fonction à valeur réelle différentiable $U : \mathbf{R}^m \rightarrow \mathbf{R}$. La valeur de la fonction potentiel peut être vue comme de l'energie ; ainsi le gradient du potentiel est une force. Le gradient est utilisé pour définir un champ de vecteurs. L'approche basée sur les fonctions de potentiel dirige un robot comme s'il était une particule qui se déplace dans ce champs de vecteurs. Le gradient peut être intuitivement vue comme une force qui agit sur la particule chargée positivement représentant le robot et qui le pousse vers une particule fixe chargée négativement, qui représente l'objectif. Les obstacles sont, eux aussi, chargés positivement, ce qui induit une force répulsive qui éloigne le robot des obstacles. La combinaison de ces forces forme un champs qui dirige le robot vers la destination et l'éloigne des obstacles.

Le robot suit un chemin de descente, en suivant l'inverse du gradient, $-\nabla U$. Il s'arrête de se déplacer, quand le gradient s'annule. Le problème de cette approche est l'existence de minima locaux différents de l'objectif, où le robot peut rester piégé, sans atteindre son but. C'est pour éviter ce problème, que les fonctions de navigation ont été proposées.

Fonctions de navigation

Les fonctions de navigation ont été formellement définies par Koditchek et Rimon dans [64, 91]. Une fonction $\varphi : F \rightarrow [0, 1]$, ($F \subset \mathbf{R}^n$ est un ensemble borné), est dite fonction de navigation si elle vérifie les propriétés suivantes :

- Elle a un seul minimum au point $q_d \in F^\circ$, où F° représente l'intérieur de l'ensemble F et q_d représente la configuration objectif,
- Sa matrice hessienne est de rang maximal aux points critiques.

Ce travail a été étendu au problème des *véhicules multiples* 2D par Loizou, Dimarogonas et Kyriakopoulos dans [70]. Le problème traité dans le plan est le suivant : m véhicules mobiles se déplacent dans un espace $W \subset \mathbf{R}^2$. Chaque véhicule $i \in \{1, .., m\}$ occupe un

disque V_i de rayon r_i, dans l'espace de travail :

$$V_i = \{q \in \mathbf{R}^2 : \|q - q_i\| \leq r_i\},$$

où q_i est la position (le centre de masse par exemple) de chaque véhicule. La configuration des véhicules est donnée par $q = [q_1^T, ..., q_m^T]^T$.

Les fonctions de navigation sont des cartes de valeur construites en utilisant des fonctions coût, dont l'opposé du gradient s'oriente vers la configuration objectif et s'éloigne des obstacles. La fonction de navigation considérée pour le véhicule i est alors donnée par :

$$\varphi_i(q_i) = \sigma_d \circ \sigma \circ \hat{\varphi}_i(q_i)$$

où $\sigma_d = x^{\frac{1}{k}}$, $\sigma = \frac{x}{1+x}$ et $\hat{\varphi}_i = \frac{\gamma_i(q_i)}{G_i(q_i)}$ est la fonction coût. Ainsi $\varphi_i(q_i) = \frac{\gamma_i(q_i)}{(\gamma_i(q_i)^k + G_i(q_i))^{\frac{1}{k}}}$. Le coefficient k est fixé à une valeur suffisamment grande pour que la fonction φ ait un seul minimum. La fonction $\gamma_i(q_i) = \|q_i - q_{di}\|^2$ représente la distance entre le véhicule i et sa destination q_{di}. Il s'agit de la partie attractive de la fonction de navigation. La fonction $G_i(q_i)$ est la partie répulsive de la fonction de navigation. Elle est, par construction, un indicateur de la proximité d'une collision impliquant le véhicule i. En effet, elle tend vers zéro quand une collision est imminente, et elle augmente lorsque le danger d'une collision diminue.

Initialement, les fonctions de navigation peuvent mener à un comportement indésirable, comme la rotation du mobile sur lui-même. Pour corriger ce défaut, les fonctions dipolaires de navigation ont été développées.

Fonctions de navigation dipolaires

Les fonctions de navigation dipolaires assurent, aux lignes intégrales (tangentes en chaque point au champ de vecteurs) du champs potentiel résultant, d'être tangentes à l'orientation désirée au niveau de l'objectif. La fonction de navigation dipolaire pour le véhicule i est définie ainsi :

$$\varphi_i = \frac{\gamma_i(q_i)}{(\gamma_i(q_i)^k + H_{nh}(q_i)G_i(q_i))^{\frac{1}{k}}},$$

où les fonctions γ_i et G_i restent les même que pour les fonctions de navigation classique. Le terme $H_{nh}(q_i)$ rend le champ de potentiel dipolaire. Il est responsable du potentiel répulsif créé par un obstacle artificiel, utilisé pour aligner les trajectoires à la destination suivant l'orientation désirée θ_{di}. Ce terme est donné par : $H_{nh}(q_i) = \varepsilon_{nh} + ([\cos(\theta_{di}) \sin(\theta_{di})](q_i - q_{di}))^2$ avec ε_{nh} une constante positive fixée par l'utilisateur. Ceci garantit que la trajectoire soit réalisable pour les véhicules non holonomes [11] et évite les mises en rotation sur un point. Un exemple de champs d'une fonction de navigation dipolaire est présenté à la Figure 1.10.

11. Un mobile est dit non-holonome si son mouvement est contraint par une liaison $G(q, \dot{q}, t) = 0$ (avec t le temps, q les paramètres de configuration du mobile et $\dot{q}$ leurs dérivées par rapport à t) et si G n'est pas intégrable.

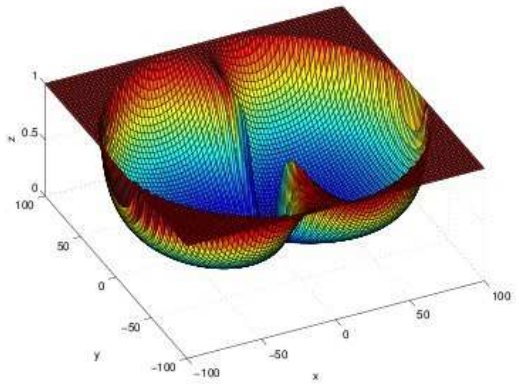


FIGURE 1.10 – Une fonction de navigation dipolaire.

Application à l'ATM

Les fonctions de navigation ont déjà démontré leur efficacité dans le contrôle de mouvement des véhicules, avec la garantie de l'évitement de collision et la convergence vers la configuration objectif. Bien qu'initialement utilisées pour la navigation de robots, des recherches sont faites pour les adapter aux problèmes de l'ATM. En effet, les fonctions de navigation ne prennent pas en compte les contraintes imposées par l'ATM (vitesses bornées, trajectoires lisses, contraintes temporelles). Elles induisent de plus des comportement rédhibitoires pour l'ATM (mise en arrêt, boucle d'attente). Pour pallier à ces problèmes, l'algorithme du modèle de contrôle prédictif (MPC) a été associé au FN par Lygeros dans [92]. L'algorithme MPC considère les FN comme une boîte noire qui produit un ensemble de trajectoires pour tous les avions en connaissant leurs destinations. Le MPC, lui, minimise une fonction coût, tout en respectant certaines contraintes. La fonction coût reflète les buts à long terme de l'avion qui sont d'atteindre la destination aussi rapidement que possible et d'éviter de tourner trop souvent. Les contraintes reflètent des aspects opérationnelles. Le MPC est appliqué de façon périodique (à chaque pas de temps t) sur un horizon temporel T à un ensemble d'avions. Il transmet à chaque pas de temps t, la destination intermédiaire optimale $[x_i^{interm}(t), y_i^{interm}(t)]$ de chaque avion i qu'il contrôle aux FN qui produisent pour chaque avion une trajectoire entre sa position actuelle et sa destination intermédiaire.

Cette association des FN avec l'algorithme MPC, a été testée sur un cluster de conflits comprenant jusqu'à 6 avions qui convergent vers le même point. La vitesse des avions a été contrainte à rester dans $[366, 540]$ Kt. Une incertitude stochastique sur la vitesse du vent, sous la forme d'une gaussienne de moyenne nulle et d'écart type $\sigma = 5.17$ m/s, a été prise en compte.

L'algorithme a réussi à générer des trajectoires sans conflits, mais ces trajectoires sont

loin d'être optimales, avec des déviations beaucoup plus importantes que nécessaire. De plus, bien que la vitesse soit contenue dans l'intervalle contrainte, sa variation n'est pas régulière.

Résolution de conflits par méthode de champs de force

La plus ancienne méthode basée sur les champs de force, pour résoudre un cluster à n avions dans le plan horizontal et sans prise en compte de l'incertitude sur la position des avions, est celle de Zeghal dans [113, 114].

Pour résoudre ce problème, Zeghal définit trois types de forces qui agissent sur l'avion :
- les forces attractives qui permettent aux avions d'atteindre leurs objectifs,
- les forces répulsives qui dépendent de la distance aux obstacles et permettent de les éviter,
- les forces de glissement qui permettent de contourner les obstacles.

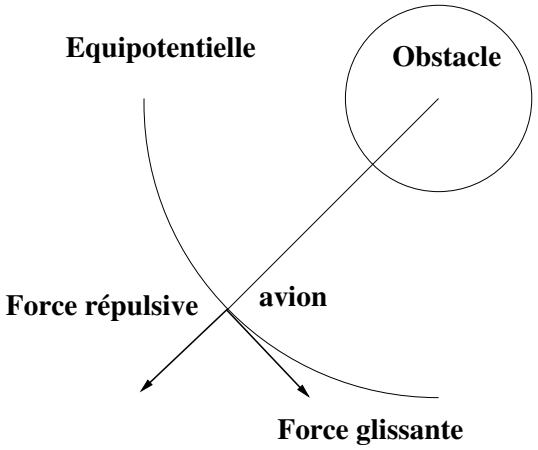


FIGURE 1.11 – Force répulsive et force de glissement

Une force de glissement est tangente à la courbe *équipotentielle de danger* passant par l'avion, alors que la force répulsive lui est normale (figure 1.11). Dans le cas simple de deux avions, on projette leurs vitesses relatives (la vitesse d'un avion relativement à l'autre) sur leurs équipotentielles de danger, et on obtient deux forces de glissement coordonnées [12] (voir figure 1.12).

Ces différentes forces s'exercent sur l'avion, en s'additionnant avec des coefficients variables suivant l'éminence du danger. La coordination d'actions n'a pas pour objectif la recherche d'optimalité, mais vise une bonne robustesse des solutions. L'utilisation de cette méthode est envisagée dans trois systèmes.

12. La coordination d'action des avions implique que chaque avion considère les autres avions qui sont potentiellement en conflit avec lui, mais qu'une cohérence des manœuvres est garantie. Cette cohérence est obtenue par la définition des paires de forces de glissement.

– Le premier est un système hybride d'avions autonomes. Il s'agit d'un système bord qui permet de gérer les conflits, entre 4 et 10 minutes dans le futur. Il fonctionne de façon autonome, à partir d'une perception de l'environnement de l'avion.
– Le deuxième est un système d'aide au contrôleur. Pour cela, le processus individuel est utilisé sous forme de processus centralisé, afin de simuler les trajectoires futures. Le résultat est un ensemble de trajectoires assurant l'évitement sur l'intervalle de temps considéré.
– Le troisième est un système mixte. Dans ce cas, certains avions sont autonomes et d'autres ne le sont pas.

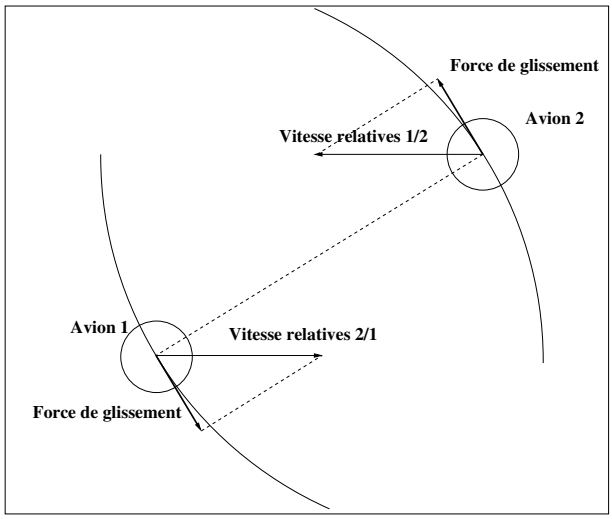


FIGURE 1.12 – Forces de glissement coordonnées

Afin de mieux gérer les grandes vitesses des avions et de maintenir des mouvements d'évitement réduits, l'approche du *point le plus près* CPA [115, 116] a été proposée. Le point CPA est le point où la distance entre le mobile et l'obstacle atteint son minimum. À cause des incertitudes sur la prédiction de la position des avions, la détermination du point CPA doit être faite périodiquement. Cette approche consiste à utiliser des forces répulsives qui dépendent de la distance au point CPA et non pas de la distance aux obstacles. Cette approche s'avère plus efficace que celle utilisant les forces répulsives classiques ou les forces de glissement.

L'approche des champs de force, a été utilisée pour résoudre des conflits dans le plan horizontal impliquant jusqu'à 10 avions qui se déplacent à vitesse constante. La robustesse des solutions est un atout majeur des travaux de Zeghal. Cependant, si ces techniques garantissent la résolution de tous les conflits à deux avions, la généralisation à n avions n'a été vérifiée que sur des exemples. De plus, la modélisation choisie ne permet aucune recherche d'optimalité globale.

Résolution de conflits par Branch and Bound par intervalles

Nous décrivons ici l'approche développée par Médioni dans [77], pour la résolution des conflits aériens. Elle se limite à la résolution de conflits dans le plan horizontal, avec des avions volant à vitesse constante et sans prise en compte de l'incertitude.

La modélisation dans [77] consiste à utiliser des manœuvres de point tournant (figure 1.6) pour les trajectoires en conflit. L'angle de déviation et de retour vers la trajectoire initiale, α, doit être fixé à l'avance, afin que les positions possibles d'un avion pendant la manœuvre aient des formes simples. L'angle α étant fixé (par l'utilisateur), la trajectoire est totalement définie par le sens de la déviation (à droite ou à gauche) et par les instants t_0 et t_1 qui correspondent, respectivement, au début de la déviation et au début du retour vers la trajectoire initiale.

L'algorithme de résolution utilisé est le *Branch and Bound* par intervalles (B&BI). Pour appliquer la *programmation par intervalles*, t_0 et t_1 seront remplacés par des intervalles de manœuvres de la forme $T_0 = [t_{0min}, t_{0max}]$ et $T_1 = [t_{1min}, t_{1max}]$ respectivement. L'angle α prendra des valeurs discrètes, suivant le sens de la déviation.

Pour mesurer la distance entre les avions, l'intervalle de temps sur lequel est menée l'optimisation est successivement subdivisé. Ensuite, on recherche, sur l'ensemble des pas de temps, les séparations minimales et maximales entre les positions possibles des avions, correspondant aux intervalles de manœuvres, à chaque pas de temps. Il y a alors trois cas de figure :

- La violation de la contrainte est certaine. Dans ce cas, aucun point de l'intervalle est admissible et l'intervalle est éliminé,
- La violation de la contrainte est impossible. Dans ce cas, l'intervalle est dit *admissible*,
- Elle est indéterminée. Il faudra alors de nouveau vérifier la contrainte, après le prochain découpage de l'intervalle de temps de manœuvres en deux.

Il existe deux approches pour appliquer l'algorithme B&BI : l'approche globale qui consiste à traiter tous les avions en même temps, et l'approche séquentielle qui les traite dans un ordre pré-établi, chaque optimisation tenant compte des trajectoires déjà optimisées comme étant fixées.

Pour l'**approche globale**, avec un problème à n avions sur un horizon temporel t_f, l'espace de recherche est E^n_{opt} où

$$E_{opt} = [0, t_f] \times [0, t_f] \times \{\alpha, -\alpha\}$$

Le but de cet algorithme est de minimiser une fonction f^n_{opt}, sur un sous-ensemble D^n_{opt} de l'espace de recherche, qui permet de définir des trajectoires d'évitement. Il s'agit du sous-ensemble D^n_{opt} où

$$D_{opt} = \{(t_0, t_1, \bar{\alpha}) \in E_{opt} | t_0 \leq t_1, \bar{\alpha} \in \{\alpha, -\alpha\}\}.$$

On dira qu'un point de D^n_{opt} respecte les contraintes de séparation si, en suivant les trajectoires définies par ce point, les n avions restent séparés deux à deux sur tout l'horizon de temps de résolution. Dans l'approche de Médioni, la valeur de la fonction f^n_{opt} en un

point de D^n_{opt} traduit son respect ou non des contraintes de séparation. Plus précisément, pour un point $t_r = ((t^1_0, t^1_1, \alpha^1), .., (t^n_0, t^n_1, \alpha^n)) \in D^n_{opt}$:

$$f^n_{opt}(t_r) = \begin{cases} \sum_{i=1}^{n}(t^i_1 - t^i_0) & \text{, si } t_r \text{ respecte les contraintes de séparation} \\ Gr & \text{, sinon.} \end{cases}$$

où Gr est une constante suffisamment grande : supérieure à la valeur de f^n_{opt} en tout point qui respecte les contraintes de séparation. aux points qui ne respectent pas les contraintes de séparation, soient plus grandes que ses valeurs aux points qui les respectent.

L'algorithme proposé dans [77] manipule une file "*d'intervalles*" notée F. Pour des sens de déviation fixés, les intervalles manipulés par l'algorithme seraient des intervalles de $\mathbf{R}^{2n}$, correspondant aux intervalles de manœuvres de chacun des n avions dont on cherche à générer des trajectoires sans conflit, c'est-à-dire de la forme $T = T^1_0 \times T^1_1 \times .. \times T^n_0 \times T^n_1$. Le domaine sur lequel serait conduite l'optimisation serait $[0, t_f]^{2n}$. Or, on a vu que quand le sens de déviation n'est pas fixé, l'espace de recherche est $E^n_{opt} = ([0, t_f] \times [0, t_f] \times \{\alpha, -\alpha\})^n$. Pour tenir compte du fait que deux sens de déviation sont possibles pour chaque avion, la file F est initialisée non pas avec un seul intervalle $[0, t_f]^{2n}$ mais avec 2^n éléments de la forme $(([0, t_f], [0, t_f], \alpha^1), .., ([0, t_f], [0, t_f], \alpha^n))$ correspondant à une des 2^n combinaison possibles pour les angles de déviation α^i des n avions. Ces éléments qu'on notera intervalles dans la suite ne sont pas à proprement parlé des intervalles, mais des intervalles associés à n sens de déviation. F est donc initialisée au début de l'optimisation avec 2^n intervalles. Les intervalles de cette file sont successivement extraits, puis coupés en deux nouveaux intervalles, pour lesquels on calcule un intervalle image (qui donne, pour un intervalle T, des bornes inférieure et supérieure pour la valeur prise par la fonction-objectif sur cet intervalle) et un estimateur (qui donne, pour un intervalle T, une valeur atteinte en un point quelconque de T par la fonction-objectif). Considérons un de ces deux intervalles ainsi engendrés. Si la borne inférieure de son intervalle image est supérieure au meilleur estimateur rencontré précédemment, on sait que cet intervalle ne peut pas contenir l'optimum. Dans ce cas, l'intervalle est supprimé. Dans le cas contraire, l'intervalle est inséré dans la file F, sauf s'il est admissible. Il est alors réduit à un point (où l'estimateur de cet intervalle a été calculé) et est inséré dans la file d'intervalles-résultats admissibles f_{sat}, à moins d'être de moins bonne qualité que le meilleur point rencontré jusqu'ici par l'algorithme. Dans ce dernier cas il est éliminé. La file d'intervalles-résultats admissibles f_{sat}, est initialement vide.

L'approche globale traite assez rapidement divers cas de conflits à deux avions. Les cas de conflits à trois avions mènent à des problèmes beaucoup plus grands, que cette approche traite de façon satisfaisante en terme de qualité des solutions, mais nécessitent un temps de calcul très important. Cette approche traite des conflits comprenant jusqu'à 7 avions.

La deuxième approche, **l'approche séquentielle**, suit les étapes suivantes : on fixe d'abord un ordre de priorité entre les avions, puis on engendre, pour chaque avion (en suivant cet ordre) une trajectoire d'évitement permettant à l'avion de rester séparé des avions qui le précédent dans l'ordre de priorité. Le principal problème de cette approche est que les résultats obtenus dépendent fortement, pour une même instance de conflit, de

cet ordre fixé à priori dans lequel les avions engendrent leurs trajectoires. En effet, pour un ordre de priorité donné, on peut aboutir à une situation sans solution.

L'approche séquentielle a été testé avec un conflit à 5 avions avec les 120 ordres de priorité possibles, mais seuls 90 ordres de priorité parmi les 120 ont abouti à une bonne solution avec 5 trajectoires sans conflit.

Les méthodes de Bne semblent pas très bien adaptées au problème de résolution de conflits pour plusieurs raisons :

- Elles supposent la simplification du modèle de prévision de trajectoires, afin de pouvoir en donner une expression analytique. Dans le cadre d'une résolution dans le plan horizontal, cela peut être envisageable sur un horizon temporel réduit, mais si l'on veut tenir compte des profils de montée et de descente des avions, cela ne paraît pas réaliste.
- Les méthodes de B&BI ne permettent pas de résoudre des problèmes de grande taille et contraignent l'utilisateur à envisager une approche séquentielle, sous-optimale, dont la qualité du résultat dépend largement de la stratégie initiale d'ordonnancement des avions.

Résolution de conflits par programmation linéaire mixte

Le problème considéré ici, est la résolution de conflits aériens dans le plan horizontal, avec des avions se déplacant à vitesse constante. Pallatino, Feron et Bicchi ont montré dans [84], qu'il était possible de linéariser ce problème, avec des résolutions en vitesses ou des résolutions en cap, et ce, moyennant l'ajout d'un certain nombre de variables booléennes.

L'approche adoptée est une approche d'optimisation numérique centralisée. La résolution des conflits se fait de façon coopérative, entre les différents avions. Les avions sont modélisés par un axe qui indique la direction de mouvement et par un disque dont le rayon est la moitié de la distance de séparation (afin de respecter les conditions de séparation) et dont le centre est la position de l'avion.

Ce problème peut être décliné sous deux formes différentes : un premier problème où il est permis d'agir uniquement sur la vitesse, et un second problème où on ne peut changer que la direction des avions. Dans les deux cas, le problème de résolution de conflit est réécrit sous la forme d'un problème de programmation linéaire mixte, et est résolu par un outil de programmation linéaire mixte en nombre entiers tel que le CPLEX [58].

Dans le cas du problème avec changements de vitesse (*Velocity Change*, VC), les avions volent suivant une direction fixée et peuvent uniquement modifier leurs vitesses, avec la contrainte que cette vitesse reste dans un intervalle borné. Pour un avion i, sa vitesse v_i appartient à $[v_{i,min}, v_{i,max}]$, avec les données $v_{i,min}$ et $v_{i,max}$ qui sont les vitesses minimale et maximale que peut prendre l'avion (pour des vols commerciaux, ces données sont telles que $\frac{v_{i,max}-v_{i,min}}{v_{i,min}} \leq 0.1$).

La modélisation proposée pour le problème VC cherche les valeurs des accélérations et décélérations q_i, pour chaque avion i impliqué dans un conflit, qui permettent de résoudre ce conflit et de satisfaire les contraintes de bornes sur la vitesse. Pour un problème à deux avions, on modélise les données relatives à chaque avion avec un triplet $(x_i, y_i, \theta_i), i =$

$1, 2$ représentant la position et la direction de avion i, un scalaire v_i représentant la vitesse initiale et deux angles l_{12} et r_{12} entre les droites sécantes tangentes aux disques représentant les avions et l'axe horizontal (figure 1.13). Ainsi, la contrainte modélisant la condition de séparation est la suivante :

$$\frac{(v_1 + q_1)\sin(\theta_1) - (v_2 + q_2)\sin(\theta_2)}{(v_1 + q_1)\cos(\theta_1) - (v_2 + q_2)\cos(\theta_2)} \geq \tan(l_{12})$$

ou

$$\frac{(v_1 + q_1)\sin(\theta_1) - (v_2 + q_2)\sin(\theta_2)}{(v_1 + q_1)\cos(\theta_1) - (v_2 + q_2)\cos(\theta_2)} \leq \tan(r_{12}),$$

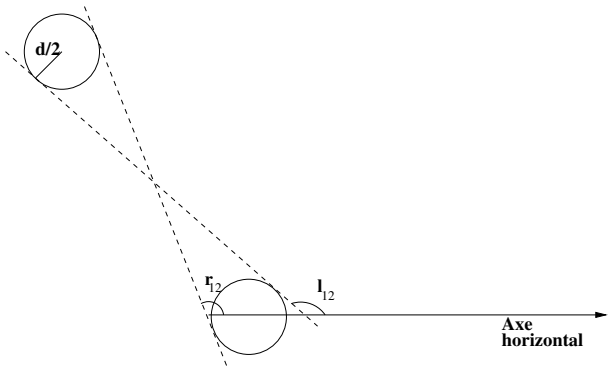


FIGURE 1.13 – Les droites sécantes tangentes aux disques de sécurité de deux avions et les angles l_{12} et r_{12}

Cette condition est l'union de deux demi-espaces (disjonction entre deux contraintes linéaires en q_i) et peut être étendue au cas de n avions, en considérant que les avions doivent être séparés deux à deux. Le but de chaque avion est d'arriver à destination en un minimum de temps tout en évitant les conflits. La fonction coût que l'on cherche à minimiser est $\sum_{i=1}^{n} -q_i$.

La modélisation proposée pour le VC, ne produit pas toujours des solutions, par exemple, quand deux avions arrivent face à face. Un tel inconvénient ne se présente pas lorsqu'on considère plutôt le problème permettant les changements de direction (*Heading Angle Change*, HAC). Pour le problème HAC, les avions volent à vitesse constante et peuvent changer instantanément leurs directions. On cherche ainsi pour chaque avion i, une valeur admissible de l'angle p_i, tel qu'avec les nouvelles directions des vols $\theta_i + p_i$, tous les conflits soient évités.

Comme pour le problème VC, on considère une situation à deux avions pour retrouver les conditions d'absence de conflit qui seront ensuite généralisées au cas de n avions. Ici, deux cas de figures se présentent. Le premier cas correspond à la situation où les deux demi-droites représentant l'extrapolation des trajectoires des avions ne sont pas sécantes (figure 1.14). Dans le deuxième cas de figure, il y a correspond intersection de ces droites. Dans le premier cas, aucun conflit ne peut se produire et on peut déduire de ce cas, un

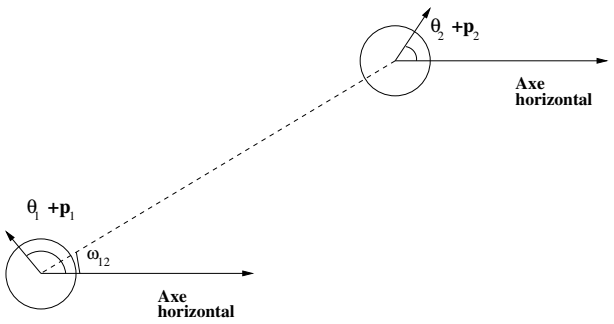


FIGURE 1.14 – Cas où les trajectoires des deux avions ne sont pas sécantes

ensemble de conditions d'évitement sous la forme, ici encore, de disjonctions de contraintes linéaires en p_i. On peut aussi déduire des conditions d'évitement linéaires sous la forme de disjonctions de contraintes en p_i du deuxième cas de figure. L'ensemble de ces conditions peut être résumé par les inégalités suivantes dans le cas n avions :

$$\frac{\theta_i + p_i + \theta_j + p_j + \pi}{2} \leq r_{ij}$$

ou

$$\frac{\theta_i + p_i + \theta_j + p_j + \pi}{2} \geq l_{ij}$$

pour chaque paire d'avions $(i,\ j)$, avec r_{ij} et l_{ij} les angles définis pour la résolution en VC pour les deux avions i et j. La fonction coût que l'on cherche à minimiser dans la problématique HAC est $\max(p_1, ..., p_n)$.

Pour la résolution en VC comme pour la résolution en HAC, les contraintes disjonctives (en "ou") sont reformulées en contraintes en "et" en ajoutant des variables booléennes, afin de formuler ce problème sous la forme d'un programme linéaire mixte avec variables binaires. Ainsi, pour une résolution en vitesse (problème VC), un conflit à n avions engendre un problème à $(2n^2 - n)$ variables et $4 \times \frac{n(n-1)}{2} + 2n$ contraintes. Pour une résolution en cap (problème HAC), le problème engendre $11 \times \frac{n(n-1)}{2} + n + 1$ variables et $35 \times \frac{n(n-1)}{2} + 2n$ contraintes.

En exécutant des tests numériques sur ces deux problèmes, le problème VC se révèle plus rapide à résoudre en termes de temps de calcul, sans parvenir néanmoins à résoudre tous les cas de conflits. La résolution du problème HAC conduit à l'évitement d'un plus grand ensemble de conflits et, de plus, conduit à des trajectoires solutions qui se révèlent moins coûteuse en terme de temps de vol que celle obtenues par la résolution du problème VC. Les deux modélisations ne permettent pas toujours d'obtenir les manœuvres optimales puisqu'elles n'autorisent pas de combiner un changement de vitesse *et* un changement de cap. La combinaison des deux leviers mènerait à un problème d'optimisation *non-linéaire* mixte avec variables binaires, pour lequel il n'existe pas de logiciel suffisamment puissant.

Pour conclure, notons que Pallatino et al. ont ainsi (en considérant les problèmes VC et HAC) pu résoudre des situations de conflits à 15 avions, mais les hypothèses requises

sur les profils des trajectoires permis sont assez contraignantes (avions stables, vitesses constantes, pas de gestion du retour sur trajectoire). Toutefois, ces travaux sont parmi les rares recherches existantes prenant en compte l'aspect combinatoire du problème de résolution de conflit.

Résolution de conflits par colonies de fourmis

Nous présentons ici l'approche adoptée par Olive dans [82], pour un problème de résolution de conflit impliquant n avions dans le plan horizontal et sans considération d'incertitude. Les manœuvres autorisées sont des changements de cap de 10, 20 ou 30 degrés à gauche ou à droite (figure 1.15).

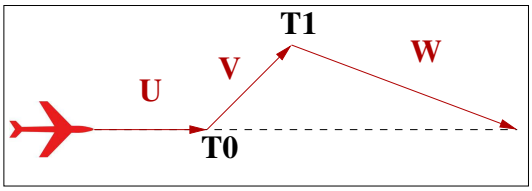


FIGURE 1.15 – La manœuvre d'évitement autorisée

Dans la modélisation proposée par Olive, le temps est discrétisé et la trajectoire d'un avion est modélisée par un graphe. Chaque nœud représente un temps et une position donnée d'un avion. La transition de la position i à la position $i+1$ est représentée par une arête.

L'algorithme de résolution utilisé est un algorithme d'optimisation par colonies de fourmis (*Ant Colony optimization*, ACO). Olive utilise est un ACO classique, tel que présenté par Dorigo dans [30]. La seule différence est que la fourmi est remplacée par n paquets de fourmis représentant les n avions.

L'algorithme se déroule ainsi : la première fourmi choisit son chemin aléatoirement et laisse sur son passage d'autant plus de phéromones que le chemin est court. Les nouvelles fourmis vont choisir leur chemin en fonction de la quantité de phéromones qu'elles trouvent localement. En effet, une fourmi choisit le nœud suivant avec une probabilité qui dépend de la quantité de phéromones qui se trouve sur l'arête qui relie les deux nœuds. Une dissipation des phéromones par évaporation est mise en œuvre afin d'éviter les phénomènes de convergence prématurée. Si n est le nombre d'avions alors un paquet de pn fourmis sont crées pour représenter chaque avion à chaque génération, avec p un paramètre fixé par l'utilisateur (Durand le fixe à 10 dans [37]). Le nombre total des fourmis sera de pn^2. Au début de l'exécution de l'algorithme, une quantité initiale de phéromones est répartie sur l'ensemble du graphe afin d'assurer une probabilité égale pour chaque chemin d'être choisi. Ensuite, un paquet de pn fourmis démarre à partir du nœud correspondant au point de départ de chaque avion. À chaque génération, certaines fourmis parviendront au point de destination de l'avion qu'elles représentent. D'autres, qu'on considérera comme *perdues*, si elles entrent en collision avec d'autres fourmis représentant d'autres avions, ne parviendront pas à atteindre la destination. Une fourmi peut être dans un des trois

états U, V ou W représentés par la figure 1.16. À l'état U, l'avion ne s'écarte pas de sa trajectoire initiale. À T_0, l'avion est à l'état V puisqu'il commence une manœuvre d'éloignement et à T_1 l'avion commence un retour sur sa trajectoire initiale, il est alors à l'état W.

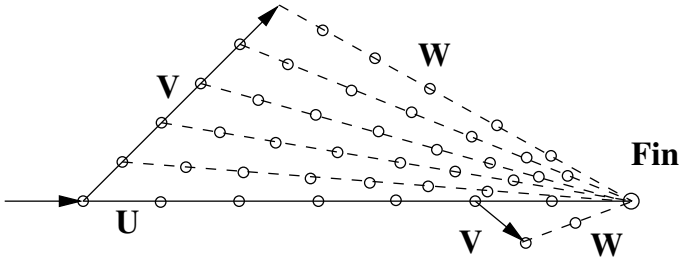


FIGURE 1.16 – Les différents états de la fourmi

On associe à chaque chemin un *score* dont la valeur se cumule tout au long du chemin. Le chemin est d'autant meilleur que le score est plus faible. Si la fourmi est dans l'état U le score ne change pas, si elle est dans l'état V, 2 points sont ajoutés au score et si elle est dans l'état W, 1 point est ajouté au score.

La quantité de phéromones laissée sur un chemin est nulle si il y a un conflit. Dans le cas de non-conflit, elle est égale à $\frac{n-n_{out}}{n} \times \frac{\tau_0}{s_{path}}$, où n_{out} est le nombre courant de fourmis perdues sur ce chemin, s_{path} est le score du chemin et τ_0 est la quantité initiale de phéromone présente sur le chemin. L'algorithme s'arrête quand le score obtenu, par chaque paquet de fourmis représentant chaque avion, ne diminue plus ou si le temps de résolution arrive à terme.

Dans les zones à forte densité, il est parfois difficile de trouver avec ACO une solution, même mauvaise, au problème. Dans ce cas, les contraintes de résolution de conflit sont relaxées dans les premières générations, pour permettre à l'algorithme de trouver des solutions avec un petit nombre de conflits restants. Puis, les contraintes sont progressivement réintroduites en espérant pouvoir résoudre ainsi tous les conflits.

Selon [82], l'algorithme ACO a résolu un cluster de conflits contenant jusqu'à 30 avions, au bout de 65 générations.

1.3 Conclusion

L'étude bibliographique présentée dans ce chapitre a permis de constater la diversité des approches qui ont été tentées pour résoudre le problème de la planification de trajectoire d'avion, tant sur le plan pré-tactique que sur le plan tactique. Une synthèse comparative de ces approches est donnée au tableau 1.1. Il est intéressant de remarquer que les recherches dans ce domaine ont commencé il y a environ 30 ans, avec des moyens financiers très importants, et que les résultats produits ne permettent toujours pas de traiter ce problème dans toute sa complexité.

Beaucoup d'approches ont échoué, car elles ont tenté d'automatiser les tâches effectuées par les contrôleurs aériens. Or ces derniers utilisent principalement la perception visuelle pour traiter les situations de trafic. Cette aptitude à la perception géométrique reste toujours un défit pour l'intelligence artificielle qui tente de reproduire ce comportement au niveau des machines.

D'autres approches considèrent que les trajectoires produites ne seront pas mises en œuvre par un humain (pilote), mais directement injectées dans le FMS. Ceci correspond au contexte SESAR et offre de nouveaux horizons, car l'espace de recherche des solutions est beaucoup plus vaste.

Le problème de résolution de conflits à n avions est un problème fortement combinatoire. Actuellement, seule l'approche stochastique permet de fournir une solution au problème global (sans ordre de priorité entre les avions). Ces méthodes ne peuvent néanmoins pas garantir l'optimalité des résultats obtenus.

En ce qui concerne les approches déterministes qui ont été proposées, bien que les solutions obtenues soient prouvées sans conflit, les trajectoires proposées ne respectent pas les contraintes opérationnelles, comme la régularité des trajectoires et les contraintes sur les vitesses des avions.

Dans le chapitre suivant, nous proposons une nouvelle méthode déterministe, basée sur une analogie avec la propagation lumineuse, qui va s'attaquer, dans le cadre de la planification tactique, au sous-problème de résolution de conflits à n avions, avec un ordre de priorité entre ces avions. Notre méthode s'astreindra au respect des contraintes opérationnelles, afin de produire des trajectoires avion réalistes.

TABLE 1.1 – Synthèse comparative des différentes approches de gestion des trajectoires avions

Approche	Preuve de convergence	Trajectoires opération-nelles	Centralisé / décentralisé	Probabiliste/ déterministe	Taille du problème traité
Algorithmes Génétiques [34]	Non	Oui	Centralisé	Probabiliste	Conflits à 20 avions
AG et Programmation linéaire [78]	Non	Oui	Centralisé	Probabiliste	Conflits à 5 avions
Fonctions de Navigation [92]	oui	Non (Vitesse non régulière)	Centralisé et Décentralisé	Déterministe	Conflits à 5 avions
Forces Répulsives [114]	Non	Non (Vitesse constante)	Centralisé et Décentralisé	Déterministe	Conflits à 10 avions
B&B par intervalles globale [77]	Non	Non (Vitesse constante)	Centralisé	Déterministe	Conflits à 7 avions
B&B par intervalles séquentielle [77]	Non	Non (Vitesse constante)	Centralisé	Déterministe	Conflits à 5 avions
Programmation linéaire mixte [84]	Non	Non (Avion stable, Vitesse constante)	Centralisé	Déterministe	Conflits à 15 avions
Colonies de fourmis [37]	Non	Oui	Centralisé	Probabiliste	Conflits à 30 avions

Chapitre 2

Modélisation du problème et algorithme de résolution

Dans ce chapitre, nous proposons une modélisation des problèmes présentés au chapitre précédent, qui se base sur une analogie avec la physique de la lumière. En effet, sachant que la lumière suit des géodésiques en temps (chemins les plus courts en temps), nous proposons de nous baser sur son comportement pour générer des trajectoires avions optimisant le critère du temps.

Dans ce chapitre, nous présentons d'abord les modèles historiques qui expliquent le phénomène de la propagation de la lumière sur lesquels sont basées des méthodes de calcul des géodésiques, dont on présentera un état de l'art par la suite. Puis, on introduira l'algorithme développé durant cette thèse, pour le calcul des géodésiques dans le contexte particulier de la gestion de trafic aérien. Cet algorithme sera spécialisé pour la planification pré-tactique (cas statiques) et tactique (cas dynamique) des trajectoires avions.

2.1 Modélisation du problème

Dans cette thèse, on cherche à minimiser pour un avion donné le temps de parcours sur sa trajectoire γ. Nous définissons la fonction métrique, c, évoquée au chapitre précédent et qui sera utilisée pour préciser notre fonction objectif sur $\mathbf{R}^3$ ou $\mathbf{R}^3 \times temps$ (suivant le cas statique ou dynamique). La valeur $c(p)$ avec $p \in \mathbf{R}^3$ (ou $\in \mathbf{R}^3 \times temps$), représente le coût de passage par le point p et sera positive. Ce coût est le temps nécessaire pour parcourir une portion élémentaire de l'espace représentée par le point p .

On présente dans cette section, une modélisation du problème de planification de trajectoires dans le domaine du trafic aérien. Cette modélisation est basée sur une analogie avec la propagation de la lumière. On introduit d'abord une *carte d'indice de réfraction*, I, qui sera utilisée pour calculer les valeurs de la fonction métrique c. Cette carte d'indice est une fonction définie sur $\mathbf{R}^3$ (ou $\mathbf{R}^3 \times temps$, suivant le cas statique ou dynamique) avec des valeurs d'indice dans $[1, +\infty[$.

Le phénomène de propagation de la lumière semble être particulièrement adapté pour la recherche de solutions au problème de planification de trajectoires. En effet, on peut

modéliser les zones à éviter (les sous-ensembles que nous avions notés Ω_i) qui représentent les zones congestionnées ou les zones avec une mauvaise météo, par des zones d'indices de réfraction élevés. Rappelons que l'indice de réfraction d'une substance est une mesure de la vitesse de la lumière dans cette substance. Il est exprimé comme le rapport de la vitesse de lumière dans le vide à sa vitesse dans le milieu considéré. Dans le contexte de la modélisation de notre problème, nous attribuerons un indice de réfraction proportionnel au niveau auquel nous souhaiterions pénaliser, dans notre fonction-objectif, les différentes zones que la trajectoire γ est susceptible de traverser. Comme, la lumière est d'autant plus ralentie dans une zone que l'indice qui lui est associé est élevé, la lumière (et donc notre trajectoire) aura tendance à éviter les zones d'indices élevés. La fonction métrique c représente donc le temps nécessaire pour parcourir une portion élémentaire de l'espace d'indice I.

Le même concept peut être utilisé pour *interdire* à une trajectoire de pénétrer dans des zones interdites, tels les volumes de protection des autres avions. Cela peut être fait en attribuant des indices de réfraction infinis à ces volumes de protection. En pratique, on se contentera d'attribuer des indices de réfraction barrières à ces obstacles, c'est-à-dire de valeur beaucoup plus élevée que les autres valeurs de la fonction I. Les zones où il n'y a aucune contrainte se voient attribuer un indice de réfraction égal à 1, ce qui correspond à l'indice du vide, où la lumière se déplace avec la vitesse maximale.

Pour chaque avion dont on cherche à planifier la trajectoire, on introduit une source de lumière au niveau de son point de départ. Cette source émet des rayons lumineux dans toutes les directions de l'espace. Le chemin suivi par le premier faisceau lumineux qui atteint le point de destination est le chemin le plus court que nous cherchons et correspondra à la trajectoire synthétisée de l'aéronef. En effet, selon le principe de Fermat, la lumière suit le chemin le plus court en temps. Cela peut aussi être interprété mathématiquement par le fait que la lumière suit des géodésiques en temps, c'est-à-dire qu'elle utilise un plus court chemin (si il existe) entre deux points dans un espace muni d'une métrique qui correspond au temps de déplacement sur un chemin. De plus, la lumière cherche à éviter les zones qui sont associées à des indices de réfraction élevés (dans notre cas les contraintes souples associées aux zones Ω_i à éviter), car elle est ralentie à l'intérieur de ces zones, et elle ne peut pas traverser des zones d'indices infinis associées à nos contraintes dures de contournement des volumes de protection des autres avions.

Un autre argument justifie la modélisation de notre problème de planification de trajectoires avion par la propagation lumineuse. Outre le fait qu'elle s'accorde avec la fonction-objectif du problème et qu'elle répond aux contraintes d'évitement souples et dures, la trajectoire qui sera ainsi déterminée satisfera aussi nécessairement les contraintes de bornes sur la vitesse de l'avion. En effet, la vitesse de la lumière étant finie et manipulant convenablement l'indice de réfraction du milieu qu'elle traverse, on pourra garantir des solutions avec des vitesses qui restent dans un intervalle donné. Il s'agira tout simplement d'imposer aux valeurs de la fonction I, de rester dans une gamme appropriée. De plus, si on considère que le profil d'indice (valeurs de la fonction I) définit un relief (des "montagnes" représentent les valeurs élevées et des vallées représentent les valeurs faibles), alors on peut montrer (voir [16]) que les courbes géodésiques sont celles qui ont

une courbure minimale sur ce relief. On s'assurera donc ainsi que les trajectoires trouvées par notre approche satisferont bien les contraintes sur la courbure.

Bien qu'on puisse, dans le contexte futur de SESAR, envisager de transmettre des courbes générales au bord afin de dicter une trajectoire aux avions, nous nous restreindrons dans cette thèse à rechercher des *approximations* de courbes géodésiques, représentées par une suite de segments. Cette restriction remporte un avantage en vue de la méthode de calcul de géodésiques que nous envisageons : il nous sera plus facile de rechercher des approximations de géodésiques plutôt que des géodésiques réelles.

On vient de voir que les courbes géodésiques en temps étaient des solutions potentielles à notre problème de planification de trajectoires et que la propagation lumineuse pourrait s'avérer être un bon modèle pour retrouver ces géodésiques. Dans la section suivante, on parcourra les modèles historiques proposés pour expliquer la propagation lumineuse.

2.2 Modèle historique de propagation de la lumière

Historiquement, l'un des aspects les plus "évidents" *a priori* sur les propriétés de la lumière était sa propagation en ligne droite. Cette observation a donné lieu à la théorie corpusculaire selon laquelle la lumière serait constituée de corpuscules se déplaçant en ligne droite. Dans les milieux homogènes, la trajectoire de ces corpuscules est constituée de portions de droites obéissant à un ensemble de lois qui constituent le fondement de l'optique géométrique. Cette théorie ne parvenait cependant pas à expliquer certains phénomènes optiques, comme la diffraction, mise en évidence par Grimaldi en 1665. Cela a conduit Huygens, puis Fresnel et d'autres, à proposer une théorie ondulatoire, selon laquelle la lumière se comporterait plutôt comme une onde. Cette théorie est loin d'être intuitive, car la plupart des phénomènes lumineux que l'on observe ne présentent pas de variation temporelle apparente, ce qui est pourtant le propre d'une onde. Elle s'avérera néanmoins fructueuse pour expliquer à la fois l'optique géométrique et les phénomènes que cette dernière ne parvenait pas à expliquer correctement.

Optique géométrique

L'optique géométrique décrit la propagation de la lumière en introduisant la notion de "*rayons lumineux*", qui en sont les objets de base. Ils obéissent aux règles empiriques suivantes : dans un milieu homogène et isotrope (ou dans le vide), les rayons lumineux sont des portions de droites. De plus, à l'interface entre deux milieux, ces rayons obéissent aux lois de Descartes relatives à la réflexion et à la réfraction.

Les milieux traversés par la lumière sont caractérisés par leur indice de réfraction n, qui intervient dans les lois de Descartes. Les lois de Descartes pour la réfraction s'énoncent ainsi : Soit S une surface séparant deux milieux M_1 et M_2 homogènes et isotropes d'indices respectifs n_1 et n_2 ; soient $\vec{u_1}$ un rayon incident qui se propage dans M_1 et $\vec{u_2}$ le rayon réfracté correspondant, qui se propage dans M_2. On a alors :

– le rayon incident $\vec{u_1}$, le rayon réfracté $\vec{u_2}$ et la normale $\vec{n}$ à la surface S au point d'incidence, forment un unique plan appelé *plan d'incidence*,

– les angles d'incidence i_1 et de réfraction i_2 sont liés par la relation

$$n_1 \sin(i_1) = n_2 \sin(i_2),$$

De plus, la vitesse v de la lumière dans un milieu est donnée par $v = \frac{c}{n}$, avec c la vitesse de la lumière dans le vide et n l'indice du milieu. Ces lois peuvent être retrouvées par le principe de Fermat qui affirme que la lumière se propage d'un point à un autre sur des trajectoires telles que la durée du parcours soit extrémale.

Cela dit, deux phénomènes caractéristiques de la propagation lumineuse restent inexpliqués par l'optique géométrique : les interférences lumineuses et la diffraction. Le premier est bien mis en évidence par l'expérience des fentes de Young. Le deuxième est mis en évidence en faisant passer de la lumière à travers un simple trou circulaire suffisamment petit. Ces deux phénomènes seront expliqués par la théorie ondulatoire de la lumière.

Optique ondulatoire

Huygens a introduit la théorie ondulatoire dans son "Traité de la lumière" [57] paru en 1690. Il y concevait la lumière comme la propagation d'ondes dans un milieu. L'arrivée de lumière en un point y crée une nouvelle ondelette qui se propage de manière sphérique. L'accumulation de ces ondelettes forme alors un nouveau front d'onde, qui se propage à son tour dans le milieu (figure 2.1). Ce principe permet alors d'expliquer qualitativement un grand nombre de phénomènes optiques.

Ces concepts ont été repris par Fresnel, qui a associé à ces ondes une *phase* et une *amplitude*. Cette notion de phase permet d'expliquer les phénomènes d'interférences. Le principe de Huygens-Fresnel s'énonce ainsi : *Tout point sur un front d'onde peut être considéré comme une source isotrope d'une ondelette, dont la phase est égale à celle de l'onde originale et qui se propage vers l'avant à la même vitesse que cette onde. Le nouveau front d'onde formé est l'enveloppe de toutes ces ondelettes (il est tangent à ces ondelettes).*

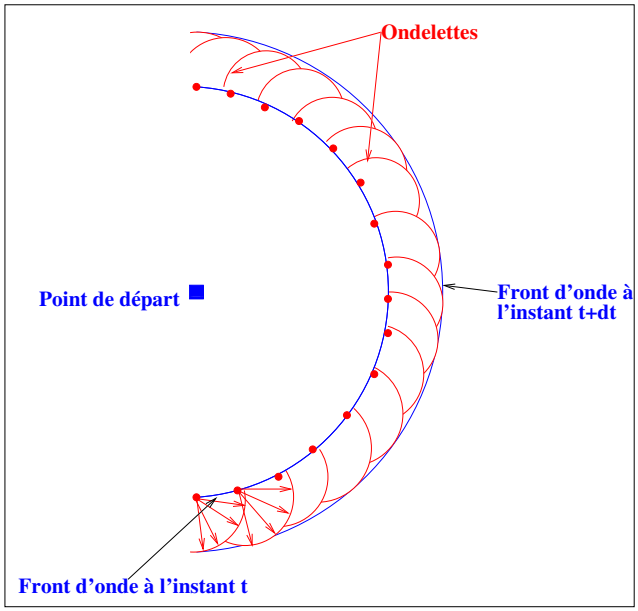


FIGURE 2.1 – Le principe de Huygens-Fresnel utilisé pour déterminer le front d'onde à l'instant $t + dt$, à partir du front d'onde à l'instant t.

Pour avoir plus de détails sur les propriétés de la propagation de la lumière, on peut se référer, par exemple, à [49].

2.3 État de l'art des méthodes de calcul de géodésiques

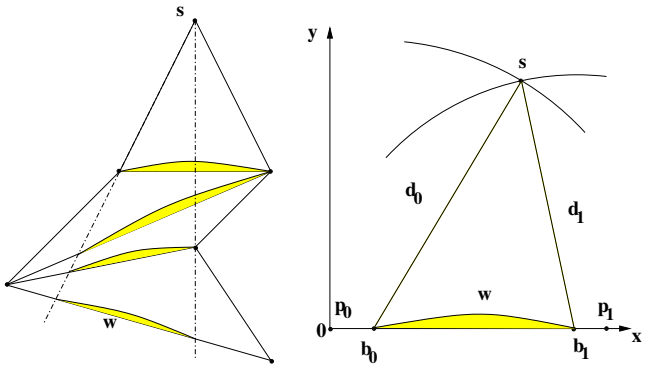
Dans cette section, nous présentons quelques-unes des méthodes, parmi les plus utilisées, pour le calcul des chemins géodésiques.

2.3.1 Algorithme MMP

L'algorithme MMP a été introduit par Mitchell, Mount et Papadimitriou en 1987 dans [75]. Surazhsky et al. [102] ont ensuite proposé une implementation efficace de cet algorithme. Étant donnée une surface plane par morceaux S définie par un maillage triangulaire et étant donné un sommet source $v_s \in S$, l'algorithme MMP calcule la fonction distance géodésique $D : S \to \mathbf{R}$. Cette fonction associe à tout point $p \in S$, la valeur $D(p)$ qui représente la distance géodésique entre p et v_s. Une fois la fonction D complètement définie, on peut retrouver le plus court chemin entre v_s et tout point de S avec un algorithme de "*backtracing*".

L'idée de base de MMP est de regrouper l'ensemble des plus courts chemins qui peuvent être paramétrisés automatiquement. Ce résultat est obtenu en divisant chaque arête du maillage en un ensemble d'intervalles appelés *fenêtres*. Ces fenêtres sont ensuite propagées

à travers les faces de S en se basant sur l'algorithme de Dijkstra [29], qui est fameux en recherche opérationnelle pour le calcul du plus court chemin sur un graphe.



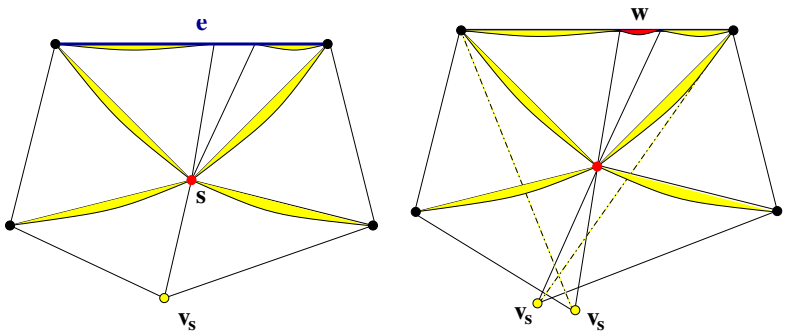
(a) (b)

FIGURE 2.2 – (a) Faisceau de rayons issus du nœud s passant par la fenêtre w à travers une séquence de triangles non dépliés. (b) La position de la source s relativement à la fenêtre w appartenant à l'arête $\overrightarrow{p_0p_1}$.

Pour calculer le plus court chemin sur une fenêtre, on considère le plus court chemin du sommet source v_s à un point p appartenant à une arête e. On suppose dans un premier temps que ce chemin ne passe pas par un sommet *selle* (dont le total des angles dont il est sommet est supérieur à 2π). Dans ce cas, lorsque toutes les faces par lesquelles passe ce chemin sont dépliées dans un plan unique, le chemin forme une ligne droite. On considère l'ensemble des points voisins de p sur l'arête e dont les plus courts chemins vers la source passent par la même séquence de faces. Ces chemins forment un faisceau de lignes droites émanant du sommet source quand les faces concernées sont dépliées et sont représentées par une fenêtre w de l'arête e comme le montre la figure 2.2(a). Le champ de distance $D(p)$ pour la fenêtre w est représenté comme suit. Les extrémités de la fenêtre sont stockées en utilisant des valeurs paramétriques scalaires $b_0,\ b_1\ \in\ [0, \|e\|]$ qui mesurent la distance le long de l'arête, avec $\|e\|$ la longueur de l'arête e. Ensuite, la position du sommet source (relativement à la fenêtre en déroulant les faces concernées dans un plan) est codée à l'aide de distances d_0, d_1 par rapport aux extrémités de la fenêtre. Enfin, une direction binaire τ (en haut ou en bas) est codée pour représenter le coté de l'arête dans lequel le sommet source se trouve. À partir du tuple $(b_0, b_1, d_0, d_1, \tau)$, on peut retrouver la position du sommet source comme le montre la figure 2.2(b).

On considère, à présent, le cas où le plus court chemin passe par un ou plusieurs sommets selle. Soit s le sommet selle le plus proche de w. L'ensemble des plus courts chemins des points appartenant à w forme, dans le plan déplié entre s et e, un faisceau de lignes droites émanant de s comme le montre la figure 2.3. Ces plus courts chemins partagent le même trajet entre s et v_s. Le champ de distance est, dans ce cas, caractérisé par la position de ce sommet *pseudosource* s relativement à e et par *la distance pseudosource* $\sigma = D(s)$ entre s et v_s.

Étant donné une fenêtre w sur une arête e_1, son champ de distance est propagé à travers une face f adjacente pour définir de nouvelles fenêtres sur les deux arêtes opposées



(a) (b)

FIGURE 2.3 – (a) Projection orthogonale d'un maillage 3D à proximité d'un nœud selle (rouge). Une partie de l'arête supérieur e n'est pas visible si on propage des rayons en ligne droite à partir de v_s. (b) Le même maillage est déplié dans le plan du triangle supérieur (dont l'un des sommet est s et l'une des arêtes est e). Il en résulte deux images de v_s. Tous les plus courts chemins à partir de v_s vers la fenêtre w (rouge) passent par la pseudosource s.

e_2 et e_3. On calcule ainsi la façon dont le faisceau de lignes droites est étendu à une face supplémentaire dans le plan déplié. Soit w' la nouvelle fenêtre sur l'une des deux arêtes opposées comme le montre la figure 2.4(a, b). Pour définir le champ de distance sur w', les rayons issus de la pseudosource s sont étendus aux extrémités de w jusqu'à la nouvelle arête, pour obtenir le nouvel intervalle $[b'_0, b'_1]$. Ensuite, les distances d'_0, d'_1 entre ces nouvelles extrémités et la pseudosource s sont calculées. La distance pseudosource reste inchangée $\sigma' = \sigma$.

L'algorithme MMP propage l'information de distance à partir du sommet source en se basant sur l'algorithme de Dijkstra. Quand les fenêtres sont créées, elles sont placées dans une file de priorité triée en fonction de la distance à la source. Quand une fenêtre est retirée de la file, elle est propagée sur une face adjacente. Quand une arête contient déjà une fenêtre, l'intersection entre la nouvelle et l'ancienne fenêtre est calculée pour retrouver le champ de distance minimale associé.

2.3.2 Algorithme *fast marching*

L'algorithme de *fast marching* a été introduit en 1995 par Sethian dans [100]. Cet algorithme vise à retrouver une solution approximative, pour l'équation Eikonal [1] de propagation d'onde

$$\|\nabla u\| = F(x, y) \in \Omega, F(x, y) > 0$$

sur un maillage, où ici, Ω est un sous ensemble de $\mathbf{R}^2$ (mais les résultats restent valable pour $\mathbf{R}^n$), F désigne la vitesse à laquelle l'onde se propage en fonction de la position (x,y) et u l'onde que l'on cherche à retrouver.

1. L'équation Eikonal est une équation aux dérivées partielles du premier ordre, dites d'Hamilton-Jacobi.

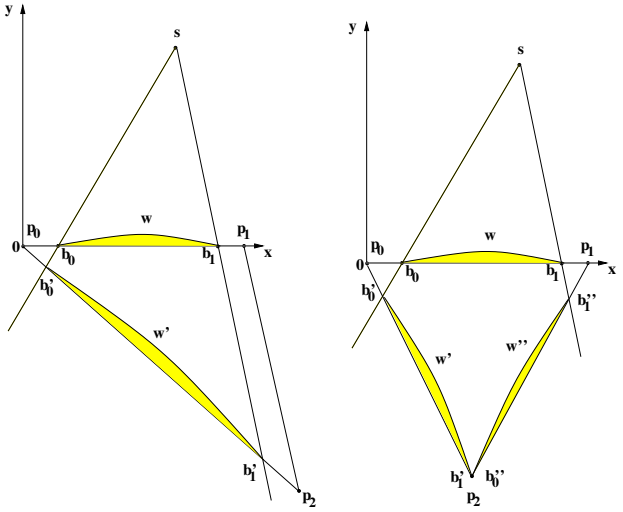


FIGURE 2.4 – (a) Propagation d'une fenêtre w dont résulte une seule fenêtre w' sur l'arête $\overrightarrow{P_0P_2}$. (b) Propagation d'une fenêtre w dont résulte deux fenêtres w' et w'' sur les arêtes opposées $\overrightarrow{P_0P_2}$ et $\overrightarrow{P_2P_1}$.

Maillage orthogonal

L'algorithme de fast marching a d'abord été proposé pour résoudre l'équation Eikonal sur un maillage orthogonal. Il utilise le principe de Huygens de propagation de front d'onde et l'algorithme de plus court chemin sur un graphe Dijkstra. La solution u de l'équation Eikonal peut être retrouvée par le principe de Huygens ainsi : des fronts d'onde circulaires sont lancés à partir des points de la frontière actuelle de l'onde, avec un rayon proportionnel à F. L'enveloppe de ces fronts d'onde est ensuite utilisée pour générer un nouvel ensemble de points, le processus est répété jusqu'à obtenir la solution Eikonal. L'algorithme de fast marching imite ce comportement et utilise un maillage pour retrouver une solution d'une version discretisée de l'équation Eikonal :

$$\left[\begin{array}{l} \max(D_{ij}^{-x}u, -D_{ij}^{+x}u, 0)^2 + \\ \max(D_{ij}^{-y}u, -D_{ij}^{+y}u, 0)^2 \end{array} \right]^{\frac{1}{2}} = F_{ij} \tag{2.1}$$

avec les opérateurs de différence $D_{ij}^{-x}u = \frac{u_{i,j}-u_{i-1,j}}{h}$ et $D_{ij}^{+x}u = \frac{u_{i+1,j}-u_{i,j}}{h}$ et où $u_{i,j}$ désigne la valeur du nœud de coordonnées (i,j) et h l'espacement unitaire du maillage. La même désignation est utilisée pour les opérateurs de différence D^{-y} et D^{+y} sur l'autre direction de coordonnées.

L'algorithme se déroule ainsi : il part d'un ensemble de nœuds appelés nœuds *acceptés*, où la valeur de la fonction u est connue (il s'agit de la condition initiale du problème). Ensuite, tous les nœuds du maillage qui sont des voisins directs des nœuds acceptés, sont mis dans un ensemble "*Proche*" et la valeur potentielle de la fonction u en ces points est calculée. Les nœuds restants sont mis dans un ensemble "*Loin*". La valeur de u pour les nœuds de cet ensemble est fixée à $+\infty$. Puis, la phase de calcul s'exécute ainsi :

1. Début de la boucle : soit p le nœud de l'ensemble *Proche* qui a la plus petite valeur de la fonction u. Le nœud p est ajouté à l'ensemble *Accepté* et retiré de l'ensemble *Proche*.
2. Tous les voisins du point p, qui appartenait à l'ensemble *Loin*, sont enlevés de cet ensemble et ils sont mis dans l'ensemble *Proche*.
3. la valeur de u est recalculée pour tous les nœuds voisins du nœud p qui appartiennent à l'ensemble *Proche* en résolvant l'équation 2.1, avec uniquement les valeurs des nœuds appartenant à l'ensemble *Accepté*.
4. Retourner au début de la boucle.

Pour le calcul de la solution de l'équation 2.1 en un nœud de coordonnées (i, j) du maillage, on renomme la valeur de la fonction u au niveau de ses nœuds voisins : $u_a = u_{i-1,j}$, $u_b = u_{i+1,j}$, $u_c = u_{i,j-1}$, $u_d = u_{i,j+1}$ comme le montre la figure 2.5. Pour prendre

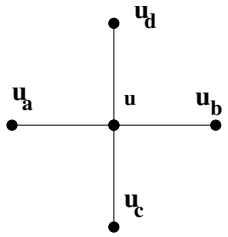


FIGURE 2.5 – Notation pour le calcul de la valeur en un nœud u du maillage orthogonal en fonction des nœuds voisins.

en compte, par exemple, la contribution des nœuds a et c dans la valeur $u_{i,j}$, on résout l'équation :

$$(u - u_a)^2 + (u - u_c)^2 = h^2 \times F_{i,j}$$

De même, on résout aussi les équations associées à toutes les autres combinaisons possibles de paires de nœuds parmi les nœuds a, b, c et d appartenant à l'ensemble *Accepté* (dont les valeurs sont déjà connues). La plus petite valeur obtenue est celle qui sera retenue pour le nœud (i,j).

Maillage triangulaire

Un maillage triangulaire permettant de trouver des approximations de géodésiques plus précises qu'avec un maillage orthogonal, une première adaptation de l'algorithme de *fast marching* au cas des maillages triangulaires a été proposé par Kimmel et Sethian dans [63]. Elle a ensuite été améliorée par Novotni [79], puis par Tang [103].

L'algorithme se déroule en deux phases : une phase d'initialisation et une phase de calcul. La phase d'initialisation se déroule ainsi (figure 2.6) :

– Tous les nœuds sont initialisés avec une valeur de distance nulle.
– Les nœuds sont affectés à trois ensembles : *Accepté*, *Proche* et *Loin* comme suit :

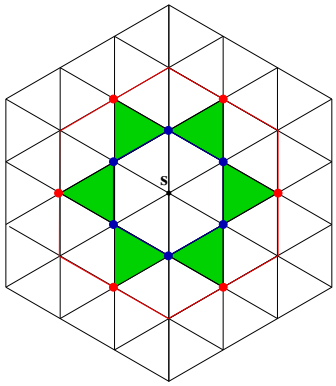


FIGURE 2.6 – Phase d'initialisation. Le nœud central est le point de départ s. Ainsi, les nœuds du *premier anneau* (anneau bleu) et s sont placés dans l'ensemble *Accepté* ; les nœuds rouges appartenant au *second anneau* autour de s sont placés dans l'ensemble *Proche* ; les autres nœuds sont placés dans l'ensemble *Loin*.

  - Si le point de départ est un nœud du maillage, les nœuds appartenant à son premier anneau et lui même sont affectés à l'ensemble *Accepté*,
  - Si le point de départ est à l'intérieur d'un triangle, les nœuds de ce triangle sont placés dans l'ensemble *Accepté* ;
- Les valeurs des distances des nœuds appartenant à un triangle avec deux sommets acceptés sont calculées et ces nœuds sont placés dans l'ensemble *Proche*,
- Tous les autres nœuds sont placés dans l'ensemble *Loin*.

Tous les nœuds de l'ensemble *Proche* sont placés dans une pile triée suivant la valeur de distance de ces nœuds. La phase de calcul se déroule comme pour le maillage orthogonal.

Pour calculer la distance géodésique à partir du point de départ s à un nœud donné v_i, on procède comme suit. Supposons que sont connues les distances réelles, notées d_{i+1} et d_{i+2}, entre s et les deux autres sommets du triangle formé par les sommets v_i, v_{i+1} et v_{i+2}. Un *point de départ virtuel* v_s (figure 2.7) est utilisé. Ce point de départ virtuel est défini comme étant l'intersection, la plus éloignée du nœud v_i, de deux cercles de centre v_{i+1} (respectivement v_{i+2}) et de rayon d_{i+1} (respectivement d_{i+2}). Suivant la situation, la distance géodésique (entre s et v_i) est calculée en fonction de l'une des formules suivantes (où on note $\|uv\|$, la distance entre les points u et v) :

- $\|v_i v_s\|$, si v_s est entre v_{i+1} et v_{i+2} (ce cas est représenté par la figure 2.7),
- $d_{i+1} + \|v_i v_{i+1}\|$, si v_s est à gauche de v_{i+1} ,
- $d_{i+2} + \|v_i v_{i+2}\|$, si v_s est à droite de v_{i+2}.

Durant la phase de calcul de la distance géodésique, deux nœuds (appelés *nœuds parents*) utilisés pour le calcul de la distance géodésique sont enregistrés pour chaque nœud. Pour retrouver le chemin géodésique vers un nœud donné, une méthode de *backtracing* est utilisée à partir de ce nœud en utilisant ses nœuds parents et son point de départ virtuel.

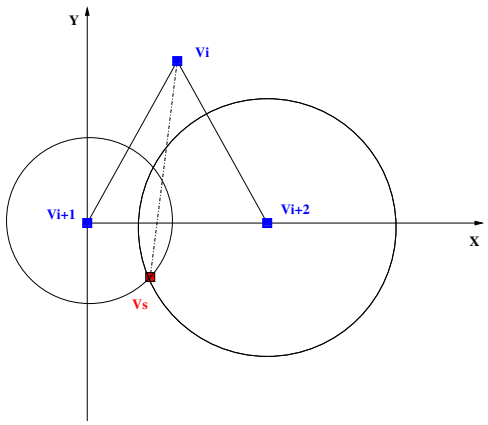


FIGURE 2.7 – Point de départ virtuel. Le nœud courant v_i et ses deux nœuds parents v_{i+1} et v_{i+2}. Dans cet exemple la longueur de la ligne en pointillé est égale à la distance géodésique calculée pour le nœud v_i.

2.3.3 Conclusion

Pour appliquer les algorithmes vus dans cette section, il faut d'abord créer un maillage sur une surface, et en particulier sur une surface contenue dans $\mathbf{R}^3$. Cependant, ces algorithmes ne permettent pas de rechercher une géodésique dans l'espace $\mathbf{R}^3$ tout entier. Une telle recherche exigerait la construction d'un maillage tridimensionnel complet, ce qui serait d'une complexité rédhibitoire. D'autre part, bien que ces algorithmes permettent de considérer des obstacles statiques (containtes dures) en les modélisant par des trous dans le maillage, ils ne permettent pas de prendre en compte des obstacles dynamiques ou des zones pénalisées (contraintes souples). Un autre inconvénient de certains de ces algorithmes est qu'ils ne permettent pas de prendre en compte des vitesses de mobile variables.

Dans la section suivante, nous proposons un algorithme original pour le calcul des géodésiques qui évite ces inconvénients.

2.4 Algorithme de propagation de la lumière (APL)

Comme mentionné précédemment, la trajectoire géodésique suivie par la lumière représente potentiellement une solution au problème de planification de trajectoire (une telle trajectoire est la plus courte en temps). Dans cette section, nous proposons un algorithme heuristique de type séparation et évaluation [9] (*branch-and-bound*, B&B), notée APL, qui a pour but de calculer approximativement des géodésiques. Ces géodésiques doivent éviter des obstacles fixes ou mobiles. Nous présentons donc les adaptations de cette idée de trajectoire suivie par la lumière pour chaque type d'obstacle.

2.4.1 APL avec obstacles statiques

Supposons que l'on recherche numériquement le chemin suivi par la lumière entre deux points dans l'espace $\mathbf{R}^n$: le point de départ, s, et le point de destination, d, avec une fonction d'indice, I, donnant pour chaque point de $\mathbf{R}^n$ une valeur d'indice de réfraction dans $[1, +\infty[$. Notons que dans le cas de notre application ATM, nous ne nous intéresserons qu'aux cas particuliers où $n = 2$ ou $n = 3$. Notre idée consiste à introduire une source de lumière au point de départ, et de simuler la propagation de la lumière provenant de cette source en se basant la théorie de la propagation ondulatoire de la lumière proposée par Huygens. Le chemin suivi par le rayon lumineux le plus rapidement arrivé en d sera la géodésique retenue.

Nous proposons ici un algorithme approximatif de propagation de front d'onde dans $\mathbf{R}^n$, que nous appelons *algorithme de propagation de la lumière* (APL). Le front d'onde lumineux est émis à partir du point de départ s et est discrétisé avec des pas angulaires dans la dimension spatiale. Il se propage dans la dimension temporelle avec un pas de temps dt.

Nous mettons en œuvre l'APL avec un B&B, qui est un cadre classique pour la résolution des problèmes d'optimisation discrète. L'algorithme B&B représente l'ensemble des solutions admissibles, par la racine d'un arbre d'énumération. Des procédés constructifs pour obtenir des bornes inférieure et supérieure pour la valeur optimale de la fonction-objectif sont d'abord appliqués à la racine. Si ces deux bornes sont égales, alors la solution optimale est trouvée, et l'algorithme est arrêté. Sinon, l'ensemble des solutions est partitionné en plusieurs sous-problèmes (nouveaux nœuds fils). Ce processus est appelé *branchement*. La méthode est ensuite appliquée de manière récursive sur les sous-problèmes correspondants, générant ainsi un arbre. Si une solution optimale est trouvée pour un sous-problème alors cette solution partielle est réalisable mais pas forcément optimale pour le problème initial. En se basant sur les bornes, les solutions réalisables sont utilisées pour éliminer des solutions partielles, réduisant ainsi la taille de l'arbre de recherche (on coupe des branches). La recherche se poursuit jusqu'à ce que tous les nœuds soient explorées ou éliminés.

Comme le but de cet algorithme est de trouver une trajectoire qui soit une approximation de géodésique, représentée par une suite de points (ou de segments reliant ces points), le terme nœud désignera dans la suite indifféremment l'un des points qui représente cette trajectoire ou l'un des nœuds de l'arbre de B&B.

Bien qu'utiliser à la fois des bornes inférieures et supérieures soit l'idéal pour l'exécution d'un algorithme B&B, on peut se contenter d'utiliser des bornes inférieures (pour un problème de minimisation), lorsque l'on ne peut pas calculer une borne supérieure. Ainsi, pour la mise en œuvre du B&B dans le cadre de notre modèle de propagation de la lumière, nous proposons de calculer une borne approximative inférieure pour un nœud donné. Comme la trajectoire recherchée doit relier le point de départ au point destination avec un temps de parcours minimal, la borne approximative inférieure que nous proposons représentera une heuristique pour le calcul du temps de parcours minimal d'une trajectoire qui passe par ce nœud. Cette borne, notée *approxLB*, est la somme des deux termes tel qu'illustré sur la figure2.8 :

$$approxLB := TimeToNode + TimeToDest$$

où le premier terme, "*TimeToNode*" est le temps requis (calculé) pour atteindre le nœud courant depuis l'origine. Le second terme, "*TimeToDest*", représente le temps restant (estimé) pour atteindre la destination depuis le nœud courant. Cette durée, *TimeToDest*, est une somme pondérée de deux termes : "*integralTime*" et "*maxSpeedTime*". Le premier terme, *integralTime*, est le temps nécessaire pour atteindre la destination en considérant l'indice de réfraction le long de la route directe, notée γ_{direct}. La vitesse du parcours sur γ_{direct} pour ce calcul dépend de l'indice rencontré le long de cette route ($vitesse = \frac{v_{max}}{indice}$, où v_{max} est la vitesse correspondant à un indice de réfraction égale à l'unité). Le second terme, *maxSpeedTime*, est le temps nécessaire pour atteindre la destination le long de γ_{direct} avec la vitesse maximale. *TimeToDest* est donc donnée par la formule (2.2) :

$$\begin{aligned} TimeToDest &:= \alpha * integralTime + (1-\alpha) * maxSpeedTime \qquad (2.2) \\ &= \alpha \int_{u_{node}}^{u_d} c(I(\gamma_{direct}(u)))du \; + \; (1-\alpha)\frac{\|\gamma_{direct}(u_d) - \gamma_{direct}(u_{node})\|}{v_{max}} \end{aligned}$$

où α est un paramètre de pondération défini par l'utilisateur ; u est l'abscisse curviligne ; u_{node} et u_d sont les abscisses curvilignes du nœud courant et de la destination ; I est la fonction d'indice et c est la fonction métrique de l'espace qui représente le temps nécessaire pour parcourir une portion élementaire de l'espace d'indice I.

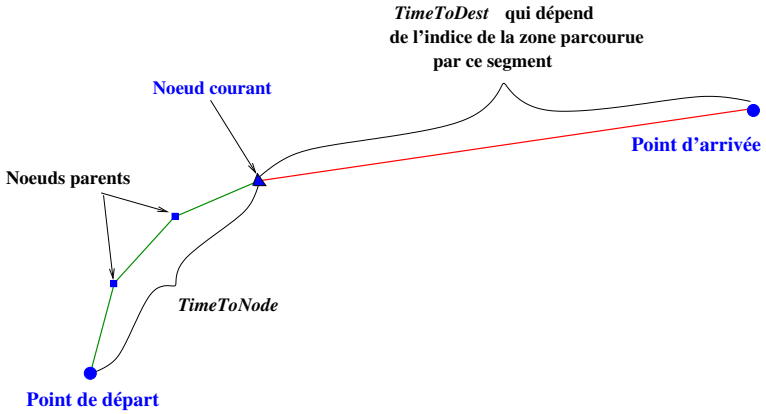


FIGURE 2.8 – Calcul de la borne approximative inférieure pour le nœud courant. Elle est calculée en additionnant le temps nécessaire pour atteindre le nœud courant et le temps pour atteindre la destination à partir de ce nœud sur une trajectoire directe (en ligne droite). Ces temps sont calculés en tenant compte de la valeur de l'indice rencontré.

La propagation de front d'onde, basée sur une discrétisation de l'espace, sera donc assuré par le processus de branchement de l'algorithme B&B. La méthode consiste à considérer qu'une source émet des rayons lumineux rectilignes à partir du nœud courant. Afin de limiter les calculs, les directions d'émission des rayons lumineux sont restreintes

au demi-espace entre le nœud courant et le point destination. En effet, les opérations standardisées des avions ne permettent pas de voler en sens inverse de la destination, mis à part dans le voisinage des aéroports de départ et d'arrivée (région de contrôle terminale, TMA). Ces rayons sont donc lancés dans toutes ces directions en fonction des angles de lancement θ_1, $\theta_2 ... \theta_{n-1}$ de la base de coordonnées sphériques de $\mathbf{R}^n$, et des pas de discrétisation angulaires : $d\theta_i$, $i = 1, 2, .., n-1$ respectivement. Chaque rayon se propage dans l'une des directions résultantes avec une vitesse qui dépend des indices de réfraction des milieux qu'il traverse. Il atteint ainsi un nœud fils du nœud courant après un pas temporel dt. Ces paramètres algorithmiques $d\theta_i$ et dt sont ajustés par l'utilisateur. Tous les nœuds qui ont la même profondeur dans l'arborescence ainsi créée représentent un même front d'onde (voir figure 2.9 pour un exemple avec $n = 2$).

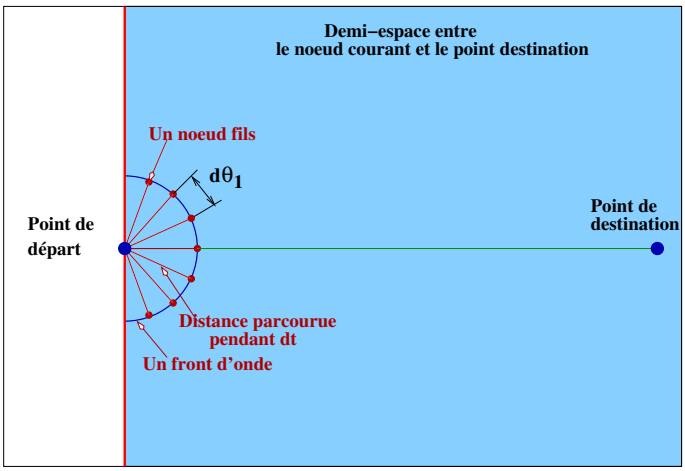


FIGURE 2.9 – Discrétisation de la moitié de l'espace avec un pas temporel dt et un pas angulaire $d\theta_1$.

L'algorithme heuristique de calcul de géodesique APL que nous proposons parcourt l'arbre de recherche pour retrouver une trajectoire dont le temps de parcours est minimal (une approximation de géodésique). La trajectoire résultante sera constituée de la suite de nœuds retrouvée par l'APL et donc de la suite de segments qui relient ces nœuds depuis le point de départ jusqu'au point destination. Comme décrit dans [68, 117, 22], plusieurs stratégies existent pour le parcours de l'arbre de recherche du B&B. En effet, la stratégie de sélection du prochain nœud à explorer (c'est à dire un nœud dont les fils n'ont pas encore été générés) reflète habituellement un compromis entre, maintenir un faible nombre de nœuds explorés dans l'arbre de recherche, et rester dans la limite de la capacité mémoire de l'ordinateur utilisé.

La stratégie du "*best-first search*" (BeFS) consiste à choisir parmi les nœuds inexplorés celui qui a la borne inférieure la plus faible. Avec cette stratégie, il n'y a pas de calcul superflu de bornes, une fois la solution optimale trouvée. Ainsi, le but de cette stratégie est de maintenir un nombre de nœuds explorés le plus faible possible. L'inconvénient est que des problèmes de mémoire peuvent apparaître si le nombre de nœuds critiques, c'est à dire

dont la borne inférieure est plus faible que la valeur de la solution optimal, devient trop grand. Cette situation revient à utiliser une stratégie de "*breadth-first search*" (BFS), où tous les nœuds du même niveau doivent être explorés avant de passer au niveau suivant.

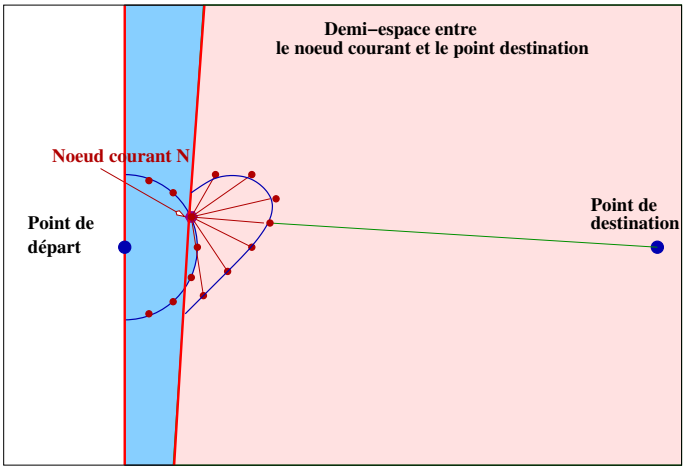


FIGURE 2.10 – Lancement des rayons à partir de nœud courant N.

L'alternative est d'utiliser la stratégie du "*depth-first search*" (DFS) qui consiste à choisir le nœud inexploré le plus profond dans l'arbre. La mémoire requise en termes de nombre de nœuds à stocker en même temps est désormais majorée par le nombre de niveaux dans l'arbre de recherche, multiplié par le nombre maximum de fils d'un nœud, ce qui est habituellement un nombre tout à fait gérable. L'inconvénient est que si la solution courante est loin de la solution optimale, il est possible de calculer inutilement un grand nombre de bornes.

Dans la mise en œuvre d'APL, nous avons choisi une stratégie de parcours de l'arbre de recherche dont la priorité est de trouver rapidement une solution réalisable. Elle combine DFS comme principe général avec BeFS comme critère de sélection secondaire. Elle consiste donc à rechercher les noeuds de profondeur maximum dans l'arbre, et de choisir parmi ces nœuds celui qui comporte la meilleure borne inférieure.

Les figures 2.9, 2.10 et 2.11 illustrent le fonctionnement de l'algorithme dans $\mathbf{R}^2$. La figure 2.9 montre le front d'onde initial à partir du point de départ, s. La figure 2.10 illustre la déformation du front d'onde à partir du nœud courant N, au contact d'un indice non-homogène dans toutes les directions. La figure 2.11 représente la trajectoire résultante consituée de la suite de nœuds ascendants au nœud final. Le nœud final est un nœud "*suffisamment proches*" du point destination et qui a la borne inférieure, définie précédemment, la plus faible parmi tout les nœuds suffisamment proche du point destination. Pour définir la notion de suffisamment proche, l'utilisateur doit fixer une tolérance $\varepsilon > 0$ sur la distance au point destination. Ainsi, un nœud est considéré suffisamment proche de la destination si sa distance au point destination est inférieure à ε.

Afin de décrire précisément l'algorithme APL, nous avons besoin d'une sous-procédure "*LaunchRays(N)*" pour le nœud courant N. Cette sous-procédure est utilisée comme pro-

cessus de branchement pour l'algorithme B&B :

LaunchRays(N, I)
i. Discrétiser le demi-espace entre le nœud N et le point destination avec un pas de temps dt et des pas angulaire $d\theta_i$, $i = 1, 2, .., n-1$ ii. Déterminer les nouveaux nœuds fils ainsi que leurs valeurs en utilisant les règles suivantes : Pour tout rayon lumineux, s'il arrive dans une région d'indice I, sa vitesse dans cette région est $v = \frac{v_{max}}{I}$, avec v_{max} la vitesse maximale. On associe à chaque nœud fils une valeur égale à sa borne inférieure approximative décrite précédemment. iii. Enlever le nœud N de l'arbre et y rajouter ses fils.

Les principales étapes de l'algorithme APL sont donc les suivantes :

APL(s, d, I)
1. Fixer TrajSolution := $\emptyset$. Fixer upperBound := $+\infty$. 2. *LaunchRays(s, I)* (figure 2.9). 3. Tant qu'il reste des nœuds inexplorés dans l'arbre, choisir un nœud N suivant DFS puis BeFS comme décrit précédement. Puis : Si distance$(N, d) \leq \varepsilon$ et valeur du nœud $N \leq$ upperBound alors TrajSolution := Liste des points ascendants qui ont mené à N, upperBound := valeur du nœud N (figure 2.11). Sinon *LaunchRays(N, I)* (figure 2.10).

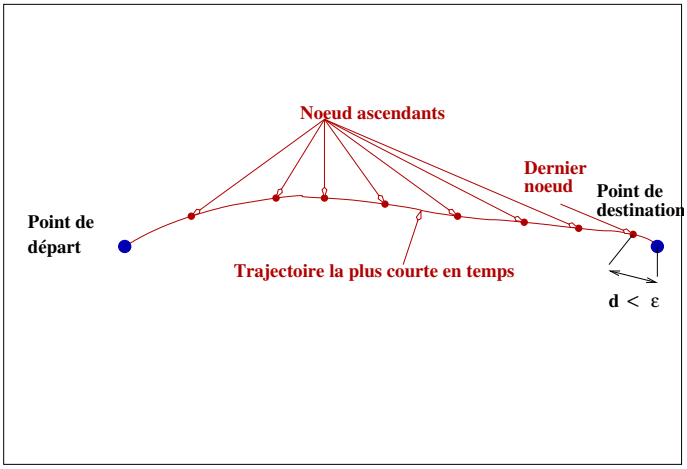


FIGURE 2.11 – La trajectoire la plus courte en temps entre le point de départ et le point destination.

Nous venons de décrire l'algorithme APL dans les cas où l'on cherche le plus court chemin entre deux points de l'espace $\mathbf{R}^n$. Dans le paragraphe suivant, nous allons décrire les adaptations que l'on doit apporter à l'APL pour qu'il puisse rechercher le plus court chemin dans l'espace $\mathbf{R}^n \times temps$.

2.4.2 APL avec obstacles dynamiques

Afin de traiter le problème de recherche du plus court chemin avec des obstacles dynamiques, ce qui est crucial pour la planification tactique dans notre application ATM, on adapte l'algorithme pour prendre en compte la dimension supplémentaire du temps. On recherche donc une approximation de géodésique entre deux points dans l'espace-temps $\mathbf{R}^n \times [0, +\infty[$, sur lequel est défini une fonction d'indice, I, qui associe à chaque point de l'espace-temps $\mathbf{R}^n \times [0, +\infty[$ une valeur d'indice de réfraction dans $[1, +\infty[$. De plus, on considère que les trajectoires suivies par les obstacles (les autres avions dans l'application ATM) est une entrée du problème qui est modélisée par la fonction d'indice, I. En réalité, comme on le verra au chapitre suivant dans l'application ATM, l'APL sera appliqué séquentiellement sur chaque avion dont on cherche la trajectoire. Ainsi, les trajectoires, résultantes des $n - 1^{\text{ième}}$ étapes d'application de l'APL, seront une entrée du problème « rechercher la trajectoire du $n^{\text{ième}}$ avion ».

La dimension temporelle est différente des dimensions spatiales. En effet, la propagation sur cette coordonnée est faite dans une seule direction et dans un seul sens, du passé vers le futur. Dans le cas purement spatial (dans $\mathbf{R}^n$) vu à la section 2.4.1, on devait relier un point de départ, s, à un point destination, d, avec une trajectoire γ. Une manière simple d'étendre l'APL, pour prendre en compte la dimension supplémentaire du temps, est d'ajouter une coordonnée temporelle aux points s et d, et de trouver une trajectoire 4D qui les relie. En forçant les faisceaux lumineux à se propager dans une seule direction dans la dimension temporelle, l'algorithme précédent (dans $\mathbf{R}^n$) peut être directement adapté à ce nouveau système de coordonnées. Cependant, les trajectoires résultantes peuvent ensuite violer la contrainte décrite par l'équation (1.2) du problème général, qui correspond dans le cas de notre application ATM, à la violation des contraintes de bornes sur la vitesse des avions. En effet, si par exemple, dans le cas ATM, le chemin d'origine γ_{old} entre les deux points s et d est une ligne droite avec les dates t_s et t_d associées respectivement à s et d, la nouvelle trajectoire produite par l'extension espace-temps simple de l'algorithme peut être plus longue. Dans ce cas, afin d'atteindre le temps objectif t_d à la destination d, la trajectoire synthétisée peut induire un profil de vitesse qui est incompatible avec les contraintes de vol. Dans certaines autres situations, le chemin généré par l'extension espace-temps simple de l'APL peut être plus court dans la dimension spatiale que γ_{old}, induisant ainsi potentiellement une réduction excessive de la vitesse.

Pour éviter ces problèmes, nous proposons une adaptation de l'APL, qui remplace le temps cible (objectif) par un *segment temporel cible* à la destination, comme illustré par la figure 2.12. Cette modification assure la faisabilité des trajectoires résultantes du point de vue de la contrainte sur la vitesse.

Mis à part, cette adaptation relative à la coordonnée temps, l'algorithme reste iden-

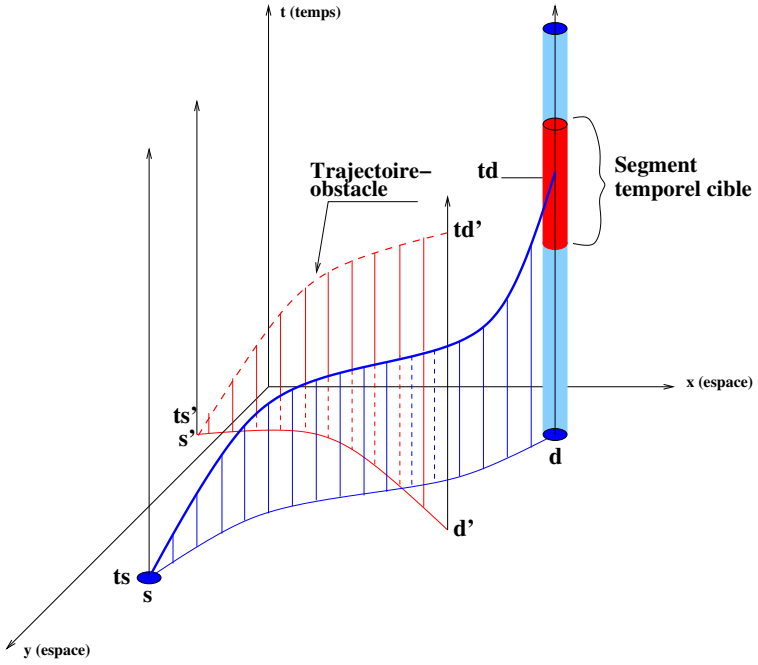


FIGURE 2.12 – Deux trajectoires dans le système de coordonnées $\mathbf{R}^2 \times temps$. La trajectoire en pointillés représente un obstacle en mouvement (un autre avion qui se déplace de s' vers d' et atteint sa destination t'_d). La trajectoire en trait continu est contrôlée par l'APL et vise, à la destination, un *intervalle* de temps contenant t_d (segment temporel cible).

tique à celui proposé dans la section 2.4.1. On le notera par la suite APLd.

Dans ce chapitre, nous avons défini l'algorithme APL qui permet de retrouver une approximation de géodésique (suivant le critère du temps) entre deux points de l'espace $\mathbf{R}^n$ (cas statique) ou $\mathbf{R}^n \times temps$ (cas dynamique). Dans le chapitre suivant, nous allons appliquer les deux versions de l'algorithme dans le contexte de la gestion du trafic aérien. La première version de l'algorithme (statique) dans $\mathbf{R}^2$ sera utilisée pour la planification pré-tactique où l'on doit associer à un avion une trajectoire qui évite des zones considérées statiques. La deuxième version de l'algorithme (dynamique) dans $\mathbf{R}^3 \times temps$ sera utilisée pour la planification tactique où l'on doit associer à un ensemble d'avions dont les trajectoires initiales étaient prévues d'entrer en conflit, de nouvelles trajectoires sans conflit.

Chapitre 3

Applications à l'ATM et résultats numériques

Maintenant que nous avons défini un algorithme pour la génération de géodésiques, nous allons, à présent, l'appliquer pour la planification pré-tactique et tactique des trajectoires avion, avec les adaptations nécessaires pour chaque cas.

L'algorithme APL, défini au chapitre précédent, a été programmé en java. Toutes les expéreinces numériques dont les résultats sont présentés dan ce chapitre ont été conduites sur une machine de 2.33 GHz avec un système d'exploitation Mandriva Linux et une RAM de 1 Go. Dans chacuns des problèmes-tests traités dans ce chapitre, le coefficient de pondération α, dans l'équation 2.2 pour le calcul de la borne inférieure approximative, est fixé à 0.9. De plus, la tolérance ε, nécessaire pour la condition d'arrêt de l'APL, est fixée à 1 Nm.

Pour la mise en œuvre de l'APL dans ce chapitre, dans la phase de branchement (figure 2.10), au lieu de lancer des rayons lumineux dans le demi-espace entre le noeud courant et le point destination, nous choisissons de les lancer dans une surface plus restreinte, notée S (figure 3.1) qui est la projection dans un plan d'un cône de révolution dont le sommet est le nœud courant, dont l'axe de révolution est la droite qui relie le nœud courant au point destination et dont la génératrice fait un angle φ avec l'axe de révolution. Dans tous nos tests, nous avons choisi $\varphi = \frac{\pi}{6}$. Cette restriction empêche les changements de cap soudains de plus de φ et permet à la trajectoire de l'aéronef de rester dans une certaine enveloppe autour de la route directe (rappelons que l'un des objectifs de SESAR est de satisfaire au mieux les demandes des compagnies aériennes et donc d'assigner aux avions de nouvelles trajectoires aussi proches que possible de leurs trajectoires initiales).

3.1 Cadre pré-tactique

Dans cette section, nous allons appliquer la version statique de l'algorithme APL pour le problème de la planification pré-tactique de trajectoires. Bien que le problème-test traité ici soit de nature purement académique, notre algorithme pourra facilement être appliqué à des problèmes réels avec un pré-traitement convenable des données.

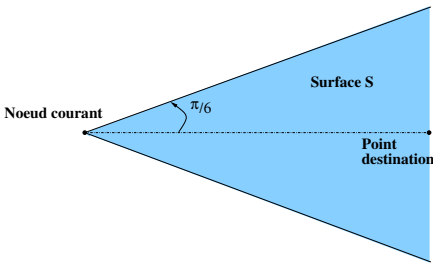


FIGURE 3.1 – Surface restreinte de lancer de rayons S.

3.1.1 Description du problème

Bien que le problème général de la planification pré-tactique soit d'assigner à un avion en-Route (phase de montée, phase de croisière, phase de descente) une trajectoire qui évite des zones congestionnées et des zones avec une mauvaise météo, nous nous contentons ici d'un problème-test mettant en jeu la trajectoire d'un avion en phase de croisière uniquement. Ainsi, au lieu de chercher une trajectoire dans $\mathbf{R}^3$, on cherchera une trajectoire dans $\mathbf{R}^2$, l'altitude étant fixée au niveau de vol de croisière de l'avion concerné.

Nous considérons le problème académique suivant : Déterminer la trajectoire entre un point de départ donné, s, et un point destination donné, d, d'un avion qui évolue à une vitesse de 450 Kt dans un espace aérien carré de $(200 \times 200)\text{Nm}^2$ avec quatre zones Ω_i données, $i \in \{1, .., 4\}$ à éviter.

Nous conduisons nos tests sur cinq instances différentes de ce problème. Les coordonnées des centres des quatre zones Ω_i notées a_i et b_i avec $i = 1, 2, 3, 4$ et les coordonnées des points de départ (s_x, s_y) et des points destination (d_x, d_y) de l'avion pour chaque instance du problème sont présentées dans la table 3.1.

	s_x	s_y	d_x	d_y	a_1	b_1	a_2	b_2	a_3	b_3	a_4	b_4
(a)	39	1	1	39	10	10	28	13	34	18	36	44
(b)	1	1	39	39	20	8	20	28	30	9.2	32	18
(c)	1	1	39	39	20	10	0	38	26	0	20.8	28
(d)	1	1	39	39	16	8	18	22	28	10	32	40
(e)	1	1	39	39	8	10	10	20	36	18	26	19.6

TABLE 3.1 – Caractéristiques des cinq instances-tests représentées par la figure 3.2.

3.1.2 Résultats

Pour tester ce problème, nous utilisons un système de coordonnées qui est échelonné suivant la séparation standard. On utilise ainsi une grille $[0, 40] \times [0, 40]$ dont l'unité est la séparation horizontale standard : 5 Nm.

Notre modélisation requiert que nous associons à chaque instance une fonction d'indice de réfraction I définie sur $[0, 40] \times [0, 40]$ et qui prend des valeurs dans $[1, +\infty[$. Les

contraintes d'évitement de zones Ω_i étant des contraintes souples, nous ne donnons pas de valeur infinie à I. La fonction I que nous avons choisie est la suivante :

$$I(x,y) = \max(1, \sum_{i=0}^{4} e^{-\frac{((x-a_i)^2+(y-b_i)^2)}{k}}),$$

où le coefficient k représente un coefficient de pénalisation associé aux zones à éviter. Dans nos tests, nous avons fixé la valeur du coefficient k à 10. Ainsi, plus l'avion est près du centre de Ω_i, plus l'indice est élevé et de même lorsque k est choisi grand. L'indice de réfraction I tel que défini ici est bien borné inférieurement par 1.

On obtient ainsi pour chacunes des cinq instances-tests les cartes d'indices représentées par la figure 3.2, où les fortes valeurs d'indices sont représentées en rouge et les faibles valeurs, notamment les valeurs d'indice de réfraction égal à 1 (sans pénalisation), sont représentées en bleu.

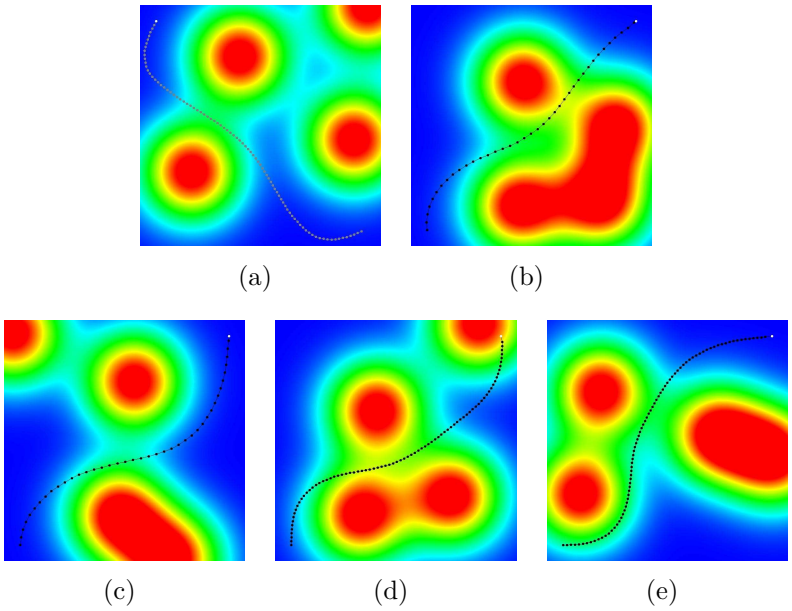


FIGURE 3.2 – Les trajectoires d'avions dans $\mathbf{R}^2$ calculées par l'APL et contournant les zones à éviter.

Pour la mise en œuvre de l'algorithme APL, l'utilisateur doit faire certain choix. Les valeurs de paramètres que nous avons choisies et qui ont conduit aux résultats numériques présentés dans ce chapitre sont motivées par l'expérience de spécialistes de l'ATM ainsi que par de nombreux tests dont nous ne rapportons pas les détails ici. Nous avons fixé l'angle de discrétisation $d\theta_1$ (figure 2.9) à $\frac{\Pi}{36}$. Le pas de discrétisation temporel dt est fixé au temps nécessaire pour voler une demie norme de séparation horizontale (la vitesse de l'avion étant 450 Kt, on a $dt = 20$ secondes. La vitesse maximale v_{max} pour le lancement de rayons est fixée à la vitesse de l'avion, soit 450 Kt.

Les trajectoires solutions calculées par l'APL sont représentées par des courbes pointillées partant du bas vers le haut. Chaque trajectoire a été calculée en moins de 5 secondes

de temps CPU. Comme le montre la figure 3.2, les trajectoires produites par l'APL sont des approximations de géodésiques qui évitent les zones à indice fort et traversent les « vallées » à faible valeurs d'indice. La trajectoire est garantie d'avoir une vitesse bornée inférieurement, ce qui est crucial dans nos applications ATM. Par ailleurs, ces trajectoires sont, par construction, des séquences de segments qui peuvent donc être gérés par le FMS.

Comme nous l'avons mentionné précédemment, bien que le problème test que nous considérons dans ce paragraphe soit académique, l'APL peut être appliqué dans un contexte réel. Les trajectoires devront alors éviter de véritables zones avec une mauvaise météo ou des zones congestionnées. Il faudra alors trouver une fonction I qui modélise bien ces zones et le degré de pénalités qu'on voudra leur associer. De plus, pour être plus réaliste, il sera intéressant de traiter le cas où ces zones se déplacent car en réalité ces zones se déplacent un petit peu sur l'horizon temporel (une à deux heures à l'avance). En fait, ce dernier problème des zones à éviter qui se déplacent, peut déjà être pris en compte par notre algorithme puisqu'on dispose déjà de la version APL-dynamique. L'évaluation pratique de l'APL-dynamique est le sujet de la prochaine sous-section. Pour ce qui est d'expérimenter APL sur des données réelles mettant en jeu une zone météo à éviter, nous reportons cet aspect planification pré-tactique à des travaux futurs.

3.2 Cadre tactique

Dans cette section, nous allons appliquer l'algorithme APL pour la résolution des conflits *entre avions* dans la phase tactique du contrôle aérien. Cette phase correspond à l'action du contrôleur sur son secteur et a une durée moyenne de 20 minutes.

Dans le cas de résolution de conflits, l'APL sera appliqué à *plusieurs* aéronefs. On considère comme entrée du problème un ensemble d'aéronefs en conflit pour lesquels de nouvelles trajectoires sans conflit doivent être conçues. Pour chaque aéronef a impliqué dans la zone de conflit, l'APL générera une nouvelle trajectoire avec un temps de parcours "minimal" (nous reviendrons par la suite sur cette notion de temps de parcours minimal), entre le point d'entrée de l'avion, s_a, et son point de sortie, d_a, de la zone concernée. Les obstacles ne sont plus statiques, comme pour la section précédente, mais dynamiques (deux données du problème). En effet, chaque aéronef doit considérer les zones de protection des autres aéronefs comme des contraintes dures (il est interdit de pénétrer dans la zone de protection d'un autre avion). Les aéronefs étant mobiles, les obstacles sont dynamiques et l'APL doit être appliqué dans l'espace 4D ($\mathbf{R}^3 \times temps$).

Pour traiter ce problème, nous proposons la méthodologie suivante. L'APLd est séquentiellement appliqué à chacun des avions impliqués dans le conflit. Nous assignons d'abord une trajectoire au premier avion sans tenir compte des autres aéronefs (sans la considération de contraintes). Le premier avion maintiendra donc sa trajectoire initiale (avant résolution). Puis, l'APL cherche une trajectoire pour l'aéronef suivant en considérant la zone de protection autour de la trajectoire du premier avion comme une contrainte, et ainsi de suite jusqu'au $n^{\text{ième}}$ avion qui considère les trajectoires des $n-1$ avions précédents comme des contraintes. La trajectoire assignée à un aéronef a à chaque étape de la résolution doit connecter deux points de l'espace 3D (le point de départ de l'avion s_a et

son point de destination d_a) et éviter les trajectoires des autres avions qui sont considérées comme des contraintes fixes (données connues). Bien évidemment, le choix de l'ordre de considération des avions peut conduire à des résultats de qualité variable et donc à des solutions sous-optimales. Dans les tests que nous présentons, il nous a suffi d'un ordre de résolution des avions choisi au hasard pour obtenir de bon résultats. En pratique, certains critères opérationnels peuvent être utilisés afin de sélectionner une séquence particulière (par exemple : règle du premier arrivé, premier servi, certains appareils peuvent avoir une priorité plus élevée, la longueur de trajectoire, etc ..). Ainsi, le premier avion aura une trajectoire qui minimise vraiment le temps de parcours. Les autres avions auront des trajectoires sous-optimales. En effet, cette approche induit une plus grande déviation sur les derniers avions, mais cette caractéristique se retrouve aussi au niveau opérationnel lorsqu'un contrôleur décide de modifier la trajectoire d'un seul avion pour résoudre un conflit. De plus, étant dans la phase tactique, les rallongements induits sont faibles par rapport à la longueur totale de la trajectoire. Ainsi, minimiser le temps de parcours de la trajectoire d'un avion donné dépend de son ordre de résolution par l'APLd.

Voici, plus précisément, comment nous adaptons l'APL au problème de résolution de conflit entre aéronefs :

- Travaillant en 4D ($\mathbf{R}^3 \times temps$), la propagation dans la dimension temporelle est faite exclusivement dans un sens (du passé vers le futur) avec un pas de temps dt. Le segment temporelle cible que nous choisissons est $[t_s, t_d + \delta \times (t_d - t_s)]$, où t_s et t_d sont respectivement les dates associées aux points de départ et destination et où δ est un paramètre de retard toléré, choisi par l'utilisateur.
- Afin de résoudre les conflits en-Route, l'APL autorise uniquement les déviations latérales afin de garder le profil vertical optimal (prescrit en entrée) de l'avion. Les leviers d'action pour assurer l'évitement du conflit sont les changements de cap qui constitueront donc nos variables d'optimisation. Ceci est en adéquation avec la pratique commune en ATM. En effet, pour les aéronefs en-Route, les contrôleurs aériens utilisent en général des changements de cap pour résoudre les conflits afin de minimiser l'impact sur l'efficacité des vols. Les manœuvres verticales ne sont pas utilisées en-Route (elles sont principalement utilisées dans la zone terminale) car elles sont plus coûteuses du point de vue de la consommation de carburant. De plus, les manœuvres verticales sont moins confortables pour les passagers. Nous ajusterons donc dans l'APL la propagation des rayons lumineux dans la dimension verticale de façon à épouser le profil vertical initial de l'avion. Ainsi, l'utilisateur doit uniquement fournir *un* angle de discrétisation, $d\theta_1$, comme si on travaillait dans ($\mathbf{R}^2 \times temps$).
- La vitesse initiale des rayons lumineux émis n'est plus constante (elle était égale à la vitesse maximale, v_{max}, dans la procédure LaunchRays), mais dépend du profil de vitesse spécifique de l'aéronef considéré. Ce profil de vitesse est une donnée du problème. Ce profil donne la vitesse de l'avion à chaque instant. Ainsi, on lance les rayons à partir du nœud courant, avec une vitesse déterminée par le profil de vitesse de l'aéronef à l'instant *TimeToNode* associé à ce nœud.
- À la $n^{\text{ième}}$ étape d'exécution, l'APL synthétise une trajectoire pour le $n^{\text{ième}}$ avion

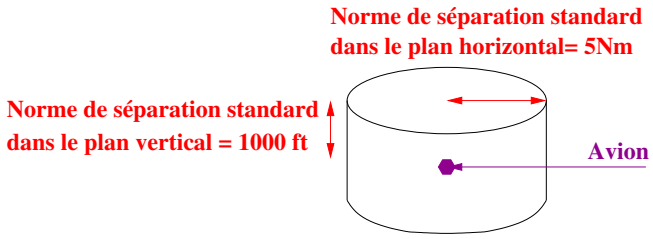


FIGURE 3.3 – Enveloppe de protection cylindrique d'un avion en 3D.

dans l'espace $\mathbf{R}^3 \times temps$ qui doit éviter les tubes 4D représentant les trajectoires des $n-1$ avions précédents (déjà calculés). La section d'un tel tube 4D à l'instant t correspond à l'enveloppe de protection d'un avion (dans $\mathbf{R}^3$). Il s'agit d'un cylindre dont la base est un disque de rayon égal à la distance horizontale de séparation minimale entre deux avions (5 Nm) et dont la hauteur est le double de la distance verticale minimale de séparation (2×1000 ft), comme illustré par la figure 3.3. Afin de garantir des trajectoires avions sans conflit, l'APL élimine directement tout rayon qui pénètre dans un tel tube 4D dans le processus de branchement du B&B.

La fonction indice de réfraction associée au problème de résolution de conflits, doit garantir l'évitement des tubes 4D des avions précédents. Tel que mentionné plus haut, la fonction d'indice, I, prend à une valeur constante élevée (notée I_{max}, fixée par l'utilisateur) à l'intérieur de ces tubes. En tout autre point de $X(t) \in \mathbf{R}^3$, la fonction I prend la valeur minimale unitaire ($I(X(t)) = 1$). Plus précisément, soient $Y_i(t)$, $i = 1, 2, ..., n-1$ les positions à l'instant t des $n-1$ avions déjà traités. À l'étape n, la fonction d'indice I est donnée par la formule suivante pour chaque point $X(t) \in \mathbf{R}^3$:

$$I(X(t)) = \begin{cases} I_{max} & \text{Si } \exists\, i \in \{1, 2, .., n-1\} \text{ tel que } (d_v(X(t), Y_i(t)) \leq 1000\ ft) \\ & \quad \text{et } (d_h(X(t), Y_i(t)) \leq 5\ Nm), \\ 1 & \text{Sinon.} \end{cases}$$

où, en considérant deux points $A = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}$ et $B = \begin{pmatrix} b_x \\ b_y \\ b_z \end{pmatrix} \in \mathbf{R}^3$, on définit les distances verticale et horizontale comme suit : $d_v(A, B) = |b_z - a_z|$ (distance dans le plan vertical) et $d_h = \sqrt{(b_x^2 - a_x^2) + (b_y^2 - a_y^2)}$ (distance dans le plan horizontal).

Nous venons d'exposer les différentes adaptations à prendre en compte dans l'exécution de l'APLd pour résoudre les conflits dans la phase tactique du contrôle aérien. Dans les prochaines sous-sections, nous testons numériquement l'APLd. D'abord, en sous-section 3.2.1, nous présentons les résultats obtenus sur un problème académique dont on connaît la solution. Puis, nous considérons en sous-section 3.2.2 un problème réel de grande dimension. Dans ces deux sous-sections, pour la mise en œuvre de l'APLd, outre les paramètres qui ont été fixés pour tout le chapitre, nous fixons la constante I_{max} de la fonction d'indice à 2 (pour les zones interdites). Nous fixons aussi l'angle de discrétisation $d\theta_1$ à $\frac{\Pi}{36}$ et le paramètre de retard toléré à $\delta = 0.1$.

3.2.1 Problème académique

Dans cette section, on présente un problème académique, qui a été beaucoup traité dans la littérature, appelé *problème du rond-point*. Ensuite, on expose les résultats de l'APLd sur ce problème ainsi qu'une comparaison avec les résultats trouvés par d'autres méthodes.

Description du problème du rond-point

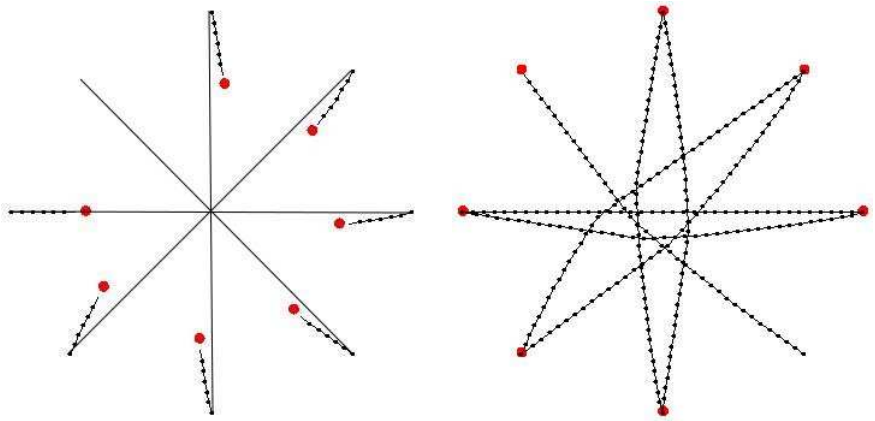
Le problème du rond-point est un problème artificiel impliquant P avions, situés initialement sur un cercle de rayon R donné, à un même niveau de vol (FL) donné, exprimé en centaines de *pieds*, et qui convergent à une même vitesse v (donnée) vers le centre du cercle. Les P avions seront donc en conflit au centre du cercle. Il s'agit d'un problème symétrique et on s'attend à retrouver une symétrie également dans la solution. En effet, la solution, classique pour les contrôleurs aériens, à ce problème consiste en un rond-point avec tous les avions déviés vers la gauche ou tous vers la droite.

Résultats

L'instance particulère de ce problème que nous considérons est caractérisée par les données suivantes : $P = 7$ avions ; un rayon $R = 100$ Nm ; la vitesse des avions est fixée à $v = 450$ Kt et le niveau de vol est FL= 300 (un niveau de vol typique pour un aéronef en croisière).

L'APLd a été séquentiellement appliqué à chacun des avions impliqués dans le conflit. On place les tubes 4D associés aux avions sur une grille 4D dont l'unité dans le plan (x, y) est la distance standard de séparation horizontale et dont l'unité dans le plan vertical est la distance standard de séparation verticale. La vitesse v_{max}, dans la procédure LaunchRays, est fixée à la vitesse des avions 450 Kt. De plus, exactement comme pour la mise en œuvre de l'APL pour le cas statique, on choisit un pas de temps dt égal à la durée nécessaire pour voler une demi-distance horizontale de séparation, soit $2.5Nm$.

La solution calculée par l'APLd pour cette instance du problème est satisfaisante. L'ensemble des trajectoires résultantes a été trouvé en moins de 30s de temps CPU. Le résultat est une situation sans conflit avec le premier avion qui ne dévie pas de sa route initiale tel qu'attendu, alors que tous les autres avions sont déviés vers la gauche, tel que souhaité par les contrôleurs aériens, comme le montre la figure 3.4. Le premier avion est privilégié par la construction même de l'algorithme (il considère qu'il est seul dans l'espace). On retrouve néanmoins le comportement « rond-point » avec les avions suivants. Pour envisager de retrouver un rond-point complet (tous les avions sont déviés de façon parfaitement symétrique), il faudrait envisager une stratégie d'optimisation globale qui agirait simultanément sur la trajectoire de tous les avions. Notre approche séquentielle, considérant un ordre de priorité entre les avions a certes l'inconvénient de produire une solution sous-optimale pour le problème de résolution de conflits, mais elle a l'avantage d'être viable pour des problèmes réels de grande taille, comme on le verra dans la prochaine sous-section.



(a) (b)

FIGURE 3.4 – Résolution de conflit avec 7 avions pour le problème du rond-point. Les droites dans (a) représentent les trajectoires directes des avions (avant résolution). Les trajectoires solutions synthétisées par l'APL sont représentées par des courbes en pointillées.

Comme nous l'avons déjà mentionné, ce problème est une référence classique dans la planification de trajectoires avion. D'autres approches ont été testées sur ce problème et ont trouvé des résultats similaires aux notres. On peut citer l'approche de Durand [35], qui a été appliquée sur une variante de ce problème dans laquelle les trajectoires ne peuvent être modifiées qu'avec des manœuvres d'offset. Durand utilise un algorithme génétique combiné à un algorithme de programmation linéaire pour aborder cette variante du problème. Le résultat obtenu par Durand sur un problème similaire avec $P = 6$ avions est montré sur la figure 3.5. Le premier avion garde une trajectoire directe et tous les autres avions font un offset vers la gauche.

Des résultats semblables ont été trouvés avec $P = 5$ avions par Pallottino et Féron [84] avec une approche en programmation linéaire mixte sur une autre variante du problème du rond point où, cette fois, le cap initial de l'aéronef est modifié par l'angle minimal qui permet d'éviter les conflits. Pour un temps de vol total des aéronef de 15 minutes, l'algorithme est appliqué itérativement tous les $\Delta T = 5$ minutes pour permettre d'atteindre l'objectif initial de chaque aéronef. En effet, une fois le conflit dépassé, le cap qui sera associé à chaque avion sera celui qui lui permettera de revenir sur son objectif initial. Une différence principale entre les résultats de Pallottino et Féron et les nôtres est que dans leur cas, tous les avions en conflit sont déviés (il n'y a pas d'avions privilégiés).

Enfin, citons les travaux d'Olive, Durand et Alliot [30, 37] qui résolvent la même variante que Durand (manœuvres d'offset) de ce problème avec $P = 30$ avions, avec une approche par colonies de fourmis. Leur solution produit encore une fois un comportement de rond-point avec toutes les trajectoires déviées comme le montre la figure 3.6.

Nous avons testé avec succès l'APL sur un problème académique. Dans la section suivante, nous considérons un problème pratique avec des données réelles pour une journée de trafic sur la France.

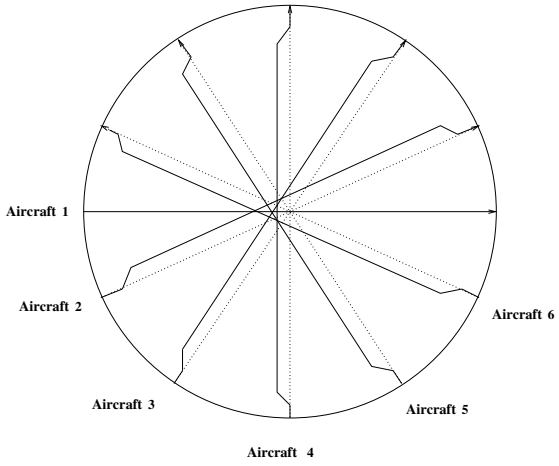


FIGURE 3.5 – Résolution d'une variante du problème du rond-point (avec manœuvres d'offset) par Durand (algorithme génétique et programmation linéaire) pour 6 avions. Ici ce sont les trajectoires solutions qui sont en trait plein.

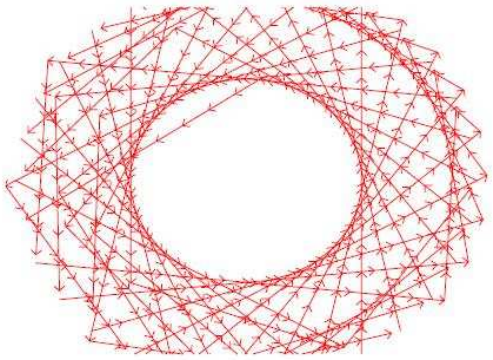


FIGURE 3.6 – Résolution du problème du rond-point par manœuvres d'offset et colonies de fourmis pour 30 avions.

3.2.2 Problème réel : journée de trafic sur la France

Afin de valider l'algorithme avec des données réelles, nous avons d'abord développé une méthodologie, appelée "*LPA-traffic*", qui à partir de trajectoires simulées d'avions pour une journée de trafic sur la France, détecte les conflits, les résout et reconstruit les trajectoires après leur résolution. Les trajectoires, qui sont une entrée de notre méthodologie LPA-traffic, sont issues du simulateur CATS dont on détaillera le fonctionnement au paragraphe suivant.

Le LPA-traffic s'effectue par fenêtres glissantes sur toute la journée de trafic. A chaque pas de simulation t (toutes les δt minutes), la prédiction de trafic a lieu sur un horizon temporel Δt ($\Delta t > \delta t$). La prédiction du trafic consiste à extraire les segments de trajectoires présents entre l'instant t et l'instant $t+\Delta t$. Les valeurs que nous avons choisies pour notre application sont $\delta t = 7$ minutes et $\Delta t = 21$ minutes. Une fois la prédiction de trafic effectuée, on détecte des *clusters de conflits* indépendants. Chacun de ces clusters constitue un sous-problème de résolution de conflits sur lequel sera appliqué l'APLd.

Le choix de 21 minutes pour la durée totale de la fenêtre temporelle est un compromis entre le nombre de conflits qu'on peut trouver dans la même fenêtre temporelle et la possibilité d'agir sur les trajectoires pour éviter les conflits. En effet, si la durée de la fenêtre temporelle est trop grande, on risque de se retrouver avec un nombre important de conflits imbriqués. D'un autre coté, si la durée de la fenêtre temporelle est trop courte, le champ d'action est trop restreint et on ne peut plus agir sur les trajectoires pour éviter les conflits.

L'intervalle de temps $\delta t = 7$ minutes correspond au temps de vol pendant lequel l'avion va effectivement suivre la trajectoire proposée par l'application au pas de la simulation courant. L'horizon de prédiction (et de résolution) étant limité, des « effets d'horizon » néfastes peuvent apparaître. En effet, il est possible de trouver un conflit juste après la fin de la fenêtre temporelle courante. Ce conflit ne pourra pas être traité dans la fenêtre temporelle suivante car les avions seront en conflit dès le début de cette fenêtre, sans latitude suffisante pour les séparer. En effet, le début de la fenêtre fixe les conditions initiales de position des avions pour un sous-problème de résolution de conflits. C'est pourquoi, on ne fait pas glisser la fenêtre de Δt mais uniquement de δt. D'autre part, afin de retrouver les conflits au centre de la fenêtre temporelle et non pas à la fin (ce qui risquerait d'engendrer des situations où on ne peut pas résoudre le conflit dû, encore une fois, à un manque de marge de manœuvre), on ne détecte les conflits que sur une fraction f (nous avons choisi $f = \frac{2}{3}$ dans nos tests) de la durée de la fenêtre temporelle Δt. Plus précisément, à un pas de la simulation t, les segments de trajectoires sont extraits entre l'instant t et l'instant $t+\Delta t$, mais on ne détecte les conflits qu'entre l'instant t et l'instant $t + f \times \Delta t = t + 14$ minutes dans notre mise en œuvre (voir figure 3.7).

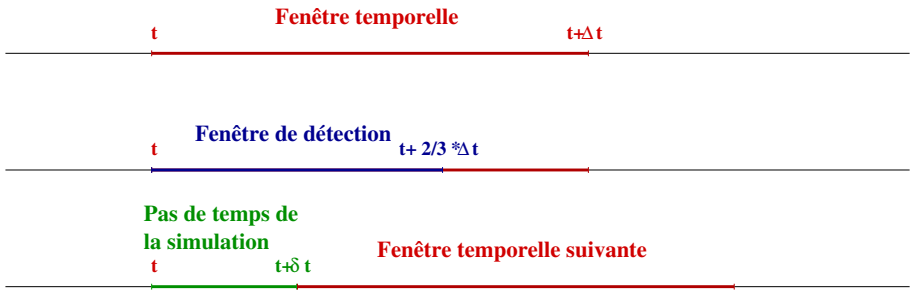


FIGURE 3.7 – Principe des fenêtres glissantes aux pas descrétisation t et $t + \delta t$

Dans les paragraphes suivants, nous détaillons le fonctionnement de notre méthodologie LPA-traffic. Nous décrivons d'abord les données en entrée issues du simulateur CATS.

Nous précisons la base de données utilisée pour stocker ces données. Ensuite, nous détaillons les phases de détection des conflits, de résolution et de réintégration des segments de trajectoires modifiés après la résolution des conflits. Enfin, nous exposons les résultats obtenus par l'APLd lorsqu'il est appliqué sur tous les sous-problèmes de résolution de conflits identifiés pour une journée de trafic sur la France.

Simulateur CATS

Le simulateur CATS (*Complete Air Traffic Simulator*) [2, 15, 72] a été développé en 1992 par le pôle POM (Planification, Optimisation et Modélisation du trafic aérien) de la DSNA/DTI [1], afin de pouvoir disposer d'un outil simple et léger, fournissant une représentation réaliste du trafic aérien sur l'ensemble de la France.

Il s'agit d'un simulateur de trafic en-Route basé sur un modèle d'exécution discret : les positions et les vitesses d'un avion sont calculées à des pas de temps fixes : toutes les 5, 10 ou 15 secondes.

Données en entrée :

Le simulateur utilise une base de plan de vols correspondant à une journée de trafic dans l'espace aérien français. Ces données proviennent du Système de traitement initial du plan de vol (STIP). Les données utilisées pour chaque vol sont :

- le type d'avion,
- l'heure de départ,
- la route prévue,
- le niveau de vol demandé.

Le simulateur dispose également d'une base de données de balises, donnant les coordonnées géographiques des balises en service dans la zone considérée. De plus, les performances des avions sont disponibles sous la forme de tableaux qui décrivent la vitesse sol, la vitesse verticale, la consommation du carburant en fonction de l'altitude, le type de l'avion et le segment de vol (croisière, montée ou descente). Deux principales bases de données des performances avions sont utilisées :

- Les tables de performance avion CAUTRA [2], issues du système français du traitement de données de vol.
- La base de données des performances avion BADA (*Base of Aircraft DAta*) issue du modèle d'énergie total d'EUROCONTROL. Il y a 69 types d'avion différents qui y sont décrits. L'airbus A320 est l'appareil utilisé par défaut.

Configuration du trafic :

Le simulateur permet de modifier la configuration du trafic. En effet, les aéronefs peuvent utiliser différent modes de navigation : Ils peuvent soit suivre des routes directes vers leurs destinations, soit suivre leur route réelle, définie par une succession de balises décrites dans le plan de vol. De plus, un accroissement de la densité de trafic peut être

1. Direction des Services de la Navigation Aérienne : Direction de la Technique et de l'Innovation.

2. L'appellation CAUTRA (Coordinateur AUtomatique du TRafic Aérien) regroupe l'ensemble des systèmes effectuant la gestion des informations plan de vol, leur distribution sur les positions de contrôle, la corrélation avec les pistes radar, l'archivage,...

simulé. Cet accroissement est obtenu en modifiant les heures de décollage des vols de l'échantillon. Toutes les heures sont divisées par un facteur (appelé facteur de compression de temps), ce qui revient à concentrer le trafic sur les premières heures de la journée (le nombre total de vols restant inchangé, la durée effective de la journée se trouve réduite d'autant). Si l'on observe le trafic à l'intérieur d'une fenêtre temporelle choisie dans une période de trafic stable, on obtient un lien direct (linéaire) entre le facteur de compression de temps et le nombre d'avions en vol à un instant donné.

Le trafic simulé n'a pas subi de *pré-régulation*. On appelle pré-régulation le contrôle effectué avant le départ des avions pour éviter les surcharges au décollage des aéroports ou dans les secteurs denses. Dans la pratique, les plans de vol sont déposés plusieurs heures à l'avance afin de permettre cette pré-régulation.

Données en sortie :

Le simulateur offre la possibilité de mémoriser les trajectoires de l'ensemble des avions sous la forme d'une succession de *plots* (points à 4 dimensions) dans des fichiers. Ces fichiers seront utilisés comme entrée pour notre méthodologie LPA-traffic. Dans les fichiers tests que l'on a utilisés, les plots sont échantillonnés toutes les 15 secondes. La configuration du trafic choisie est celle qui suit la route réelle sans accroissement par compression de temps de la densité de trafic. Nous souhaitons tester une situation réaliste du trafic actuel.

Base de données

Les trajectoires des avions pour une journée de trafic sont extraites des fichiers CATS et sont placées dans une base de données MySQL [3]. Nous avons eu recours à la base de données, car nous ne pouvions pas stocker toutes les trajectoires en mémoire (environ 8000 trajectoires). L'autre avantage de la base de données est qu'elle permet de récupérer directement l'ensemble des segments présents dans une zone de l'espace, ce qui nous sera utile par la suite. L'inconvénient est au niveau du temps d'exécution, car le temps d'accès disque est plus grand que le temps d'accès mémoire.

Détection des conflits

Comme le nombre de segments de trajectoire pour une fenêtre temporelle donnée peut être très grand, on a effectué la détection de conflits en deux phases. La première phase consiste à retrouver les conflits macroscopiques, c'est-à-dire retrouver les sous ensembles de segments qui sont dans la même zone géographique et donc potentiellement en conflit. Ensuite, la deuxième phase consiste à identifier, pour chaque sous-ensemble de segments présents dans une même zone géographique, les segments qui sont réellement en conflit.

3. MySQL est un serveur de bases de données relationnelles basé sur le langage SQL (*Structured Query Language*). Ce langage permet de manipuler des données en recherchant, en ajoutant, en modifiant ou en supprimant des données dans les bases de données. MySQL a été développé dans un souci de performances élevées en lecture et n'est pas orienté vers des performances de mises à jour fortement sécurisées.

Détection de conflits macroscopiques :

On commence par associer à chaque segment de trajectoire, entre les instants t et $t + f \times \Delta t$ (on ne détecte les conflits que sur $f = \frac{2}{3}$ de la fenêtre temporelle), le parallélépipède rectangle qui l'englobe (c'est-à-dire le plus petit parallélépipède rectangle qui contient le segment de trajectoire orienté suivant le repère orthogonal $(\vec{x}, \vec{y}, \vec{z})$) augmenté d'une demie norme de séparation dans les plans horizontal et vertical (voir Figure 3.8). Ensuite, on détecte l'intersection des parallélépipèdes. Tous les segments dont les parallélépipèdes se chevauchent sont mis dans le même sous-ensemble de segments appelé *cluster macroscopique de conflit*.

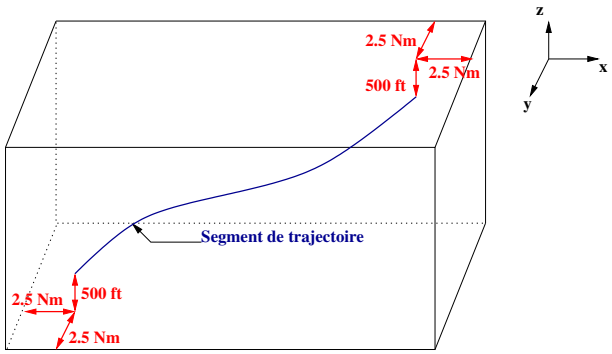
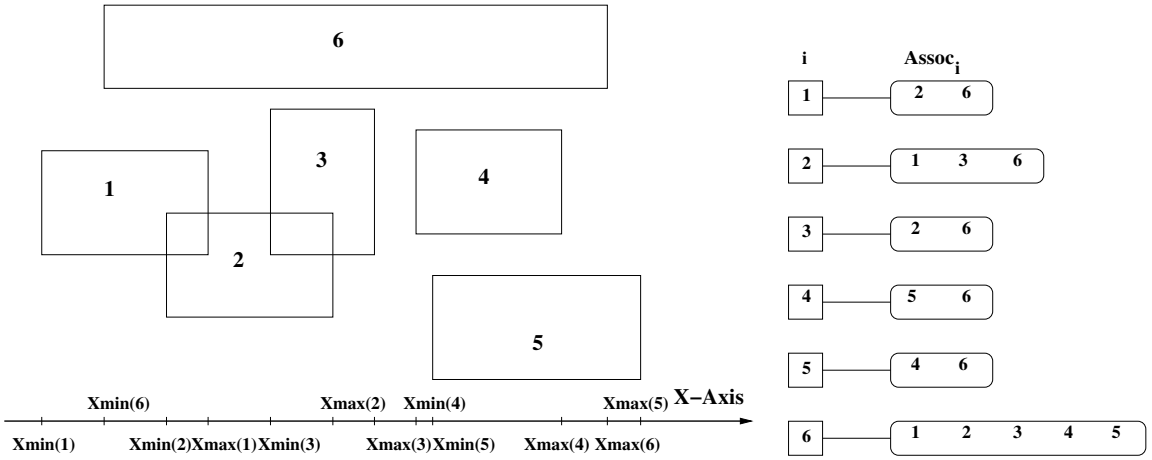


FIGURE 3.8 – parallélépipède rectangle englobant un segment de trajectoire, augmenté d'une demie norme de séparation dans le plan horizontal et vertical

Voici l'algorithme que nous proposons pour détecter les parallélépipèdes qui se chevauchent. Soit n le nombre de parallélépipèdes dont on doit vérifier l'intersection. On détecte d'abord successivement l'intersection de leurs projections sur chacun des axes x, y et z. Pour mieux comprendre le fonctionnement de notre algorithme pour la détection de chevauchement des parallélépipèdes, nous l'expliquons sur l'exemple de la figure 3.9. Dans cet exemple, la détection est faite pour $n = 6$ rectangles (en 2D), mais nous appliquons le même principe en 3D.

On appelle cette procédure de détection des intersections sur l'axe x : DetectX. Elle se déroule comme suit. On commence par projeter les n parallélépipèdes sur l'axe x. On se retrouve alors avec n intervalles dont on doit vérifier l'intersection. Afin d'éviter la combinatoire de la comparaison des intervalles deux à deux (complexité $O(n^2)$), on trie les bornes des intervalles par ordre croissant. Ensuite, on associe à chaque intervalle i un ensemble $Assoc_i$ qui contiendra tous les intervalles dont au moins l'une des bornes se trouve entre $x_{min}(i)$ et $x_{max}(i)$, où $x_{min}(i)$ et $x_{max}(i)$ sont respectivement la borne inférieure et la borne supérieure de l'intervalle i. Réciproquement, l'intervalle i sera associé à chacun des intervalles j de $Assoc_i$ ($j \in Assoc_i \Rightarrow i \in Assoc_j$).

Pour l'exemple bidimensionnel avec $n = 6$ rectangles, la figure 3.9(a) représente la projection des rectangles sur l'axe x et le résultat de l'application de la procédure DetectX est représenté par la figure 3.9(b). Le pseudo-code de la procédure DetectX se trouve à



(a) Projection des rectangles sur l'axe x.

(b) La liste des ensembles $Assoc_i$ résultants de DetectX.

FIGURE 3.9 – Détection de l'intersection de rectangles.

l'annexe B.1.

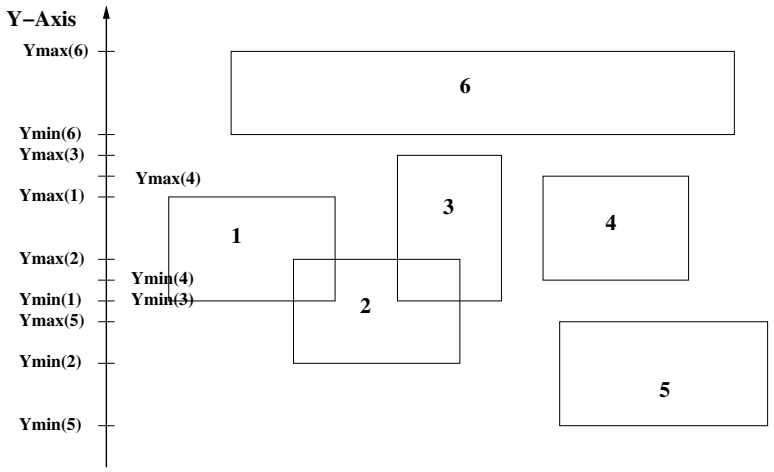
Ensuite, on détecte les intersections des parallélépipèdes sur l'axe y (procédure DetectY). Comme pour DetectX, on commence par projeter les n parallélépipèdes sur l'axe y. Ensuite, pour détecter les intersections, on procède à partir de la liste des ensembles $Assoc_i$ qui résulte de DetectX, afin de vérifier qu'il y a aussi intersection par rapport à l'axe y. Les ensembles d'associations étant beaucoup plus petits que l'ensemble initiale des parallélépipèdes, on se contente ici de comparer les intervalles deux à deux. Ainsi, si deux intervalles i et j ne se croisent pas en y, on enlève i de $Assoc_j$ et on enlève j de $Assoc_i$

La Figure 3.10(a) représente la projection des rectangles sur l'axe y pour l'exemple ci-dessus. La figure 3.10(b) présente les données en entrée pour la procédure DetectY ainsi que l'élimination des associations pour les intervalles qui ne se croisent pas sur l'axe y.

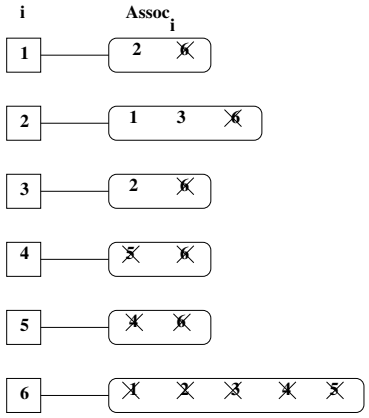
Après l'application de la procédure DetectY, chaque ensemble $Assoc_i$ ainsi mise à jour contient les rectangles qui intersectent le rectangle i. La figure 3.10(c) présente ce résultat pour notre exemple. Le pseudo-code de la procédure DetectY se trouve à l'annexe B.2.

Pour le cas général tridimensionnel, une troisième étape est requise. Elle consiste à détecter les intersections des parallélépipèdes sur l'axe z. La procédure DetectZ est identique à DetectY, si ce n'est qu'on projette cette fois les parallélépipèdes sur l'axe z et que l'on part de la liste des associations qui résulte de DetectY au lieu de celle qui résulte de DetectX.

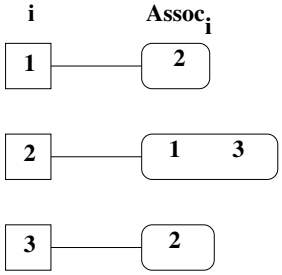
Une dernière procédure appelée Fusion sert à former les véritables clusters de conflits que nous traiterons avec l'APLd comme autant de sous-problèmes. La procédure Fusion prend en entrée un parallélépipède p et considère donnée la liste des associations issue de DetectZ. Fusion(p) calcule récursivement la fermeture transitive des parallélépipèdes en conflit avec le parallélépipède p. Pour cela, la procédure Fusion parcourt l'ensemble $Assoc_p$ et, pour chaque parallélépipède k de l'ensemble $Assoc_p$: p est d'abord enlevé de



(a) Projection des rectangles sur l'axe y



(b) La liste des ensembles $Assoc_i$ résultants de DetectX.



(c) La liste des ensembles $Assoc_i$ résultants de DetectY.

FIGURE 3.10 – Détection de l'intersection de rectangles.

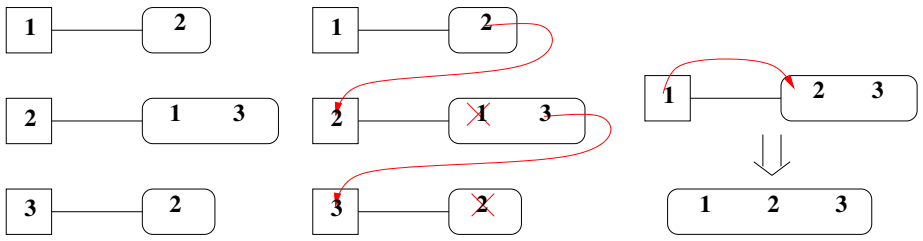
$Assoc_k$ (pour éviter une récursivité infinie), on récupère l'ensemble résultant de Fusion(k) et on l'ajoute dans le cluster résultat. Enfin, une fois l'ensemble $Assoc_p$ parcouru, le parallélépipède p est ajouté au cluster résultat.

Pour l'exemple avec $n = 6$ rectangles, la figure 3.11(a) représente les données en entrée pour la procédure Fusion qui est appliquée au rectangle 1. Sur la figure 3.11(b), qui représente le déroulement de la procédure Fusion(1) :

- $Assoc_1$ est parcouru, on trouve 2 ;
- on élimine 1 de $Assoc_2$ et on applique Fusion(2) ;
  - $Assoc_2$ est parcouru, on trouve 3 ;
  - on élimine 2 de $Assoc_3$ et on applique Fusion(3) ;
    - $Assoc_3$ est vide on rajoute 3 au résultat de Fusion(3) ;
  - on rajoute 2 au résultat de Fusion(2).
- on rajoute 1 au résultat de Fusion(1).

La dernière étape de la procédure, qui consiste à ajouter 1 au cluster résultat, est représentée par la figure 3.11(c).

Le pseudo-code de la procédure Fusion se trouve à l'annexe B.3.



(a) La liste des ensembles $Assoc_i$ résultants de DetectZ.

(b) Récursivité.

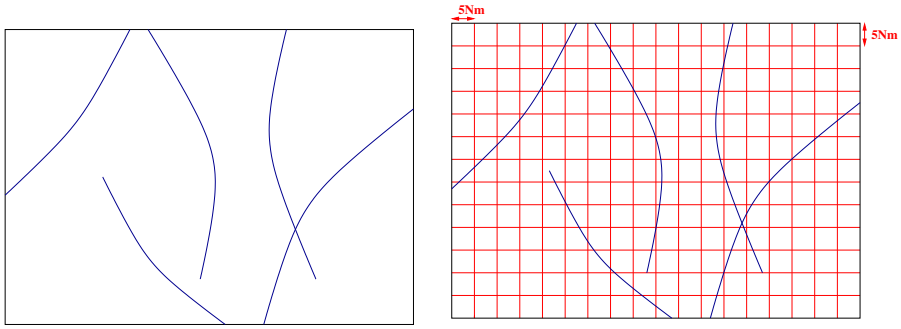
(c) Résultat

FIGURE 3.11 – Déroulement de la procédure Fusion appliqué au rectangle 1.

Une fois que la procédure Fusion est appliquée à chaque parallélépipède, on se retrouve avec une liste de *clusters macroscopique de conflits*. Il faut maintenant détecter pour chacun d'eux les conflits réels.

Détection des conflits réels :

Pour détecter les conflits réels au sein d'un cluster macroscopique de conflits, on doit, ici aussi, éviter la combinatoire de la comparaison deux à deux des plots (points 4D) des différents segments qui appartiennent au cluster macroscopique. Pour cela, on cherche le rectangle qui englobe les projections dans le plan horizontal de ces segments (c'est-à-dire le plus petit rectangle orienté suivant le repère orthogonal $(\vec{x}, \vec{y})$ qui contient les projections des segments) (voir figure 3.12(a)). On place ensuite une grille sur le rectangle englobant dont la dimension de la case élémentaires est $(5Nm \times 5Nm)$. Les projections des segments sont placées sur cette grille (voir figure 3.12(b)).



(a) (b)

FIGURE 3.12 – Rectangle englobant des segments (a) et grille de détection de conflits (b).

Les cases qui sont parcourues par les segments sont mémorisées avec les dates d'entrée t_{in} et de sortie t_{out} des segments dans ces cases ainsi que les altitudes d'entrée z_{in} et de sortie z_{out}. Pour qu'il y ait un *conflit potentiel* entre deux segments i et j, il faut qu'ils passent par une même case(notée $c_{i,j}$) ou par deux cases voisines (notées respectivement c_i et c_j) dans un même intervalle de temps et avec une différence d'altitude inférieure à la

norme de séparation verticale (voir figure 3.13(a)). Dans ce cas, enfin, on vérifie pour les plots des segments i et j qui se trouvent dans ces cases adjacentes c_i et c_j (ou la même case $c_{i,j}$) et qui ont la même date associée, si la distance de leurs projections dans le plan horizontal est inférieure à la norme horizontale de séparation (voir figure 3.13(b)). Si c'est le cas pour au moins un couple de plot, les segments i et j sont considérés en conflit et sont placés dans le même *cluster élémentaire de conflits*. Une fois tous les conflits détectés, les clusters élémentaire de conflits qui ont au moins un segment en commun sont fusionnés pour former les *cluster de conflits* (voir figure 3.13(c)).

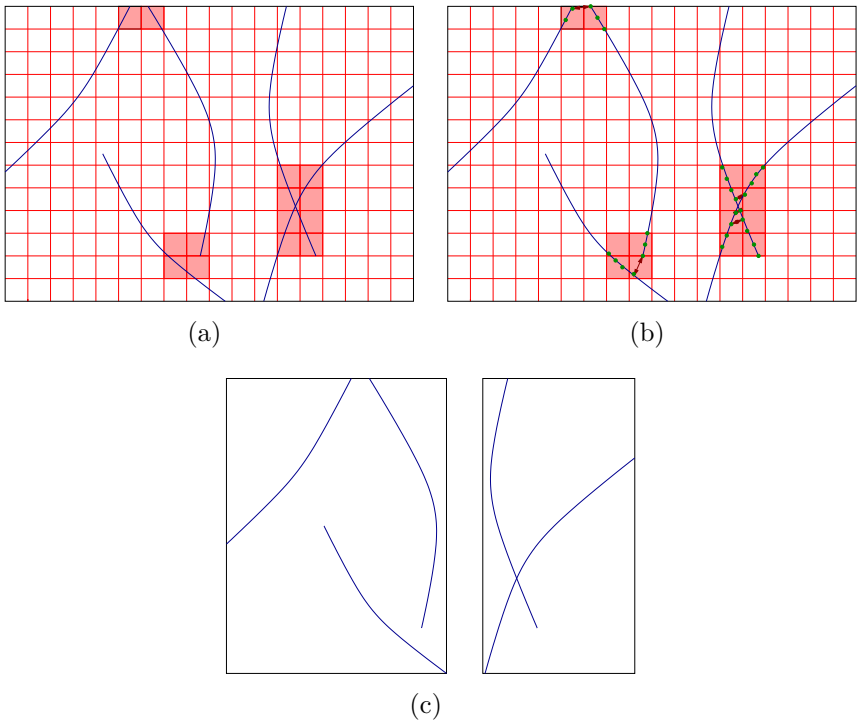


FIGURE 3.13 – (a) En rose : cases parcourues par des segments potentiellement en conflit. (b) Comparaison des plots (en temps et en altitude) des segments potentiellement en conflit. (c) À gauche : un cluster de conflit obtenu après fusion de deux clusters élémentaires de conflits. À droite : un cluster de conflits issu d'un cluster élémentaire de conflits.

Les clusters de conflits ainsi construits constituent donc autant de sous-problème à résoudre par l'APLd.

Résolution des conflits

Pour résoudre un cluster de conflit avec l'APLd, on est amené à modifier tous les segments des trajectoires concernées extraits entre l'instant t et $t + \Delta t$ (la résolution de conflits, contrairement à la détection, a lieu sur toute la fenêtre temporelle). Pour chaque segment a appartenant au cluster, l'APLd doit retrouver un nouveau segment qui relie le point de départ s_a au point destination d_a du segment a, tout en évitant d'entrer en conflit

avec les autres segments du cluster. Les nouveaux segments proposés par l'APL peuvent induire des conflits *secondaires* avec des segments qui se situent dans le voisinage mais pas dans le cluster de conflits. Afin d'éviter ce problème, on limite la zone où l'on permet la résolution. Ainsi, on autorise les nouveaux segments de trajectoire (proposées par le solveur) uniquement à l'intérieur d'un *parallélépipède de résolution*. Le parallélépipède de résolution est défini comme le plus petit parallélépipède rectangle orienté suivant le repère orthogonal $(\vec{x}, \vec{y}, \vec{z})$ qui contient les segments appartenant au cluster de conflits. De plus, afin de prendre en compte les segments de trajectoire non initialement en conflit mais qui risquent d'induire des conflits secondaires, on les considère dans un premier temps comme des contraintes du problème. On appellera ces segments : *segments contraintes*. Pour cela, on considère un *parallélépipède de contraintes* qui est défini comme étant égal au parallélépipède de résolution augmenté d'une norme de séparation dans les plans horizontal et vertical (voir figure 3.14). Les segments des trajectoires contraintes sont les segments contenus dans le parallélépipède de contraintes, n'appartenant pas au cluster de conflits. Les contraintes supplémentaires qu'ils induisent seront modélisés par leurs tubes de protection 4D dans lesquels l'indice de réfraction prendra la valeur d'indice I_{max}. Ces tubes 4D sont en effet ainsi des zones interdites dans le parallélépipède de résolution de façon à ne pas modifier les segments contraintes. Cependant, si par la suite il s'avère qu'aucune solution n'est trouvée au problème de résolution de conflit, on devra revenir à ce stade, relaxer ces contraintes pour permettre de modifier *aussi* ces trajectoires contraintes.

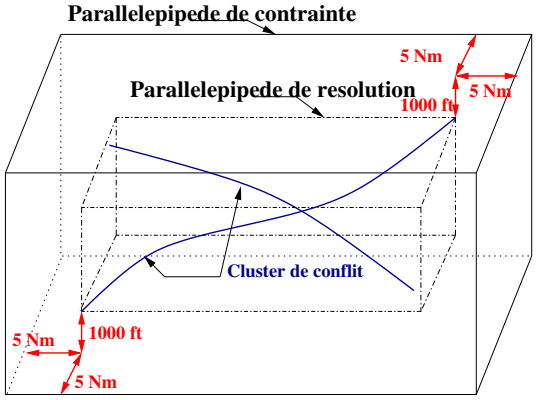


FIGURE 3.14 – Parallélépipède de résolution et parallélépipède de contraintes

Pour résoudre les conflits d'un cluster de conflits donné, l'APLd est appliqué de façon séquentielle aux segments de trajectoire appartenant au cluster de conflits. Comme on l'a mentionné précédemment, l'ordre dans lequel on traite ces segments a une influence sur la qualité de la solution et sur l'existence parfois même d'une solution. En effet, les nouveaux segments de trajectoire générés, dans un certain ordre, peuvent bloquer l'espace de résolution pour les segments de trajectoire suivants. Dans un premier temps, nous avons choisi un ordre aléatoire de résolution. Cet ordre est placé dans une *pile de résolution*. S'il ne produit pas de solution sans conflit (pour l'un des segments de trajectoire, on ne trouve pas de segment de rechange qui relie son point de départ s à son point destination

d sans violer les zones interdites dans le parallélépipède de résolution), nous modifions l'ordre en commençant par la trajectoire qui n'a pas trouvé de solution avec l'ordre de résolution précédent (on la place au sommet de la pile de résolution). Si à nouveau, on ne trouve pas de solution, on place le nouveau segment sans solution au sommet de la pile, et ainsi de suite jusqu'à ce que l'on trouve une solution ou que les deux premiers segments de trajectoire de la pile se bloquent dans les deux sens de résolution, c'est-à-dire si l'ordre de résolution segment 1 puis segment 2 ne donne pas de solution et que l'ordre de résolution segment 2 puis segment 1 ne donne pas non plus de solution. Un exemple de manipulation de l'ordre de résolution est illustré par la figure 3.15. Dans le cas de blocage de la résolution, on tente de relâcher les contraintes. Les segments contraintes, inclus dans le parallélépipède de résolution ne sont alors plus considérés comme des contraintes du problème, mais sont introduits dans le cluster de conflits et sont traités comme les segments initialement en conflit. On considère de nouveau un ordre de résolution aléatoire pour le nouveau cluster de conflits mais on commence par traiter les segments initialement en conflit. Ensuite, en cas de non résolution, on remet toujours le segment qui n'a pas trouvé de solution en haut de la pile.

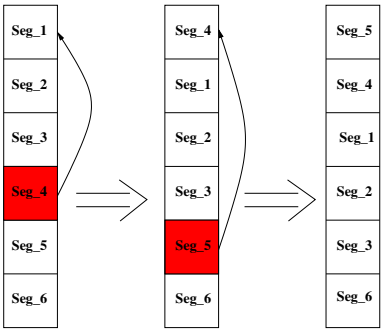


FIGURE 3.15 – Exemple de manipulation de l'ordre de la pile de résolution. Les segments en rouge sont les segments qui n'ont pas trouvé de solution dans l'ordre de résolution courant.

Comme on résout des conflits en-Route, l'APLd a besoin d'une fonction qui lui permet d'obtenir à chaque instant l'altitude de chacun des avions, afin que l'algorithme puisse ajuster la propagation dans la dimension verticale sur le profil vertical initial de l'avion. Pour cela, on récupère à partir de la trajectoire initiale de l'avion, les altitudes correspondant à l'intervalle de temps qu'on est en train de traiter. De plus, pour le lancement des rayons lumineux, l'APLd a recours à un profil de vitesse qui permet d'obtenir la norme de la vitesse de l'avion dans le plan horizontal à chaque instant. De même que pour le profil d'altitude, on le récupère à partir de la trajectoire initiale de l'avion.

Maintenant, qu'on a résolu les clusters de conflit, on doit réintégrer les nouveaux segments produits par l'APL dans la base de données.

Réintégration des segments de trajectoires

Afin de réintégrer un nouveau segment s_{new} de trajectoire produit par l'APLd dans la base de données, on commence par supprimer l'ancien segment s_{old} (avant résolution) qui lui correspond dans la base. Si s_{new} et s_{old} ont la même longueur, il n'y a pas de traitement à faire pour le reste de la trajectoire t_{suite} (la partie de la trajectoire qui est en aval de s_{old}). Sinon, si s_{new} est plus long ou plus court que s_{old}, les plots de t_{suite} doivent être recalculés. Une manière simple de recalculer ces plots est de les redater en les échantillonnant toutes les 15 secondes à partir du temps final de s_{new} (l'instant associé au point destination dans s_{new}), et d'associer à chaque plot l'altitude et la vitesse qui correspondent à sa nouvelle date dans les profils d'altitude et de vitesse de la trajectoire. Malheureusement, avec cette manière simple de recalcul, la vitesse associée à un plot donné, étant décalée, elle ne correspond plus à la vitesse nécessaire pour relier ce plot au plot suivant. Pour éviter ce problème, on recalcule les dates réelles de passage aux plots de t_{suite}, en se basant sur le profil de vitesse de la trajectoire. Ensuite, on calcule les véritables plots échantillonnés toutes les 15 secondes sur t_{suite} à partir de l'avant-dernier plot de s_{new} (voir l'exemple de la figure 3.16). Les ancien plots de t_{suite} sont supprimés de la base de données et sont remplacés par les nouveaux.

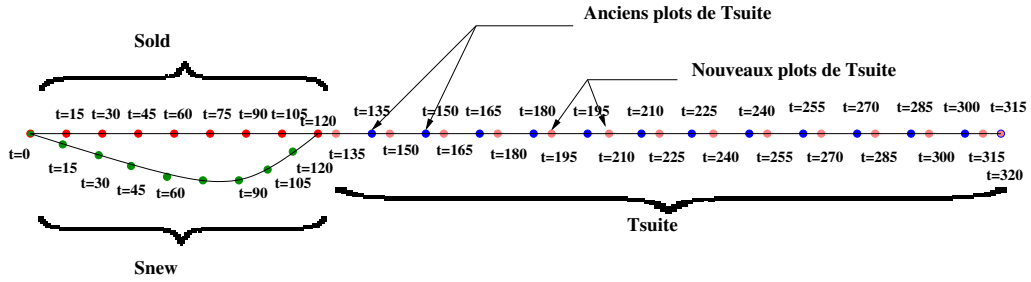


FIGURE 3.16 – Remplacement du segment s_{old} par le segment optimisé s_{new} et recalcul des nouveaux plots de t_{suite}.

On vient de détailler dans les paragraphes précédents : la récupération des trajectoires des avions pour une journée de trafic sur la France, la détection des conflits, leurs résolutions et finalement la réintégration des trajectoires ainsi optimisées, dans la base de données des trajectoires. Dans le prochain paragraphe, on exposera les résultats trouvés pour un problème réel qui correspond à une journée de trafic.

Résultats

Dans ce paragraphe, on présente les résultats que l'on a retrouvés pour une journée de trafic sur la France (12 août 2008) avec plus de 8000 avions. Pour la résolution des conflits, l'APLd est appliqué avec, outre les paramètres qui ont été fixés pour toute la section 3.2, un pas de discrétisation temporel $dt = 15$ secondes qui est le pas d'échantillonnage de nos données en entrée.

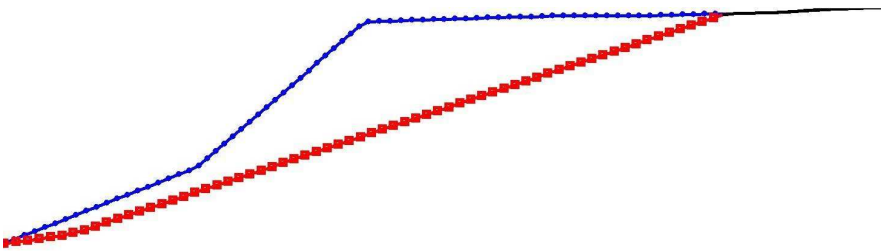


FIGURE 3.17 – Exemple de résolution de conflit qui induit une distance plus courte. La courbe simple représente la partie future de la trajectoire, t_{suite}. La courbe avec des cercles représente la trajectoire initiale dans la fenêtre temporelle courante, s_{old}, et la courbe avec des carrés représente la trajectoire produite par l'APLd, s_{new}.

Les trajectoires initiales (avant la résolution des conflits) induisent un nombre total de clusters de conflits, comprenant de 2 jusqu'à 10 avions en conflits réels, égal à 3344. L'algorithme résout presque tous les conflits. Il n'y a que 28 cas pour lesquels l'APLd n'a pas réussi à trouver des trajectoires alternatives sans conflit mais ces situations, très particulière, correspondent à certains avions qui étaient déjà en conflit au début de la simulation, dès leurs points de départ ; il s'agit donc d'instance qui ne peuvent pas être résolues. Au total, seules 1501 trajectoires ont été modifiées par l'algorithme pour atteindre une telle situation sans conflit.

Il est intéressant de noter que dans de nombreux cas, les nouvelles trajectoires calculées, en plus d'être exemptes de conflit, sont plus courtes que leurs trajectoires initiales associées (celles qui suivent les balises). Ceci s'explique ainsi : l'APLd recherche le plus court chemin et propose un chemin direct dès que c'est possible. En moyenne, la longueur des trajectoires calculées est raccourcie de 4.41 Nm par avion, avec un minimum de 51,9 Nm. D'autre part, certaines trajectoires subissent une extension de leur longueur avec un maximum de 9,86 Nm. Les quantiles associés à la modification de la longueur des trajectoires sont présentés dans la table 3.2.

Quantile	Distance en Nm
$\frac{1}{4}$ quantile	-4.94
$\frac{1}{2}$ quantile	-1.65
$\frac{3}{4}$ quantile	-0.07

TABLE 3.2 – Modification de la longueur des trajectoires après résolution par l'APL.

Le temps de calcul global est d'environ 17 heures de temps CPU.

Quelques exemples spécifiques de résolutions de conflit sont affichés dans les figures 3.17, 3.18 et 3.19. Les parties passées et futures des trajectoires sont représentées par des courbes simples. La fenêtre temporelle considérée est représentée par des courbes avec des carrés ou des cercles. La courbe avec des cercles représente la trajectoire initiale (avec des conflits),

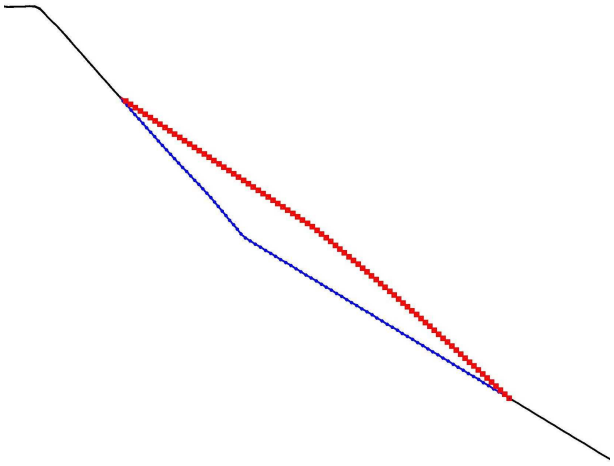


FIGURE 3.18 – Exemple de résolution de conflit qui n'induit pas de changement dans la longueur de la trajectoire.

s_{old}, et la courbe avec des carrés est la solution produite par l'APLd (sans conflit), s_{new}.

Toute trajectoire solution produite par l'APLd respecte les contraintes de vitesse de l'avion (en suivant le profil de vitesse initial de l'avion). Elle est lisse, comme requis pour les avions de ligne (grâce à la contrainte sur le changement de cap, lorsque les rayons sont lancés à partir de la position actuelle). Les trajectoires synthétisées sont ainsi parfaitement adaptées pour être suivies par le FMS.

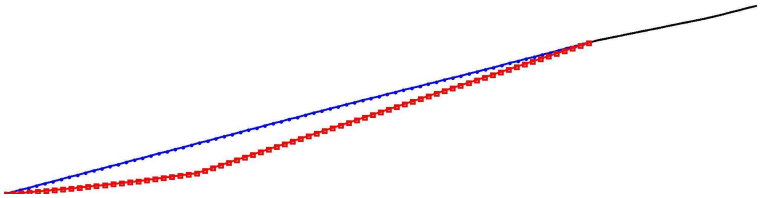


FIGURE 3.19 – Exemple de résolution de conflit qui induit une distance plus longue.

Certaines perspectives pour des travaux futures laissent déjà entrevoir des améliorations prometteuses. D'abord, nous nous attendons à réduire sensiblement le temps de calcul en réécrivant certaines parties de l'application en langage C. Dans une deuxième étape, pour une fenêtre de temps donnée, les clusters de conflits (qui sont, rappelons-le, indépendants) pourront être résolus sur plusieurs processeurs en parallèle.

Conclusion

Dans cette section, nous avons obtenu des résultats satisfaisants pour une journée de trafic réel sur la France : résolution des conflits à 100% (pour ce qui est des conflits pour lesquels une solution existait), satisfaction des contraintes opérationnelles (sur la vitesse des avions et la courbure des trajectoires). Par contre, nous n'avons pas pris en compte les incertitudes sur les positions des avions. En effet, nous avons supposé que nous connaissions précisément, à chaque instant, la position 4D de l'avion. Cette hypothèse n'est pas réaliste dans le contexte opérationnel actuel (limites des capacités actuelles des outils de navigations), comme nous allons le voir dans le chapitre suivant. Nous nous proposons dans le chapitre suivant de tester une journée de trafic réel, en prenant en compte cette incertitude sur la position des avions.

Chapitre 4

Gestion des incertitudes

Ce chapitre aborde une difficulté supplémentaire : le fait qu'en pratique on ne connaît pas la position exacte des avions à tout moment dans le futur, contrairement à ce que nous avons supposé jusqu'ici.

Nous présentons d'abord cette problématique dans la prochaine section. Ensuite, nous décrivons en section 4.2, les différents modes possibles de gestion de l'incertitude par les FMS futur. Nous prenons, par la suite, en compte les hypothèses d'incertitude de l'un de ces modes des FMS futur pour proposer une adaptation de notre méthodologie LPA-traffic, afin de traiter le problème de résolution de conflit avec incertitude en section 4.3. Enfin, nous présentons des résultats numériques encourageant pour ce problème difficile en section 4.4.

4.1 Problématique

Dans la section 3.2.2 du chapitre précédent, dans notre méthodologie LPA-traffic, nous avons extrait les segments de trajectoires présents entre l'instant courant t et l'instant $t + \Delta t$, afin de prédire le trafic sur un horizon temporel Δt. Ceci implique que chaque avion suive parfaitement sa trajectoire prévue sur cet horizon temporel. Cette hypothèse n'est pas réaliste. En effet, un prédicteur de trajectoire (figure 4.1) va, à partir d'une situation initiale, d'une liste d'objectifs en vitesse, en cap et en altitude, à partir du vent, de la température et d'un modèle de performance avion, calculer la position de l'avion au pas de temps suivant. Dans la réalité, de nombreuses sources d'incertitudes peuvent faire dévier l'avion de cette position prévue. L'incertitude que nous traitons dans ce chapitre, est donc la différence entre la position réelle de l'avion et sa position prévue, sur l'horizon temporel de prédiction Δt.

L'incertitude dépend de la précision des données météorologiques notamment de la vitesse réelle du vent et de la température réelle au niveau de vol concerné. Ces deux paramètres ont une influence directe sur la vitesse sol de l'avion. L'incertitude dépend aussi de la précision de l'information sur la masse de l'avion, qui influe sur les performances en montée, et évidemment, de la précision des conditions initiales de position et de vitesse.

Le modèle d'incertitude que nous utilisons dans cette étude est le suivant : on considère

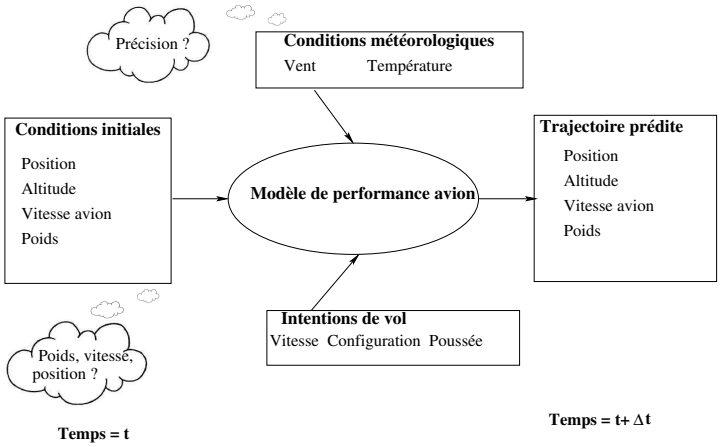


FIGURE 4.1 – Prédiction de trajectoire et sources d'incertitude.

uniquement l'incertitude sur l'axe longitudinal, c'est-à-dire le long de la trajectoire de l'avion. On néglige ainsi l'incertitude latérale et l'incertitude dans le plan vertical, sachant que le FMS actuel est capable de maintenir l'avion sur sa trajectoire prévue avec une grande précision. Le modèle d'incertitude choisi concernera donc le temps de passage prévu au différents points de la trajectoire. Pour cela, on considère que pour 95% du temps, on a une erreur maximale sur la valeur du vent de ± 15 Kt. En considérant le pire cas, c'est-à-dire le cas où la vitesse des avions est la plus faible (autour de 250 Kt), cette erreur maximale sur la vitesse du vent correspond à une *erreur temporelle maximale* autour du temps prévu de passage à une position donnée de la trajectoire. L'erreur temporelle maximale ainsi définie est donc de $\pm 6\%$ du temps total écoulé à partir début de la prédiction (voir figure 4.2).

La prise en compte de ce modèle d'incertitude change le problème de résolution de conflits par rapport à celui exprimé dans 3.2, notamment en ce qui concerne la zone de protection d'un avion (zone interdite pour les autres avions). En effet, le problème s'exprime à présent ainsi : soit $\gamma(t)$ la trajectoire d'un avion, avec t le temps de parcours de la trajectoire. Pour un point $P = \gamma(t_{expected})$, avec $t_{expected}$ le temps de passage prévu en ce point, on récupère à partir de la courbe de la figure 4.2, la différence avec le temps de passage au plus tard δt_{late} et la différence avec le temps de passage au plus tôt δt_{early}. La zone de protection autour de P ne sera plus le cylindre représentée par la figure 3.3, mais sera l'union des cylindres de protection de tous les points $\gamma(t)$, avec $t \in [t_{expected} - \delta t_{late}, t_{expected} + \delta t_{early}]$. Cette nouvelle zone de protection est représenté dans la plan horizontal par la figure 4.3. Dans le plan vertical, la zone de protection n'est pas modifiée puisqu'on ne considère pas d'incertitude dans ce plan.

Avec ce nouveau modèle, très conservateur, on risque donc de détecter beaucoup plus de conflits que dans la section 3.2.2. On risque même de ne plus trouver de solutions à certains conflits, puisque le problème devient beaucoup plus contraint. En effet, la figure 4.4 illustre par un simple exemple de l'évolution du problème à traiter. Deux avions, dont les routes se croisent, sont représentés sur la figure 4.4(a) par leur zones de protection

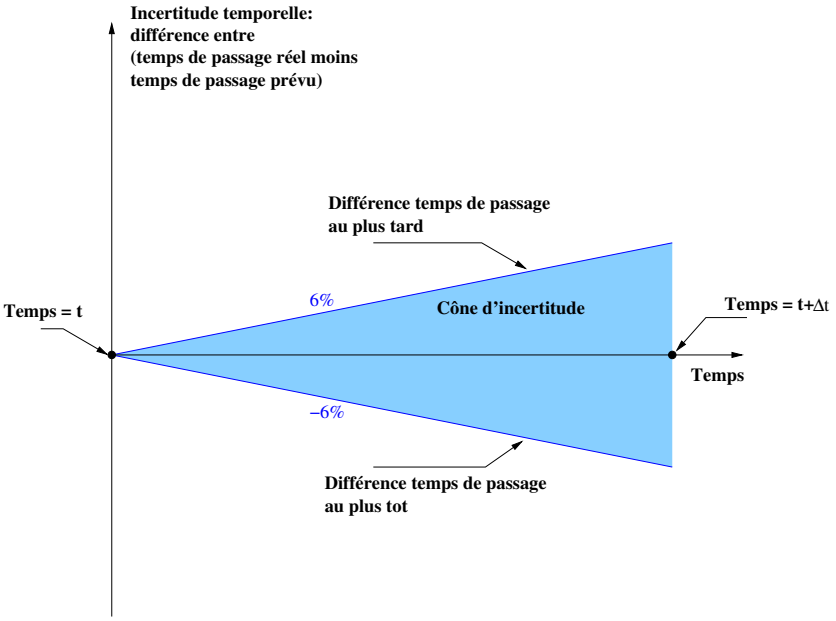


FIGURE 4.2 – modèle d'incertitude.

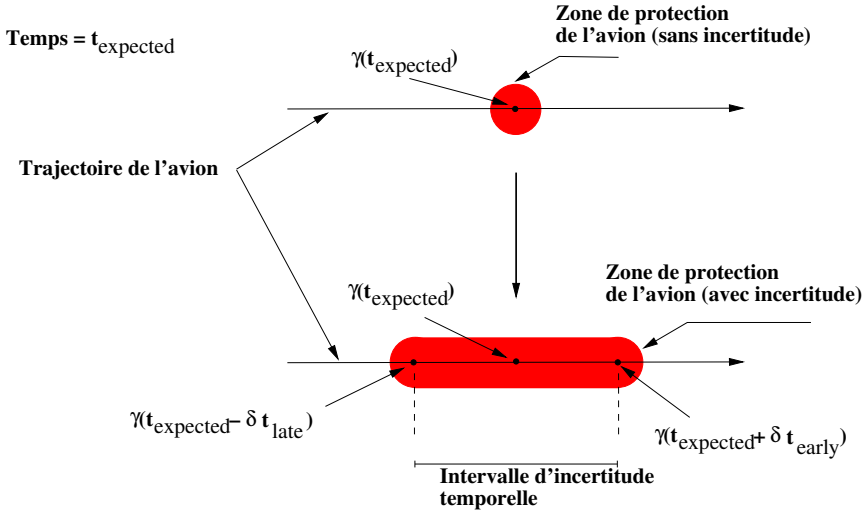
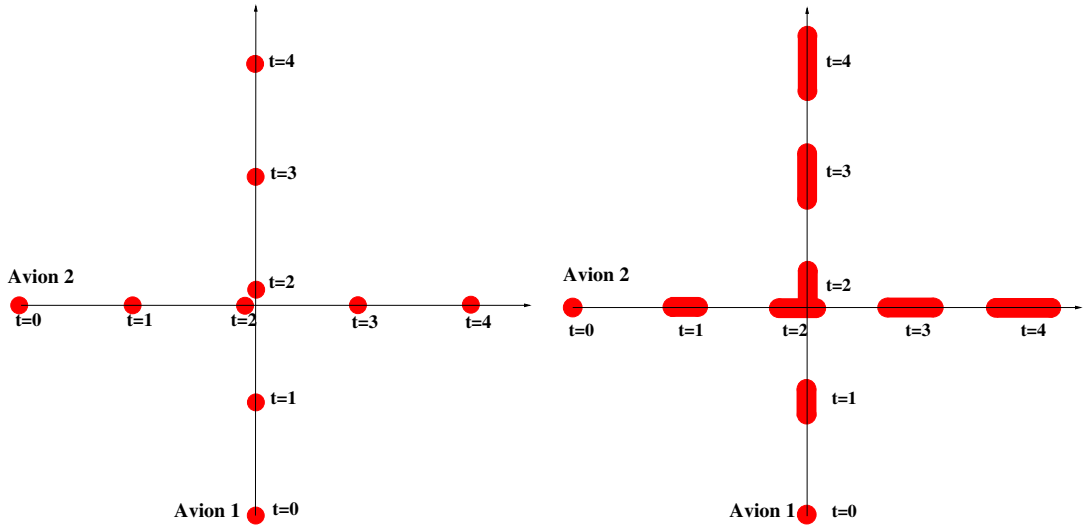


FIGURE 4.3 – Zone de protection d'un avion dans le plan horizontal, sans et avec prise en compte de l'incertitude.

sans incertitude à différents pas de temps. Dans la figure 4.4(b), ils sont représentés par leurs zones de protection avec incertitude à ces mêmes pas de temps. Dans ce dernier cas, un conflit est detecté, alors que dans le premier cas, il ne l'est pas.

Afin de réduire l'incertitude et donc afin de réduire la difficulté du problème, nous décrivons les différentes pistes pour le développement des FMS futurs dans la section suivante.



(a) Sans incertitude (b) Avec incertitude

FIGURE 4.4 – Exemple (2D) de l'influence de l'incertitude sur la détection de conflit.

4.2 FMS du futur

Afin de réduire l'incertitude décrite précédemment, différents modes de contrôle de l'avion sont envisagés pour le FMS futur. Ces modes sont détaillés dans les paragraphes suivants.

Mode contrôle en vitesse

Ce mode ne vise pas à proprement parler à réduire l'incertitude, mais plutôt à déplacer ce qui est communément appelé *la boule d'incertitude* autour de la position de l'avion (zone de protection d'un avion avec incertitude), pour ne pas entrer en conflit avec la boule d'incertitude d'un autre avion. Le levier d'action pour ce faire est un contrôle en vitesse. Plus précisément, on se permettra d'accélérer ou de décélérer l'avion dans la limite, considérée comme acceptable en pratique, de [-3%, +6%] de la vitesse initiale de l'avion et ce, dans la fenêtre temporelle que nous envisageons (Δt = 21 min). Ce mode de fonctionnement est déjà disponible avec le FMS actuel, puisque le FMS contrôle les différentes commandes de l'avion (poussée, roulis, tangage) et, par conséquent, il contrôle la vitesse de l'avion. Ce mode est le plus simple du point de vue bord. Il est cependant le moins intéressant du point de vue sol puisqu'il ne réduit pas l'incertitude comme telle, en se contentant de la déplacer de façon à minimiser son impact en termes de conflits.

Mode RTA final

Comme décrit dans l'introduction, ce mode est déjà disponible sur le FMS actuel. Il s'agit d'imposer à l'avion, d'arriver à un point donné de la trajectoire, à un instant donné appelé RTA (*Required Time of Arrival*) avec une tolérance de ±10s. Une courbe

enveloppe approximative de l'incertitude temporelle (qu'on appellera simplement *courbe d'incertitude* dans la suite) avec un point RTA en boucle fermée est donnée par la figure 4.10. Comme avant, cette courbe est valable à 95% du temps. Elle évolue d'abord comme la courbe en boucle ouverte (figure 4.2) avec une pente de ±6% du temps total écoulé à partir début de la prédiction. Puis, en s'approchant de l'instant RTA, le FMS actuel est programmé pour commencer à agir sur la vitesse de l'avion à partir de l'instant $\frac{2}{3}$ (RTA-t), t étant l'instant de début de la prédiction, pour compenser l'erreur en temps de façon à ce que l'avion soit à la position initialement prévue à l'instant RTA.

Ce mode est actuellement utilisé en approche finale. Il s'agit d'une amélioration par rapport au mode précédent, puisque l'incertitude est réduite au voisinage du point RTA. Cependant, comme ce point est de fait imposé en fin de trajectoire, l'incertitude ne commence à diminuer que sur cette partie de la trajectoire. Ce point RTA final n'a donc pas d'influence sur la détection et la résolution des conflits en début de trajectoire, puisque la courbe d'incertitude sur cette partie de la trajectoire est identique à celle en boucle ouverte.

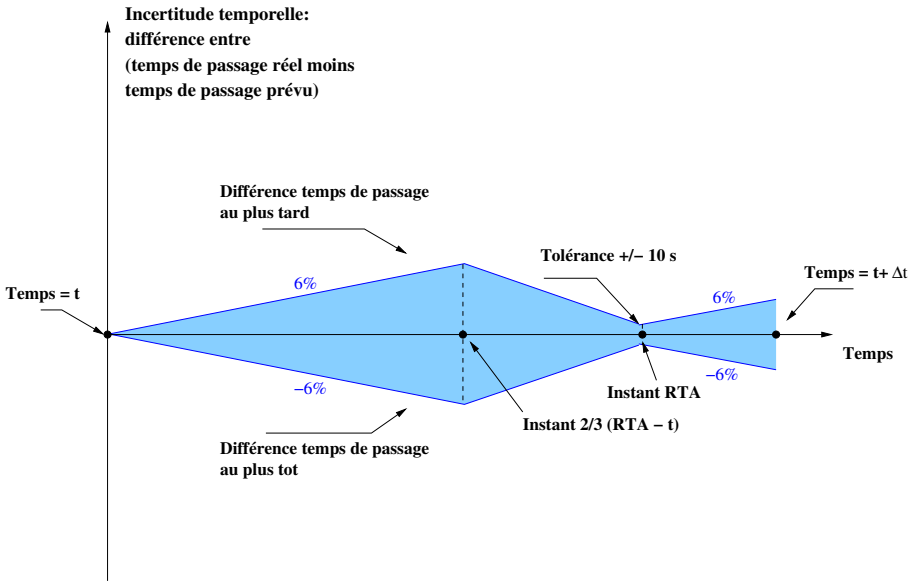


FIGURE 4.5 – Modèle d'incertitude avec un point RTA.

Mode multi-RTA

Contrairement aux deux modes précédents, ce mode n'est pas encore disponible sur le FMS actuel. Il s'agit d'une évolution du mode RTA final, puisqu'au lieu d'imposer un seul point RTA sur la trajectoire de l'avion en phase d'approche, avec le mode multi-RTA, on va pouvoir imposer un nombre n de points RTA tout au long de la trajectoire de l'avion. La courbe d'incertitude reste la même que dans le mode précédent, avec l'incertitude qui diminue jusqu'à ±10 secondes au voisinage de chaque point RTA et repart avec une pente

de ±6% à partir de ce point. À l'inverse du mode précédent, le mode multi-RTA permet d'agir sur la détection et la résolution de conflits tout au long de la trajectoire. Par contre, ce mode est plus exigeant du point de vue bord, puisqu'il implique un asservissement plus régulier de la vitesse de l'avion, que dans le cas précédent.

C'est l'application du mode multi-RTA que nous traiterons dans ce chapitre. Le but de cette étude est de déterminer le nombre n de points RTA à imposer sur la trajectoire des avions et leurs emplacements (spatio-temporel), dans le but de retrouver des résultats de résolution de conflits aussi satisfaisants que ceux du chapitre précédent.

Mode full 4D

Ce mode est le plus exigeant du point de vue bord et le plus intéressant du point de vue sol. Ce mode exige d'une part, comme avant, que l'avion suive parfaitement sa trajectoire, et d'autre part, le mode full 4D requiert de plus que l'avion passe par chaque point de sa trajectoire *à l'instant prévu*, Cela implique qu'il n'existe aucune incertitude sur la position de l'avion. Le mode full 4D revient à avoir un mode multi-RTA avec un nombre n infini de points RTA et avec une tolérance nulle. On retombe dans ce cas sur la modélisation du problème qu'on a faite dans le chapitre précédent, sans incertitude sur les position des avions.

4.3 Adaptation de LPA-traffic

Afin de prendre en compte d'emblée une situation réaliste de trafic aérien, nous considérerons l'incertitude précédemment décrite sur la position des avions dans notre traitement d'une journée de trafic sur la France. Dans un premier temps, on considérera directement le trafic avec une incertitude en boucle ouverte, c'est-à-dire sans imposer de point RTA. Cette étude permettra de constater l'impact de l'introduction de l'incertitude sur la détection et la résolution de conflits. Dans un deuxième temps, on se permettra d'imposer certains points RTA aux différentes trajectoires en conflit (l'incertitude sera alors en boucle fermée, ce qui donnera une marge de manœuvre supplémentaire pour résoudre les conflits). Ces points RTA seront imposés dans les cas où l'algorithme APLd n'aura pas réussi à générer des trajectoires sans conflit en considérant une incertitude en boucle ouverte sur la position de l'avion. Ces cas sont donc ceux où l'incertitude en boucle ouverte est trop importante pour permettre une résolution des conflits avec l'algorithme APLd.

Dans les deux cas, nous devons adapter l'APLd et, plus généralement, la méthodologie LPA-traffic,que nous avons développée dans la section 3.2.2, pour la résolution d'une journée de trafic sur la France. Dans cette section, on présentera les différentes adaptations qui doivent être apportées à LPA-traffic pour prendre en compte l'incertitude. Dans la suite, la méthodologie LPA-traffic adaptée aux incertitudes sera notée *u/LPA*.

4.3.1 Prédiction du trafic

Dans le contexte avec incertitudes, u/LPA s'effectue toujours par fenêtres glissantes, avec un horizon temporel de prédiction $\Delta t = 21$ min. Initialement (dans la version sans incertitudes), pour la prédiction du trafic sur une fenêtre temporelle donnée, on extrait les segments des trajectoires entre les instants t et $t + \Delta t$. Pour tenir compte de l'incertitude, nous considérons dans cette étape l'approche conservatrice du pire cas, c'est-à-dire le cas où l'avion arrive au point de la trajectoire où il était prévu d'arriver à $t + \Delta t$ avec une avance maximale. En d'autres termes on suppose que l'avion atteint ce point à l'instant $t+\Delta t-\delta t_{early}$, où δt_{early} est l'avance maximale correspondante dans la courbe d'incertitude (soit en boucle ouverte si on ne permet pas l'utilisation de RTA, soit en boucle fermée dans le cas où nous avons recours à l'imposition de RTA). Dans ce cas, pour prédire le

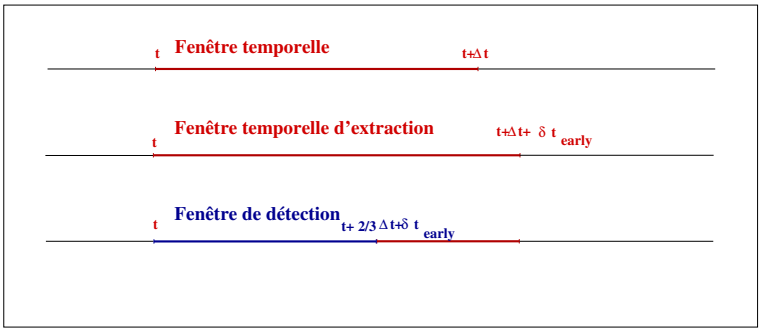


FIGURE 4.6 – Adaptation de la prédiction de trafic à l'incertitude.

trafic sur toute la fenêtre temporelle, nous devons extraire les segments de trajectoire entre les instants t et $t + \Delta t + \delta t_{early}$ (ainsi, on pourra voir les points qui seraient atteints par les avions à l'instant $t + \Delta t$, si ces avions avaient une avance maximale de δt_{early}). On doit par la suite détecter les conflits potentiels entre ces segments de trajectoire. Comme précédemment, pour retrouver les conflits au centre de la fenêtre temporelle, on détecte les conflits uniquement sur une fraction f (qu'on avait fixée $f = \frac{2}{3}$) de sa durée, donc entre l'instant t et l'instant $t + f\Delta t + \delta t_{early}$, δt_{early} étant l'avance maximale correspondant à l'instant $t + f\Delta t$ dans la courbe d'incertitude (Voir figure 4.6). La détection de conflits est l'objet du paragraphe suivant.

4.3.2 Détection de conflits

Comme précédemment, la détection de conflits est faite en deux étapes : une détection macroscopique des conflits et une détection des conflits réels.

Pour la détection macroscopique de conflits, il n'y a pas de modification par rapport au cas sans incertitudes. Il suffit d'associer à chaque segment prolongé (entre t et $t + f\Delta t + \delta t_{early}$), le parallélépipède rectangle qui l'englobe, et de détecter l'intersection de ces parallélépipèdes, déterminant ainsi les clusters macroscopiques de conflits.

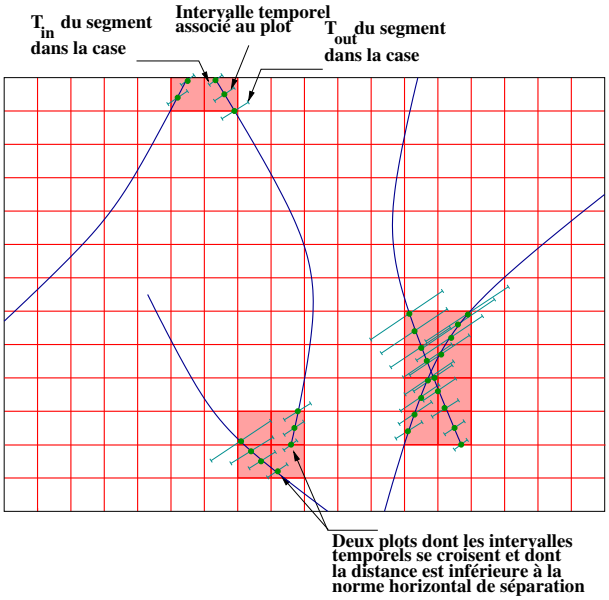


FIGURE 4.7 – Détection de conflits réels avec prise en compte de l'incertitude.

En revanche, pour ce qui concerne la détection des conflits réels, des modifications doivent être apportées, pour tenir compte de l'incertitude. En effet, initialement, les plots des segments appartenant à un cluster macroscopique de conflits étaient associés à des dates uniques (instants de passage de l'avion), puisqu'on ne considérait pas l'incertitude. Dans le contexte actuel avec incertitudes, chaque plot sera associé à un intervalle temporel d'incertitude issu de la courbe d'incertitude (en boucle ouverte ou en boucle fermée, selon le cas). La date t_{in} d'entrée d'un segment dans une case sera la borne inférieure de l'intervalle temporel associé au premier plot du segment dans cette case. Sa date t_{out} de sortie de la case sera la borne supérieure de l'intervalle temporel associé au dernier plot du segment dans cette case. Pour valider un conflit potentiel entre deux segments i et j qui se trouvent dans deux cases voisines c_i et c_j ou dans une même case $c_{i,j}$, on ne compare plus la distance entre les plots de i et j qui ont la même date, mais on compare la distance entre les plots de i et j dont les intervalles temporels associés ont une intersection non-vide (voir figure 4.7). La même règle pour détecter un cluster réel de conflit reste applicable, c'est-à-dire qu'au moins un couple des ces plots ait une distance inférieure à la norme horizontale de séparation.

4.3.3 Résolution de conflits

Dans cette partie, nous détaillons comment nous adapterons les différents aspects de la résolution des conflits doit afin de prendre en compte l'incertitude.

Enveloppe de protection

Comme dans le chapitre précédent, l'APLd adapté aux incertitudes, s'exécutera de façon séquentielle sur les segments de trajectoire des avions extraits entre les instants t et $t + \Delta t$, inclus dans un cluster de conflits. À la $n^{\text{ième}}$ étape d'exécution, les trajectoires contraintes des $n - 1$ avions déjà traités ne sont plus représentées par de simple tubes 4D dont la « section 3D » (une coupe 3D, transversale à l'axe temporel) à l'instant t est représentée par la figure 3.3, mais sont représentées par des *cônes* 4D, dont la section 3D à l'instant t est représentée par la figure 4.8. Cette section à l'instant t du cône 4D est en fait l'union de toutes les enveloppes de protection des positions possibles de l'avion qui correspondent à l'intervalle d'incertitude temporelle autour de l'instant t.

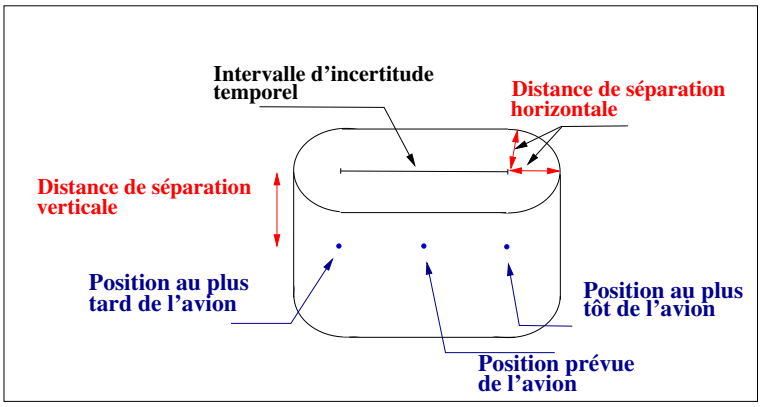


FIGURE 4.8 – Enveloppe de protection 3D d'un avion avec incertitude temporelle.

On peut d'ores et déjà apprécier l'évolution de la difficulté du problème de résolution de conflits. En effet, l'avion traité à la $n^{\text{ième}}$ étape d'exécution, devra éviter une zone interdite beaucoup plus importante que dans le cas sans incertitude.

Rayon lumineux

L'incertitude temporelle sur la position de l'avion, dont l'APLd est en train de chercher la trajectoire, devra aussi être prise en compte à l'étape de propagation du front d'onde, ce qui aura un impact sur la procédure de branchement au niveau du B&B mis en œuvre dans l'APLd.

En effet, initialement, c'est-à-dire dans le cas sans incertitude, un nœud de l'arbre de B&B représente une position spatiale de l'avion en cours de traitement à un instant donné t. Dans la nouvelle version prenant en compte les incertitudes, pour représenter la position au plus tôt et au plus tard de l'avion à l'instant t, il faut propager *deux* fronts d'onde autour du *front d'onde central* (le front d'onde initial, qui représente la position prévue de l'avion) : un front d'onde au plus tôt et un front d'onde au plus tard. Ces deux fronts d'onde peuvent être modélisés relativement au front d'onde central, en associant à chaque nœud de l'arbre représentant une position spatiale de l'avion, non pas un instant t mais

un intervalle d'incertitude temporelle autour de l'instant t. Cet intervalle d'incertitude est issue de la courbe d'incertitude (que ce soit en boucle ouverte, ou en boucle fermée).

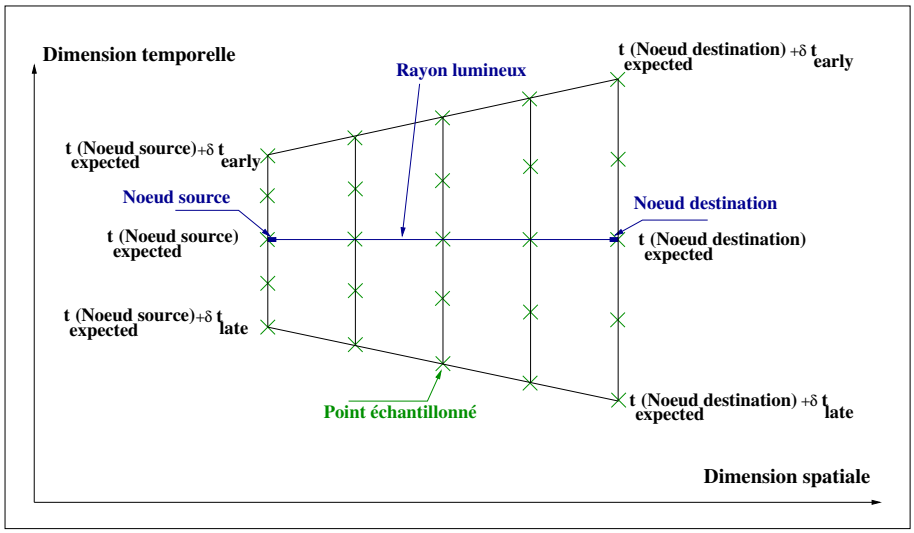


FIGURE 4.9 – Adaptation du rayon dans la phase de branchement de l'APL au cas avec l'incertitude : trapèze d'incertitude.

Comme précédemment, pour garantir la séparation des avions, on élimine d'emblée tout rayon lumineux qui pénètre une zone interdite (dans notre contexte avec incertitude : les cônes 4D des avions déjà traités). Initialement, le rayon lumineux avait un nœud source et un nœud destination, avec, pour chacun, un instant associé. Nous adaptons maintenant notre algorithme de sorte que, le rayon relie deux nœuds ayant chacun un intervalle d'incertitude temporelle associé. Il ne s'agit donc plus de vérifier si un segment pénètre une zone interdite, mais de vérifier si un *trapèze d'incertitude* (dans l'espace 4D) pénètre une zone interdite. Ce trapèze (figure 4.9) est défini par le segment initial (sans incertitude), avec chaque extrémité (nœud source et destination) représenté dans la dimension temporelle par l'intervalle d'incertitude correspondant. Afin de vérifier qu'il ne pénètre pas une zone interdite, le trapèze d'incertitude est échantillonné le long du segment spatial et de l'axe temporel et la vérification est faite pour chaque point (4D) échantillonné.

Ici encore, on peut apprécier l'augmentation de la difficulté du problème, puisque la trajectoire de l'avion en cours de traitement, n'est plus représentée par une suite de segments, mais par une suite de trapèzes dont les bases (dimension temporelle) sont de plus en plus grandes dans le cas de l'incertitude en boucle ouverte.

Points RTA

Comme mentionné précédemment, nous proposons une adaptation de notre méthodologie LPA-traffic (u/LPA) qui, dans une première phase, tente de résoudre (par exemple les conflits d'une journée de trafic) avec une incertitude en boucle ouverte sans recourir

à l'imposition de RTA. Dans une deuxième phase, dans l'éventualité où l'APLd ne réussit pas à générer des trajectoires sans conflits, l'u/LPA proposera l'imposition des points RTA, qui en réduisant l'incertitude des segments de trajectoire pourrait permettre la résolution des conflits résiduels (ceux non-résolus par la première phase). La façon dont on place ces points RTA sur une trajectoire donnée est une inconnue du problème, que nous devons déterminer. Ce sont les variables de décision d'un problème d'optimisation associé à cette deuxième phase d'u/LPA.

Pour la mise en œuvre de la deuxième phase de l'u/LPA, on doit trouver les positions (spatio-temporel) des points RTA, que l'APLd devra imposer aux avions pour générer des trajectoires sans conflits. Rappelons qu'un point RTA doit correspondre, et à un instant (un point sur l'axe du temps), et à un point spatial appartenant à la trajectoire (solution) générée par l'APLd. Il s'agit donc de contraindre l'avion à être à un endroit donné (qu'on appellera RTA spatial), à un moment précis (qu'on appellera RTA temporel). Cependant, imposer un RTA spatial implique que l'on connaisse à l'avance au moins un point de la trajectoire solution, qui évite les conflits. Or, on ne peut pas déterminer à l'avance (avant résolution) la zone de l'espace où va se trouver la trajectoire solution. Pour contourner ce problème, on impose uniquement un RTA temporel sans égard pour le moment à la position (RTA spatial). En d'autre termes, on choisit un instant RTA au voisinage duquel l'incertitude dans la dimension temporelle sera (par construction) réduite ; la position spatiale correspondante sera quant'à elle déterminée à posteriori, une fois que l'APLd aura généré une trajectoire solution. Ceci revient à dire, que pendant la construction de la trajectoire par l'APLd, et notamment lors de la procédure de branchement dans le B&B, le trapèze d'incertitude décrit précédemment aura des bases (dimension temporelle) qui correspondent à la courbe d'incertitude en boucle fermée. Cette courbe d'incertitude est complètement déterminée, par le RTA temporel imposé à la trajectoire en cours de construction. De même, les enveloppes de protection (cônes 4D) des trajectoires contraintes déjà traitées par l'APLd, sont déterminées par les courbes d'incertitude en boucle fermée selon les RTA temporels imposés à chacunes de ces trajectoires.

Il reste maintenant à préciser comment on détermine, un RTA temporel optimal à imposer à un avion particulier du cluster, de façon à générer in fine des trajectoires sans conflit. De façon intuitive, on soupçonne que pour générer des trajectoires sans conflit, il faut réduire l'incertitude sur des intervalles de temps où les avions sont les plus proches. En effet, la première difficulté des problèmes de résolution de conflit provient de la proximité spatiale des avions, à laquelle s'ajoute ensuite l'incertitude temporelle. Dans un premier temps, on propose donc d'imposer à tous les avions d'un cluster de conflit (non-résolu dans la première phase de l'u/LPA), un RTA temporel correspondant à l'instant où, avant la résolution et en considérant l'incertitude en boucle ouverte, chacun d'eux entrait en conflit avec un autre aéronef. En effet, cet instant appartient à l'intervalle de temps où, avant la résolution, les avions se rapprochaient le plus.

Illustrant cela sur l'exemple de la figure 4.4, avec deux avions dont les trajectoires se croisent. En récupérant l'instant t où les deux avions entrent en conflit, en fixant en suite le RTA temporel:= t et en exécutant l'APLd tel que décrit ci-haut, on obtient les trajectoires de la figure 4.10. L'algorithme s'appliquant de façon séquentielle aux trajectoires, l'avion

1 garde sa trajectoire initiale. Il est représenté à des instants successifs par son enveloppe de protection (dans la figure, en 2D). L'algorithme s'applique ensuite à l'avion 2. Sa trajectoire est représentée par l'union d'une suite de trapèzes (hachurée dans la figure) qui doivent éviter ces enveloppes de protection.

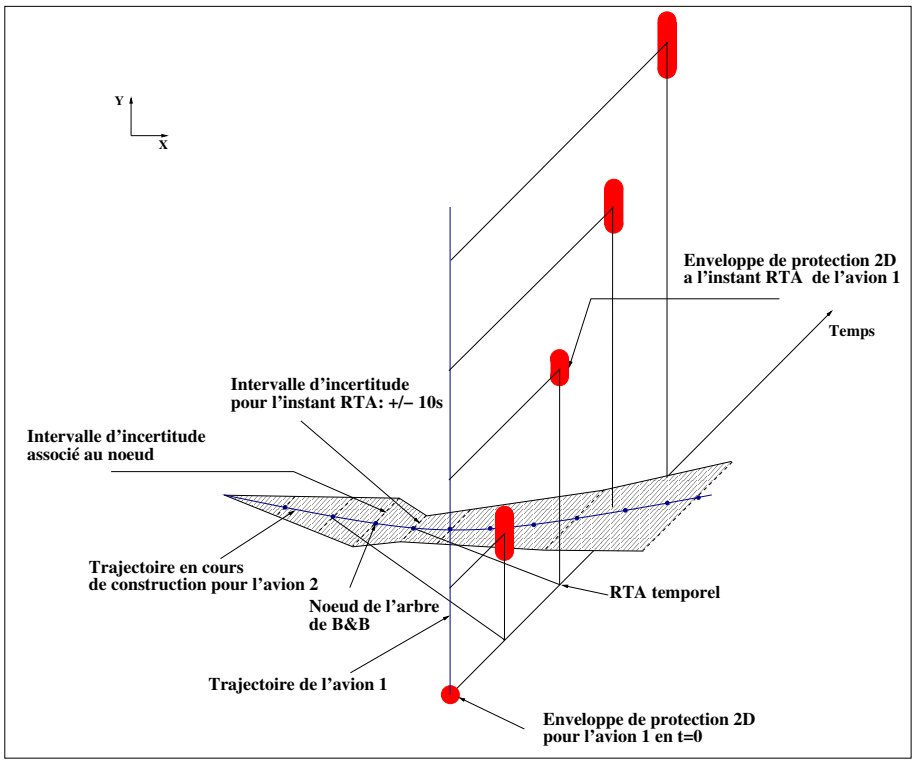


FIGURE 4.10 – Impact de l'imposition d'un RTA temporel(à l'instant où les avions entraient en conflit sur la figure 4.4) sur la résolution de conflit lors de la deuxième phase d'u/LPA.

Malheureusement, étant donné que l'APLd génère de nouvelles trajectoires en dispensant des changements de cap, les instants initiaux (avant résolution) où les conflits apparaissent (début de rapprochement dangereux entre deux ou plusieurs aéronefs) peuvent eux aussi être modifiés. Ainsi, placer le RTA temporel aux instants initiaux de début de conflits, ne permet plus de générer des trajectoires sans conflit. En effet, l'incertitude reste suffisamment importante au niveau des nouveaux points de rapprochement dangereux entre aéronefs (après changement de cap). Dans ce cas, nous proposons une nouvelle règle de placement des RTA temporels : à chaque fois que l'APLd ne trouve pas de solution pour un avion, on récupère son instant de passage prévu sur le nœud source du premier rayon lumineux (dans notre cas trapèze) qui pénètre dans une zone interdite. En effet, comme le parcours de l'arbre de B&B se fait en profondeur d'abord, il s'agit du premier instant où l'aéronef en cours s'approche de façon dangereuse d'un autre aéronef. Les RTA temporels seront fixés, dans cette seconde version, à ces nouveaux instants de rapprochement dangereux.

Nous présentons dans la prochaine section, les résultats obtenus lorsque nous appliquons notre méthodologie u/LPA sur un problème réel de résolution de conflits tenant compte de l'incertitude sur la position longitudinale des avions.

4.4 Résultats numériques

Dans cette section, nous détaillons les résultats obtenus par notre méthodologie u/LPA sur la même journée de trafic sur la France que celle traitée au chapitre précédent (journée de trafic réel du 12 août 2008) prenant cette fois en compte l'incertitude sur la position de l'avion le long de sa trajectoire (incertitude longitudinale). Tout d'abord, nous détaillons la première phase d'u/LPA qui correspond au cas sans imposition de RTA. Ensuite, nous appliquons la deuxième phase d'u/LPA, celle où nous avons recours à des RTA pour résoudre les conflits résiduels. Enfin, nous finissons par une analyse des résultats.

Phase 1 : incertitude en boucle ouverte

Dans ce paragraphe, on applique la première phase d'u/LPA, c'est-à-dire qu'on considère l'incertitude en boucle ouverte. Notre but est d'observer l'influence de l'incertitude sur le nombre de clusters de conflits détectés et résolus. On présente ici, les résultats que l'on a obtenus pour la demi-journée de trafic du 12 août 2008 allant de minuit à midi. En effet, sur une journée, deux maxima de trafic apparaissent à 8 heures et à 18 heures. Travailler sur une demi-journée de minuit à midi est donc représentatif des différentes situations de trafic sur la journée entière.

Les trajectoires initiales (avant la résolution des conflits) induisent ici un nombre total de clusters de conflits, comprenant de 2 jusqu'à 18 avions en conflit réel, égal à 2688. Notons que pour la même demi-journée, on obtenait, sans tenir compte de l'incertitude, seulement 1611 clusters de conflits. L'augmentation du nombre de conflits détectés est conforme à nos attentes.

Alors que l'APLd résolvait précédemment presque tous les conflits, avec uniquement 11 clusters pour lesquels il n'a pas réussi à trouver pour tous les avions des trajectoires alternatives sans conflit (cas où les avions étaient déjà en conflit au début de la simulation), à présent, il y a 307 clusters que l'APLd ne réussit plus à résoudre. On résout donc ainsi 88% des conflits lorsqu'on prend en compte l'incertitude en boucle ouverte contre 99% précédemment, ce qui est en dessous de l'objectif opérationnel de résolution de 95% des cas. Ceci confirme la difficulté de ce nouveau problème prenant en compte l'incertitude. Pour atteindre une telle situation (88% des conflits résolus en présence d'incertitude), 1805 trajectoires ont été modifiées, alors que précédemment seules 1349 trajectoires sont modifiées.

Comme précédemment, dans de nombreux cas, les nouvelles trajectoires calculées en tenant compte de l'incertitude sont plus courtes que les trajectoires initiales, car l'algorithme recherche toujours le plus court chemin et propose un chemin direct dès que possible. En moyenne, la longueur des trajectoires calculées est raccourcie de 3,78 Nm par avion (contre 4,32 Nm précédemment), avec un minimum de 40,6 Nm (contre 51,9 Nm

précédemment). D'autre part, certaines trajectoires subissent, cette fois encore, une extension de leur longueur avec un maximum de 12,49 Nm (contre 9,86 Nm pour le cas sans incertitudes). Les quantiles associés à la modification de la longueur des trajectoires sont présentés dans la table 4.1.

Quantile	Distance en Nm
$\frac{1}{4}$ quantile	-6,39
$\frac{1}{2}$ quantile	-2,08
$\frac{3}{4}$ quantile	+0,06

TABLE 4.1 – Modification de la longueur des trajectoires après résolution par l'application de la première phase d'u/LPA (incertitude en boucle ouverte).

Le temps de calcul global est d'environ 12 heures, contre 9 heures précédemment (résolution sans incertitudes) pour la même demi-journée de trafic.

Comme on ne réussit pas à atteindre l'objectif de résolution à 95% des cas, nous appliquons au paragraphe suivant la deuxième phase de notre méthodologie u/LPA qui permet l'imposition des points RTA, dans le but d'améliorer nos résultats.

Phase 2 : mise en œuvre des RTA

Dans ce paragraphe, nous appliquons les deux règles précédemment décrites pour le placement des RTA temporels. Rappelons que la règle 1 consiste à placer les RTA temporels aux instants où les trajectoires de deux ou plusieurs avions entraient en conflit avant la résolution. La règle 2 est, quant à elle, utilisée si, avec la régle 1, l'APLd échoue à générer une trajectoire sans conflit, pour un avion a d'un cluster. Dans ce cas, le RTA temporel pour l'avion a est fixé à l'instant où sa trajectoire en construction (avec la régle 1) entrait pour la première fois dans une zone interdite (enveloppe de protection des autres avions). On traite ici aussi la même demi-journée de trafic du 12 août 2008, de minuit à midi.

Les trajectoires initiales (avant la résolution des conflits) induisent ici, un nombre total de clusters de conflits, comprenant de 2 jusqu'à 16 avions en conflits réels, égal à 3174. On obtient un nombre supérieur de clusters de conflits au nombre obtenu dans le cas sans RTA. Ce ci peut être expliquer comme suit. Quand on a appliqué uniquement la phase 1 de l'u/LPA, les clusters, pour lesquels on n'a pas trouvé de solutions, ont été comptabilisés comme non-résolus, mais les trajectoires des avions impliqués dans ces clusters n'ont pas été modifiées. Avec la mise en œuvre des RTA, certain clusters ont été résolus et les trajectoires des avions impliqués ont été modifiées pour la fenêtre temporelle courante, mais peuvent avoir indui des conflits sur les fenêtres temporelles suivantes.

Comme prévu, en réduisant l'incertitude grâce aux points RTA, le nombre de clusters de conflits résolus diminue. A présent, il ne reste plus que 210 clusters non résolus, ce qui fait passer le pourcentage de résolution à 93%, un pourcentage toujours inférieur à l'exigence opérationnelle qui est de 95%. Pour obtenir un tel résultat, 2116 trajectoires ont été modifiées, soit en leur appliquant des points RTA uniquement (seuls les avions qui

on été traités en premier (dans l'ordre de résolution séquentielle) par l'APLd garde leur trajectoires initiales mais peuvent se voir imposer un point RTA), soit en leur imposant des changements de cap uniquement (pour les avions impliqués dans un cluster qui a été résolu par la phase 1 de l'u/LPA), soit les deux. Ainsi, 697 points RTA ont été imposés à l'ensemble des trajectoires traitées, avec un maximum de 5 points RTA par trajectoire.

En ce qui concerne le changement de longueur des nouvelles trajectoires solutions, elles ont été raccourcies en moyenne par 4,15 Nm par avion, avec un minimum de 46,6 Nm. Leur extension maximale est de 19,94 Nm. Les quantiles associés à la modification de la longueur des trajectoires sont présentés dans le table 4.2 :

Quantile	Distance en Nm
$\frac{1}{4}$ quantile	-7,31
$\frac{1}{2}$ quantile	-2,37
$\frac{3}{4}$ quantile	-0,08

TABLE 4.2 – Modification de la longueur des trajectoires après résolution par l'APL, avec incertitude en boucle fermée.

Le temps de calcul global est d'environ 14 heures pour cette même demi-journée de trafic.

Analyse des résultats

En prenant en compte un trafic plus réaliste que celui considéré au chapitre précédent, c'est-à-dire en tenant compte de l'incertitude longitudinale sur la position des avions, on a pu observer une augmentation considérable de la difficulté du problème. En effet, le nombre de conflits détectés est une fois et demie supérieur à celui du cas sans incertitude, dû à une croissance du volume des zones de protection, pour la même situation de trafic. D'autre part, le pourcentage de résolution de conflit chute à 88% dans le cas de l'incertitude en boucle ouverte, et s'améliore pour atteindre 93% en appliquant des points RTA avec les règles qu'on a mises en place dans ce chapitre. Malheureusement, ce chiffre reste en deçà de l'exigence opérationnelle de 95%. Deux pistes apparaissent pour expliquer ce résultat mitigés. La première est que les situations où l'APLd n'a pas réussi à trouver de solution, sont des situations très difficiles à résoudre, et dans ce cas, il faudrait appliquer un contrôle de type full 4D aux trajectoires concernées, au moins sur la fenêtre temporelle sans solution. La deuxième vient des règles de placement des RTA temporelles. En effet, on a donné des règles intuitives pour le placement des RTA, mais il existe probablement des règles plus judicieuses qui permettraient d'atteindre l'objectif opérationnel de 95%. Des travaux futurs pourraient impliquer la recherche de telles règles.

Conclusion

La croissance de trafic aérien dans les prochaines décennies est un défi majeur pour le système de gestion du trafic aérien ATM qui doit assurer un écoulement sûr et efficace des flux de trafic. Dans cette perspective, deux projets ambitieux d'aide au contrôle ont été lancés en Europe et aux États-Unis (SESAR et NextGEN respectivement). L'une des caractéristiques essentielles de ces projets est de contraindre davantage les aéronefs sur leurs trajectoires, principalement dans les zones congestionnées. Dans le cadre de la présente étude, nous avons considéré cette contrainte qui s'applique dans la dimension temporelle afin de réduire l'incertitude au niveau de l'abscisse curviligne de l'avion sur sa trajectoire. Ce principe permet alors d'améliorer la détection et la résolution des conflits qui sont déterminants dans le cadre du contrôle du trafic aérien.

Cette fonctionnalité est cependant assez contraignante pour les aéronefs et elle ne doit être utilisée que lorsque les autres modes classiques de résolution ont échoué.

Cette thèse a permis de démontrer qu'il était possible de synthétiser automatiquement des trajectoires sans conflit en utilisant de simples déviations latérales (sans avoir recours à des RTA) sur un nombre substantiel de situations de trafic.

Contribution

Dans un premier temps, nous avons développé un algorithme de planification de trajectoires basé sur l'analogie de la propagation lumineuse. Le principe général de cette approche est de simuler la propagation d'une onde lumineuse du point origine vers le point de destination. Cette analogie permet de générer des trajectoires optimales en temps pour lesquelles il est possible de contrôler la vitesse des avions en fixant des valeurs d'indice de réfraction adéquats. De plus, les trajectoires ainsi générées sont les plus régulières possibles (propriété des géodésiques). Cet algorithme, nommé APL, permet de produire des trajectoires optimales pour éviter des obstacles fixes (à l'échelle de la fenêtre temporelle considérée), comme par exemple des zones congestionnées ou des zones de mauvaise météo.

Afin de prendre en compte des obstacles mobiles, nous avons effectué l'extension dynamique (temporelle) de l'algorithme précédent dans le but de produire des trajectoires optimales dans un référentiel à quatre dimensions (trois dimensions spatiales et une dimension temporelle). L'extension naïve de l'algorithme dans cette dimension temporelle peut aboutir dans certain cas à des vitesses d'aéronef qui viole les contraintes de bornes qui garantissent un vol sûr. Afin d'éviter cet inconvénient, nous avons adapté l'algorithme

en relâchant, au niveau du point de destination, la contrainte de concomitance temporelle que nous avons remplacée par un segment temporel cible. Ceci permet alors d'assurer aux avions de rester dans le profil de vitesse initial. Nous avons ensuite testé l'algorithme sur une situation de trafic artificielle pour laquelle nous avons trouvé le même type de solution que d'autres approches connues. Enfin, nous avons appliqué l'APLd sur une journée complète de trafic à l'échelle de la France. Pour cela, nous avons développé un mécanisme efficace de clusterisation spatiale des conflits, permettant d'accélérer la résolution. Pour cette journée type, tous les conflits ont été résolus avec succès, sauf ceux, insolubles, pour lesquels les avions était déjà en conflit au début de la simulation.

Afin de prendre en compte l'incertitude de la prédiction de trajectoire liée aux aléas que rencontre l'avion (principalement le vent), nous avons produit une extension robuste de notre algorithme. L'incertitude, étant essentiellement localisée au niveau de l'abscisse curviligne de l'avion sur sa trajectoire, nous avons remplacé la position ponctuelle de chaque avion par un segment dont la taille grandit avec le temps. Dans ce cadre, pour déterminer sa trajectoire, chaque avion effectue la propagation conjointe de deux fronts d'onde (une front amont et un front aval) et considère les autres avions non plus comme des obstacles avec une zone de protection circulaire (dans le plan horizontale) mais ayant une forme oblongue étendue dans la dimension temporelle. Cette modification, conservatrice, augmente considérablement le nombre de conflits détectés et empêche parfois l'algorithme que nous avons développé de trouver des solutions (en modifiant les trajectoires en cap uniquement). Dans ce cas, l'algorithme est relancé en positionnant des points RTA (contraignant les avions en temps) afin de produire une solution sans conflit. La méthodologie complète est nommée u/LPA.

En résumé, les principales contributions de notre travail sont les suivantes.

- **Développement d'un nouvel algorithme, appelé APL, de planification de trajectoires basé sur une analogie de propagation lumineuse. Cette version de l'algorithme permet de planifier des trajectoires avec des obstacles statiques (évitement de zones avec une mauvaise météo ou de zones congestionnées).**
- **Extension temporelle permettant de pendre en compte des obstacles dynamiques avec application à la résolution de conflits tactiques (algorithme APLd).**
- **Extension robuste avec ajout d'aléas dans la prédiction de trajectoires et mise en place de points RTA au sein d'une méthodologie complète, nommée u/LPA, adaptée aux cas complexes de trafic réel.**.

Perspectives

Dans un premier temps, des travaux futurs devraient se concentrer sur des façons alternatives de placer des points RTA.

Dans un deuxième temps, si le placement des points RTA ne permet toujours pas de traiter les situations complexes de trafic, il faudrait considérer l'utilisation d'un levier d'action supplémentaire : la modification de la vitesse des avions afin de traiter les conflits restants. L'approche que nous avons priviligiée dans cette étude, utilise le profil de vitesse

optimal des avions, afin de satisfaire au mieux les demandes des compagnies aériennes. Une nouvelle approche permettant d'accélérer ou de décélérer les avions dans la limite de ses capacités ($[-6\%, +3\%]$ de la vitesse initiale de l'avion) engendrera de nouveaux défis motivants, à la fois en termes de modélisation et au regard des algorithmes d'optimisation en jeu.

Annexe A

Algorithmes génétiques

Les premiers travaux sur les algorithmes génétiques ont commencé dans les années cinquante lorsque plusieurs biologistes américains ont simulé des structures biologiques sur ordinateur. Puis entre 1960 et 1970, John Holland [54], sur la base des travaux précédents, développe les principes fondamentaux des algorithmes génétiques dans le cadre de l'optimisation. Malheureusement, les ordinateurs de l'époque n'étaient pas assez puissants pour envisager l'utilisation des algorithmes génétiques sur des problèmes réels de grande taille. La parution de l'ouvrage de référence écrit par Goldberg [51], qui décrit l'utilisation de ces algorithmes dans le cadre de résolution de problèmes concrets, a permis de mieux faire connaître ces derniers à la communauté scientifique et a marqué le début d'un nouvel intérêt pour cette technique d'optimisation. Parallèlement, des techniques proches ont été élaborées, dont les principales sont les suivantes :

- *Evolution Strategy* [96] ;
- *Evolutionary Programming* [42] ;
- *Genetic Programming* [65].

Dans le même temps, la théorie mathématique associée aux algorithmes génétiques s'est développée mais reste encore bien limitée face à la complexité théorique induite par ces algorithmes. Il faudra attendre 1993 pour qu'une démonstration de convergence stochastique soit établie [18, 44].

Les algorithmes génétiques s'attachent à simuler le processus de sélection naturelle dans un environnement hostile lié au problème à résoudre [24]. Ils utilisent un vocabulaire similaire à celui de la génétique naturelle, tout en rappelant que les principes sous-jacents liés à ces deux domaines sont beaucoup plus complexes dans le cadre naturel. On parlera ainsi d'*individu* dans une *population* et bien souvent l'individu sera résumé par un seul *chromosome*. Les chromosomes sont eux-mêmes constitués de gènes qui contiennent les caractères héréditaires de l'individu. On retrouvera aussi les principes de *sélection*, de *croisement*, de *mutation*, etc.

Dans le cadre de l'optimisation, chaque individu représente un point de l'espace d'état auquel on associe la valeur du critère à optimiser. On génère ensuite une population d'individus aléatoirement pour laquelle l'algorithme génétique s'attache à sélectionner les meilleurs individus tout en assurant une exploration efficace de l'espace d'état.

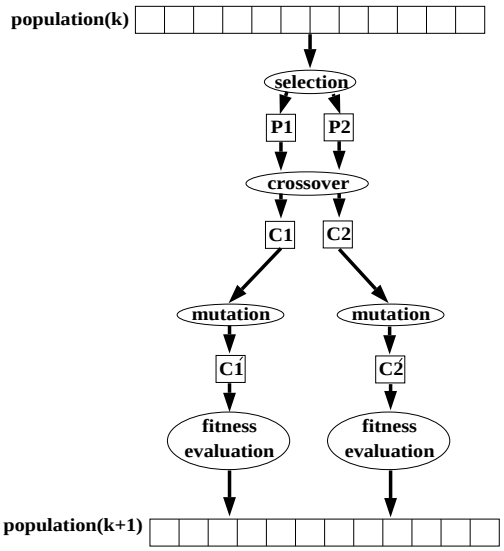


FIGURE A.1 – Principe général des algorithmes génétiques.

Les applications des algorithmes génétiques sont diverses et variées ; c'est ainsi qu'ils ont donné de bons résultats dans des problèmes d'ordonnancement [62], de contrôle adaptatif [41], dans le problème du voyageur de commerce [55], dans les problèmes de transport [111], dans la synthèse de forme [109], dans les réseaux de neurones [95], dans la synthèse moléculaire [106], dans le domaine médical [43], en filtrage [94, 8], dans les problèmes de contrôle du trafic aérien [3, 26, 28, 27, 36] etc.

Les opérations successives utilisées dans les algorithmes génétiques sont décrites sur la figure A.1.

1. On commence par générer une population d'individus de façon aléatoire (la taille N_p de la population est à paramètre à choisir par l'utilisateur).
2. Deux parents sont ensuite sélectionnés (P_1 et P_2) en fonction de leurs adaptations au critère. On applique aléatoirement l'opérateur de croisement avec une probabilité P_c (choisi par l'utilisateur) qui génère deux enfants C_1 et C_2. On modifie ensuite

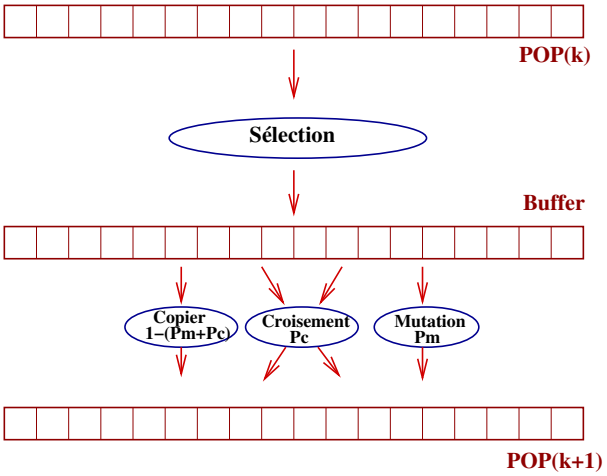


FIGURE A.2 – Autre forme équivalente.

certains gènes de C_1 et C_2 en appliquant l'opérateur de mutation avec la probabilité P_m (choisi par l'utilisateur), ce qui produit deux nouveaux individus C'_1 et C'_2 pour lesquels on évalue le niveau d'adaptation avant de les insérer dans la nouvelle population. On réitère les opérations de sélection, de croisement et de mutation afin de compléter la nouvelle population, ceci termine le processus d'élaboration d'une génération (après élimination possible de certains individus).

On reboucle ensuite N fois sur l'étape 2 (N, qui est un paramètre, représente le nombre total de générations).

On trouve parfois une autre forme utilisant une population intermédiaire qui commence par effectuer la sélection de façon globale puis applique les opérateurs de recombinaison sur les individus sélectionnés (voir figure A.2).

Pour utiliser un algorithme génétique sur un problème particulier, on doit donc disposer des six éléments suivants :

1. Un principe de codage du chromosome. Cette étape associe à chacun des points de l'espace d'état une structure de données qui synthétise toute l'information liée à ces derniers et se place donc après la phase de modélisation mathématique (voir figure A.3).
2. Un mécanisme de génération de la population initiale. Ce mécanisme doit être capable de produire une population d'individus (par exemple uniformément répartis) qui servira de base pour les générations futures.
3. Un critère permettant de déterminer l'adaptation d'un individu par rapport à l'environnement afin de classer les individus entre eux. Dans le vocabulaire des algorithmes génétiques, on nomme ce critère *fitness* (adaptation, en anglais).
4. Un principe de sélection permettant d'identifier les meilleurs individus. On sattachera à régler efficacement la pression sélective qui permet de favoriser (*intensification*) plus ou moins les individus ayant une fitness plus importante.

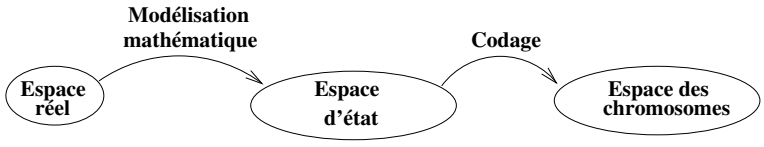


FIGURE A.3 – Le codage représente une étape supplémentaire par rapport à la phase de modélisation mathématique. On peut ainsi envisager plusieurs codages possibles pour un même espace d'état.

5. Des opérateurs permettant de diversifier la population au cours des générations en explorant l'espace d'état. De façon générale, on utilise deux opérateurs qui sont le croisement et la mutation. Le croisement vise à brasser les gènes des individus dans la population tandis que l'opérateur de mutation a pour but de générer de nouveaux gènes. La bonne adéquation du principe de codage et des opérateurs par rapport au problème traité conditionne le succès ou l'échec des algorithmes génétiques et il faudra donc apporter un soin très important à la conception de ces derniers.
6. Des paramètres de dimensionnement (taille de la population, nombre de générations à simuler, probabilités d'application des opérateurs) qui doivent être choisis par l'utilisateur de façon empirique.

Les méthodes évolutionnaires permettent de traiter efficacement des problèmes réels et fournissent de bonnes solutions pour des critères quelconques, sans hypothèses de convexité de différentiabilité, de continuité ; des boites noires ; des problèmes multi-objectifs et ce, dans des espaces de recherche de grandes dimensions. Leur principe est simple et nécessite très peu d'information par rapport au processus que l'on cherche à optimiser. Seule la capacité à évaluer les critères en tout point est requise. On peut ainsi envisager de les utiliser dans le cadre de processus d'optimisation pour lesquels la valeur de la fonction-objectif est fournie par un processus réel de simulation. Par contre, il faut bien se rappeler que ces algorithmes, complètement génériques, ne doivent être envisagés que lorsqu'il n'existe pas de méthode déterministe permettant de traiter le problème de façon efficace par exemple en exploitant une structure particulière du problème (comme les dérivées ou les propriétés mathématiques tel que la linéairité, la convexité etc.).

Annexe B

Intersection parallélépipèdes

B.1 Détection sur l'axe x

DetectX(EnsemblePara) [*EnsemblePara est l'ensemble des parallélépipèdes à traiter*]

Result $\leftarrow \emptyset$
EnsembleBornes $\leftarrow$ Projeter EnsemblePara sur l'axe x [*On obtient un ensemble de bornes min et max pour chaque intervalle qui est la projection d'un parallélépipède sur l'axe horizontal*]
Trier(EnsembleBornes) [*Trier par ordre croissant*]
Taille $\leftarrow$ Cardinal(EnsembleBornes)
Indice $\leftarrow$ 1
Tant que (Indice $\leq$ Taille) faire
  Borne $\leftarrow$ EnsembleBornes(Indice)
  Si (Borne est min) faire [*Borne est soit une borne min, soit une borne max*]
    id $\leftarrow$ id(Borne) [*Récuperer l'identifiant correspondant à Borne*]
    $\text{Assoc}_{id} \leftarrow$ clusterisation(id, EnsembleBornes(Indice+1,Taille))
    [*La fonction clusterisation rend l'ensemble des identifiants correspondant aux intervalles suivants qui ont une intersection avec l'intervalle courant (id)*]
    Si(Cardinal(Assoc_{id}) > 0) faire
      clusterId $\leftarrow$ Result.get(id) [*Récuperer l'ensemble des intervalles déjà associés à l'intervalle (id) dans Result*]
      Si (clusterId $= \emptyset$) faire
        Result $\leftarrow$ Result $\cup$ (id, Assoc_{id})
      Sinon
        clusterId $\leftarrow$ clusterId $\cup$ Assoc_{id}
      fin Si
      [*Pour tous les éléments id_2 de Assoc_{id}, mettre id dans la liste des intervalles associés à id_2*]
      $\text{Taille}_2 \leftarrow$ Cardinal(Assoc_{id})
      $\text{Indice}_2 \leftarrow 1$

```
            Tant que (Indice_2 ≤ Taille_2) faire [Parcourir les éléments de Assoc_id]
               id_2 ← Assoc_id(Indice_2)
               clusterId_2 ← Result.get(id_2)
               Si (clusterId_2 = ∅) faire
                  clusterId_2 ← clusterId_2 ∪ id
                  Result ← Result ∪ (id_2, clusterId_2)
               Sinon
                  clusterId_2 ← clusterId_2 ∪ id
               fin Si
               Indice_2 ← Indice_2 + 1
            fin Tant que
         fin Si
      fin Si
      Indice ← Indice+1
   fin Tant que
   retourner Result
```

B.2 Détection sur l'axe y

DetectY(EnsemblePara) [*EnsemblePara est l'ensemble des parallélépipèdes à traiter*]

Cluster$_X$ ← DetectX(EnsemblePara) [*Retrouver l'ensemble des clusters (liste de parallélépipèdes associés) sur l'axe horizontal*]
Pour (tout id dans Cluster$_X$.KeySet()) faire [*Parcourir la liste des identifiants (id) des parallélépipèdes qui ont une liste de parallélépipèdes associés*]
  (min$_{id}$, max$_{id}$)← Projeter le parallélépipède dont l'identifiant est id sur l'axe y
  [*On obtient les bornes min et max de l'intervalle associé à id sur l'axe y*]
  cluster$_{id}$ ← Cluster$_X$.get(id)[*Récuperer la liste de parallélépipèdes associées à id*]
  Indice ← 1
  Tant que (Indice ≤ Cardinal(cluster$_{id}$)) faire
    idAutre ← cluster$_{id}$(Indice)
    (min$_{Autre}$, max$_{Autre}$)← Projeter le parallélépipède dont l'identifiant est idAutre sur l'axe y
    [*On obtient les bornes min et max de l'intervalle associé à idAutre sur l'axe y*]
    Si (min$_{id}$ > max$_{Autre}$ ou min$_{Autre}$ > max$_{id}$) faire [*Condition de non chevauchement des intervalles*]
      cluster$_{id}$.remove(idAutre) [*Enlever idAutre de la liste des parallélépipèdes associés à id*]
      Si (Cluster$_X$.get(idAutre) ≠ ∅) faire
        Cluster$_X$.get(idAutre).remove(idAutre)
        [*Enlever id de la liste des parallélépipèdes associés à idAutre*]
      fin Si
    fin Si
    Indice ← Indice+1
  fin Tant que
fin Pour
retourner Cluster$_X$

B.3 Fusion

Fusion (id,Cluster$_Z$) [*id est l'identifiant d'un parallélépipède, Cluster$_Z$ est l'ensemble des associations issu de la fonction DetectZ*]

cluster$_{id}$ ← Cluster$_Z$.get(id)[*Récuperer la liste de parallélépipèdes associées à id*]
Cluster$_Z$.remove(cluster$_{id}$) [*Éviter une récursivité infinie*]
Indice ← 1
Tant que (Indice ≤ Cardinal(cluster$_{id}$)) faire
  idAutre ← cluster$_{id}$(Indice)
  cluster$_{Autre}$ ← Cluster$_Z$.get(idAutre)
  Si (cluster$_{Autre}$ ≠ ∅) faire
    cluster$_{Autre}$.remove(id)
    cluster$_{Autre}$ ← Fusion(idAutre,Cluster$_Z$)
    cluster$_{id}$ ← cluster$_{id}$ ∪ cluster$_{Autre}$
  fin Si
  Indice ← Indice+1
fin Tant que
cluster$_{id}$ ← cluster$_{id}$ ∪ {id}
retourner cluster$_{id}$ [*La fonction retourne un cluster de parallélépipèdes contenant id*]

Bibliographie

[1] N. Alech, N. Durand, J. M. Alliot et M. Schoenauer : Genetic algorithms for optimal plane conflict resolution. *In Second Singapore International Conference on Intelligent Systems*, 1994.

[2] J. M. Alliot, J.F. Bosc, N. Durand et L. Maugis : CATS : A complete air traffic simulator. *In 16th AIAA/IEEE Digital Avionics Systems Conference*, 1997.

[3] J. M. Alliot et H. Gruber : Using genetic algorithms for solving ATC conflicts. *In Proceedings of the Ninth IEEE Conference on Artificial Intelligence for Applications*. IEEE, 1993.

[4] L. Angerand et H. LeJeannic : Bilan du projet SAINTEX. Rapport technique R92009, Centre d'études de la navigation aérienne, 1992.

[5] D. Arbuckle : *NextGen ATM Concept of Operations : ASAS-Reliant*, septembre 2007.

[6] N. Archambault : Speed uncertainty and speed regulation in conflict detection and resolution in air traffic control. *In The 1st International Conference on Research in Air Transportation (ICRAT)*, novembre 2004.

[7] B. Bakker : Specification for short term conflict alert. Rapport technique No. 1.1, Eurocontrol, mai 2009.

[8] J.W. Bala et H. Wechsler : Learning to detect targets using scale-space and genetic search. *In Proceedings of the Fifth International Conference on Genetic Algorithms*. ICGA, 1993.

[9] E. Balas et P. Toth : *Branch and bound methods in the traveling salesman problem*. John Wiley & Sons, 1985. pp 361-401.

[10] N. Barnier et C. Allignol : 4D-trajectory deconfliction through departure time adjustment. *In The Eighth USA/Europe Air Traffic Management Research and Developpment Seminar, Napa, Californie, États-Unis*. ATM Seminar, juillet 2009.

[11] N. Barnier et P. Brisset : Facile : A functional constraint library. *In Colloquium on Implementation of Constraint and LOgic Programming Systems, CICLOPS'01*, décembre 2001.

[12] M. Ben-Akiva, A. De Palma et P. Kanaroglou : Dynamic model of peak period traffic congestion with elastic arrival rates. *Transportation Science*, 20:164–181, août 1986.

[13] D. Bertsimas et S. Stock Patterson : The air traffic flow management problem with enroute capacities. *Operations Research*, 46:406–422, 1998.

[14] J.T. Betts et W.P. Huffman : Trajectory optimization on a parallel processor. *Journal of Guidance, Control and Dynamics*, 14(2):431–439, 1991.

[15] J.F. Bosc : *Techniques d'évitement réactif et simulation du trafic aérien*. Thèse de doctorat, en informatique de l'Institut National Polytechnique de Toulouse, 1997.

[16] D. Bouteloup : *Éléments de géométrie différentielle*. École national de sciences géographiques, 2 édition, 2003.

[17] J.C. Celio : Controller perspective of aera2. Rapport technique, The MITRE corporation, février 1990.

[18] R. Cerf : *Une théorie asymptotique des algorithmes génétiques*. Thèse de doctorat, Université Montpellier II, France, 1994.

[19] G. Chaloulos, J. Lygeros et E. Siva A. Lecchini-Visintini P. Casek I. Roussos, K. Kyriakopoulos : Comparative study of conflict resolution methods. Rapport technique, iFly, mai 2007.

[20] E. Chang, R. Hu, D. Lai, R. Li, Q. Scott et T. Tyan : The story of mode s. Rapport technique 6.933, Massachusetts Institute of Technology, décembre 2000.

[21] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki et S. Thrun : *Principles of Robot Motion : Theory, Algorithms, and Implementations*. MIT Press, Cambridge, MA, 2005.

[22] J. Clausen et M. Perregaard : On the best search strategy in parallel branch-and-bound - best-first-search vs. lazy depth-first-search. *In Proceedings of POC96*, 1996.

[23] D. Colin De Verdière : Le système français de contrôle du trafic aérien : Le CAUTRA. Rapport technique, Centre d'études de la navigation aérienne, Toulouse, août 1992.

[24] L. Davis : *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New-York, 1991.

[25] M.D. Davis : *La theorie des jeux*. Armand Colin, 1973.

[26] D. Delahaye, J. M. Alliot, M. Schoenauer et J. L. Farges : Genetic algorithms for partitioning airspace. *In Proceedings of the Tenth IEEE Conference on Artificial Intelligence for Application*. CAIA, 1994.

[27] D. Delahaye, J. M. Alliot, M. Schoenauer et J. L. Farges : Genetic algorithms for automatic regrouping of air traffic control sectors. *In Evolutionary Programming'95*, pages 657–672, 1995.

[28] D. Delahaye, J.M. Alliot, M. Schoenauer et J.L. Farges : Genetic algorithms for air traffic assignment. *In Proceedings of the European Conference on Artificial Intelligence*, pages 33–37. ECAI, 1994.

[29] E. W. Dijkstra : A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[30] M. Dorigo, V. Maniezzo et A. Colorni : Ant system : Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, 26(1):29–41, 1996.

[31] V. N. Duong et P. Faure : On the applicability of the free-flight mode in European airspace. *In The 2nd USA/Europe Air Traffic Management R&D Seminar*, décembre 1998.

[32] V. N. Duong, E. Hoffman et J. P. Nicolaon : Initial results of investigation into autonomous aircraft concept (FREER-1). *In The 1st USA/Europe Air Traffic Management R&D Seminar*, 1997.

[33] V. N. Duong, E. Hoffman, J. P. Nicolaon, L. Floc'hic et A. Bossu : Extended flight rules to apply to the resolution of encounters in autonomous airborne separation. Rapport technique, Eurocontrol, 1996.

[34] N. Durand : *Optimisation de trajectoires pour la résolution de conflits en Route*. Thèse de doctorat, ENSEEIHT, Institut National Polytechnique de Toulouse, France, 1996.

[35] N. Durand : *Algorithmes génétiques et autres méthodes d'optimisation appliquées à la gestion de trafic aérien*. Habilitation à diriger des recherches. Institut National Polytechnique de Toulouse, France, 2004.

[36] N. Durand, N. Alech, J. M. Alliot et M. Schoenauer : Genetic algorithms for optimal air traffic resolution. *In Proceedings of the second Singapore Conference on Intelligence Systems*. SPICIS, 1994.

[37] N. Durand et J. M. Alliot : Ant colony optimization for air traffic conflict resolution. *In The Eighth USA/Europe Air Traffic Management Research and Development Seminar, Napa, Californie, États-Unis*, juillet 2009.

[38] L. Epstein, A. Futer et L. Medvedovsky : Multiple airport scheduling. Rapport technique, Volpe National Transportation Systems Center, Cambridge, MA, USA, 1992.

[39] F. Krella et al : ARC 2000 scenario (version 4.3). Rapport technique, Eurocontrol, Avril 1989.

[40] Eurocontrol Experimental Centre : Phare highly interactive problem solver. Rapport technique, Eurocontrol, Novembre 1994.

[41] S. J. Flockton et M. S. White : Pole-zero system identification using genetic algorithms. *In Proceedings of the Fifth International Conference on Genetic Algorithm*. ICGA, 1993.

[42] D. B. Fogel : *Evolutionary Computation. Toward a new Philosophy of Machine Intelligence*. IEEE press, 1994.

[43] S. Forrest, R.E. Smith, B. Jakornik et A.S. Perelson : Using genetic algorithms to explore pattern recognition in the immune system. *Evolutionary Computation*, 1(3):191–211, 1993.

[44] M. I. Freidlin et A. D. Wentzell : *Random Perturbations of Dynamical Systems*. Springer-Verlag, New-York, 1983.

[45] T. L. Friesz et D. Bernstein : A variational inequality formulation of the dynamic network user equilibrium problem. *Operations Research*, 41(1):179–191, 1993.

[46] T. L. Friesz et J. Luque : Dynamic network traffic assignment considered as a continuous time optimal control problem. *Operations Research*, 37(6):893–901, 1989.

[47] G. Gawinowski, J. L. Garcia, E. Hollnagel et R. Weber : Erasmus-state of the art. Rapport technique, EUROCONTROL, 2007.

[48] G. Gerla : Fuzzy logic programming and fuzzy control. *Studia Logica,*, 79(2):231–254, mars 2005.

[49] D. C. Giancoli : *Ondes, optique et physique moderne*. DeBoeck, 1993.

[50] J. Gjerlev : *Instrument flight procedures including RNAV and FMS operations*. Norsk aero forlag, 2002.

[51] D. E. Goldberg : *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading MA Addison Wesley, 1989.

[52] G. Granger, N. Durand et J. M. Alliot : Token allocation strategy for free-flight conflict solving. *In IJCAI'01*, 2001.

[53] M. P. Helme : Reducing air traffic delay in a space-time network. *In IEEE International Conference on Systems, Man and Cybernetics*, page 236–242, Chicago, 1992.

[54] J. H. Holland : *Adaptation in Natural and Artificial Systems*. University of Michigan press, 1975.

[55] A. Homaifar, G. Shanguchuan et G. A. Liepins : A new approach on the traveling salesman problem by genetic algorithms. *In Proceedings of the Fifth International Conference on Genetic Algorithm*. ICGA, 1993.

[56] J. Hu, M. Prandini, A. Nilim et S. Sastry : Optimal coordinated maneuvres for three-dimensional aircraft conflict resolution. *Journal of Guidance, Control, and Dynamics*, 25(5):888–900, 2002.

[57] C. Huygens : *Traité de la lumière*. Gauthier-Villars, éditeurs. Libraires du bureau des longitudes, de l'École polytechnique, 1690.

[58] ILOG : *ILOG Cplex User's Guide*, 1999.

[59] L. Ingber et B. Rosen : Genetic algorithms and very fast simulated re-annealing. *Mathematical Computer Modeling*, 16(11):87–100, 1992.

[60] P. Jackson : *Introduction to Expert Systems*. Addison Wesley, 1998.

[61] R. E. Kalman : A new approach to linear filtering and prediction problems. *Transactions of the ASME - Journal of Basic Engineering*, 82:35–45, mars 1960.

[62] S. Khuri, T. Back et J. Heitkoter : The zero/one multiple knapsack problem and genetic algorithms. *In Proceedings of the Symposium of Applied Computation*. ACM, 1994.

[63] R. Kimmel et J. Sethian : Computing geodesic paths on manifolds. *In Proceeding of Natianal Academy of Sciences of USA*, pages 8431–8435, 1998.

[64] D. E. Koditschek et E. Rimon : Robot navigation functions on manifolds with boundary. *Advances in Applied Mathematics*, 11:412–442, 1990.

[65] J. R. Koza : *Genetic Programming*. MIT press, 1992.

[66] J. Krozel, M. E. Peters et G. Hunter : Conflict detection and resolution for future air transportation management. Rapport technique, NASA Ames Research Center Moffett Field, CA 94035, 1997.

[67] J. K. Kuchar et L. C. Yang : A review of conflict detection and resolution modeling methods. *IEEE Transactions on Intelligent Transportation Systems*, 1, 1992.

[68] E. L. Lawler et D. E. Wood : Branch-and-bound methods : A survey. *Operations Research*, 14(4):699–719, juillet 1966.

[69] K. Lindsay, E. Boyd et R. Burlingame : Traffic flow management modeling with the time assignment model. *Air Traffic Control Quarterly*, 1(3):255–276, 1993.

[70] S. Loizou et K. Kyriakopoulos : Closed loop navigation for multiple non-holonomic vehicles. *In IEEE International Conference on Robotics and Automation*, page 4240–4245, 2003.

[71] G. Maignan : *Le Contrôle de la circulation aérienne*, volume 2572. Presse Universitaire de France, 1991.

[72] D. Maupou : Modélisation du nombre de conflits potentiels à partir d'un simulateur de trafic. Mémoire de D.E.A., Université Paul Sabatier, Toulouse, France, 2000.

[73] S. Mehrotra : On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2(4):575–601, novembre 1992.

[74] N. Metropolis et S. Ulam : The Monte Carlo method. *The American Statistical Association*, 44(247):335–341, septembre 1949.

[75] J. S. B. Mitchell, D. M. Mount et C. H. Papadimitriou : The discrete geodesic problem. *SIAM J. Computing*, 16:647–668, août 1987.

[76] F. Mora-Camino : *Système de conduite automatique et de gestion de vol*. École nationale de l'aviation civile, 2 édition, avril 1995.

[77] F. Médioni : *Méthodes d'optimisation pour l'évitement aérien : systèmes centralisés, systèmes embarqués*. Thèse de doctorat, École Polytechnique, Palaiseau, 1998.

[78] F. Médioni, N. Durand et J. M. Alliot : Algorithmes génétiques et programmation linéaire appliqués à la résolution de conflits aériens. *In Journées Évolution Artificielle Francophones*, 1994.

[79] M. Novotni et R. Klein : Computing geodesic distances on triangular meshs. *In Proceeding of the 10th International Conference in Central Europe on Computer Graphics Visualization and Computer Vision*, pages 341–347, Pilsen, 2002.

[80] A. Nuic : User manual for the Base of Aircraft DAta (BADA) revision 3.7. Rapport technique No. 2009-003, Eurocontrol, mars 2009.

[81] U.S. Department of Transportation Federal Aviation Administration : Introduction to TCAS II version 7, 2000.

[82] X. Olive : Résolution de conflit par algorithmes stochastiques parallèles. Mémoire de D.E.A., École Nationale Supérieure de l'Aéronautique et de l'Espace, Toulouse, 2006.

[83] S. Oussedik et D. Delahaye : Dynamic air traffic planning by genetic algorithms. *In Congress on Evolutionary Computation*, Washington D.C, Juillet 1999.

[84] L. Pallottino, E. Feron et A. Bicchi : Conflict resolution problems for air traffic management systems solved with mixed integer programming. *IEEE Transactions on Intelligent Transportation Systems*, 3(1):3–11, 2002.

[85] J. Pearl : *Heuristics : Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley, 1984.

[86] T.S. Perry et J.A. Adam : Improving the world's largest, most advanced system ! *IEEE Spectrum*, 28(2):22–36, Février 1991.

[87] W. Philipp et F. Gainche : Air traffic flow management in Europe. *In Advanced Technologies for Air Traffic Flow Management*, page 64–106. Springer-Verlag, 1994.

[88] Joint Planning et Development Office : *Concept of Operations for the Next Generation Air Transport System*, juin 2007.

[89] A. Podelski : *Constraint programming : basics and trends*. Châtillon Spring School, 1994.

[90] A. U. Raghunathan, V. Gopal, D. Subramanian, L. T. Biegler et T. Samad : Dynamic optimization strategies for three-dimensional conflict resolution of multiple aircraft. *Journal of guidance, control, and dynamics*, 27(4):586–594, 2004.

[91] E. Rimon et D. Koditschek : Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation*, 8(5):501–508, 1992.

[92] G. Roussos, G. Chaloulos, K. Kyriakopoulos et J. Lygeros : Control of multiple non-holonomic air vehicles under wind uncertainty using model predictive control and decentralized navigation function. *In IEEE Conference on Decision and Control*, décembre 2008.

[93] C. Le Roux : *Système de conduite du vol*. École Nationale de l'Aviation Civile, 3 édition, 2004.

[94] J. D. Schaffer et J. E. Eshelman : Designing multiplierless digital filter using genetic algorithms. *In Proceedings of the Fifth International Conference on Genetic Algorithm*. ICGA, 1993.

[95] M. Schoenauer, E. Ronald et S. Damour : Evolving nets for control. *In Proceeding of the Sixth International Conference on Neural Networks and their Industrial and Cognitive Applications*. AFIA, 1993.

[96] H. P. Schwefel : *Evolution and Optimum Seeking*. Wiley, New York, 1995.

[97] SESAR consortium : The ATM Target Concept D3. Rapport technique DLM-0612-001-02-00a, EUROCONTROL, Toulouse, séptembre 2007.

[98] SESAR CONSORTIUM : The ATM Deployment Sequence D4. Rapport technique DLM-0706-001-02-00, EUROCONTROL, Frankfurt, janvier 2008.

[99] SESAR CONSORTIUM : SESAR Master Plan. Rapport technique DLM-0710-001-02-00, EUROCONTROL, Bruxelles, avril 2008.

[100] J. A. SETHIAN : A fast marching level set method for monotonically advancing fronts. *In Proceedings of the National Academy of Sciences*, pages 1591–1595, 1995.

[101] Comité spéciale 186 : Minimum aviation system performance standards for the automatic surveillance-broadcast (ADS-B). Rapport technique DO-242A, RTCA Inc., juin 2002.

[102] V. SURAZHSKY, T. SURAZHSKY, D. KIRSANOV, S. J. GORTLER et H. HOPPE : Fast exact and approximate geodesics on meshes. *ACM Transactions on Graphics*, 24(3):553–560, 2005.

[103] J. W. TANG, F. ZHANG et M. ZHANG : Fast approximate geodesic paths on triangle mesh. *In Proceeding of the International Journal of Automation and Computing*, pages 8–13, 2007.

[104] E. TRELAT : *Contrôle optimal : théorie et applications*. Vuibert, 2005.

[105] P.L. TUAN, H.S. PROCTER et G.J. COULURIS : Advanced productivity analysis methods for air traffic control operation. Rapport technique, Stanford Research Institute, Menlo Park CA, décembre 1976.

[106] R. UNGER et J. MOULT : A Genetic Algorithm for 3D Protein Folding Simulations. *In Proceedings of the Fifth International Conference on Genetic Algorithms*. ICGA, 1993.

[107] S. VAJDA : *Théorie des jeux et programmation linéaire*. Dunod, 1959.

[108] J. VILLIERS : Contribution à une théorie du système de contrôle en route et ses perspectives d'évolution. Rapport technique, Direction Génerale de l'Aviation Civile (DGAC), Paris, août 1984.

[109] H. WATABE et N. OKINO : A study on genetic shape design. *In Proceedings of the Fifth International Conference on Genetic Algorithm*. ICGA, 1993.

[110] S. J. WRIGHT : *Primal-Dual Interior-Point Methods*. Society for Industrial Mathematics, 1987.

[111] X. YIN et N. GERMAY : Investigations on solving the load flow problem by genetic algorithms. *Electric Power Systems Research*, 22:151–163, 1991.

[112] R. ZANNI : A methodology for airspace organization. Rapport technique, Centre d'étude de la navigation aérienne (CENA), Toulouse, avril 1994.

[113] K. ZEGHAL : Champs de forces symétriques : La logique d'un système anticollision coordonné. Rapport technique, ONERA, Toulouse, 1993.

[114] K. ZEGHAL : *Vers une théorie de la coordination d'actions, application à la navigation aérienne*. Thèse de doctorat, Université Paris VI, 1994.

[115] K. Zeghal : A comparison of different approaches based on forces fields for coordination among multiple mobiles. *In IEEE International Conference on Intelligent Robotic Systems (IROS), Victoria*, octobre 1998.

[116] K. Zeghal : A review of different approaches based on force fields for airborne conflict resolution. *In AIAA Guidance, Navigation and Control Conference*, août 1998.

[117] W. Zhang : Depth-first branch-and-bound versus local search : A case study. *In Proceeding 17th National Conference on Artificial Intelligence, Austin*, pages 930–935, 2000.

Abstract

In the European framework SESAR, the need to increase the air traffic capacity motivated the 4D (space + time) aircraft trajectory planning.

In order to produce, on one side, a pre-tactical trajectory planning (an aircraft avoid congested areas or areas with bad weather) and on the other side a tactical trajectory planning (generating sets of 4D conflict-free trajectories), we introduce a new algorithm : the Light Propagation Algorithm (LPA). This algorithm is based on a wavefront propagation method inspired from light propagation analogy. Besides, this algorithm is adapted to the trajectory planning problem.

LPA produces very good results for a real traffic day over France, while satisfying specific ATM constraints.

LPA was then adapted to take into account the uncertainties concerning actual aircraft positions. Once adapted to the uncertainty LPA was tested on the same traffic day, using RTA points (Real Arrival Time). When LPA fails to resolve conflicts, RTA points are used to reduce the uncertainty concerning aircraft positions. The results obtained are very encouraging.