Rapport de projet

Impacts EGTS à Roissy-CDG

DUBOIS Auriane

CAILLET Adrien

ROBERT Baptiste IENAC17

BUIL Morgane SITA

Table des matières

[I/ Introduction et présentation du projet 3](#_Toc532309614)

[II/ Calcul de l'altitude en tout point de l'aéroport 4](#_Toc532309615)

[1/ Algorithme de triangulation de Delaunay 4](#_Toc532309616)

[2/ Calcul du plan d'un triangle et test d'appartenance à un triangle 4](#_Toc532309617)

[3/ Calcul des pentes sur chaque portion d'une trajectoire 5](#_Toc532309618)

[III/ Implémentation du modèle d'accélération 5](#_Toc532309619)

[1/ Calcul du modèle d’accélération 5](#_Toc532309620)

[2/ Calcul des nouvelles trajectoires en fonction du modèle d'accélération 6](#_Toc532309621)

[3/ Mesure et comparaison des temps de roulage 6](#_Toc532309622)

[IV/ Résolution des conflits 6](#_Toc532309623)

[1/ Algorithme de backtrack 6](#_Toc532309624)

[2/ Mesure des temps de roulage et des retards 6](#_Toc532309625)

[V/ Simulation 6](#_Toc532309626)

[VII/ Bibliographie 7](#_Toc532309627)

Table des matières

I/ Introduction et présentation du projet

II/ Calcul de l'altitude en tout point de l'aéroport

1/ Algorithme de triangulation de Delaunay

2/ Calcul du plan d'un triangle et test d'appartenance d'un point à un triangle

3/ Calcul des pentes sur chaque portion d'une trajectoire

III/ Implémentation du modèle d'accélération

1/ Calcul du modèle d’accélération

2/ Calcul des nouvelles trajectoires en fonction du modèle d'accélération

3/ Mesure et comparaison des temps de roulage

IV/ Résolution des conflits

1/ Algorithme de backtrack

2/ Mesure des temps de roulage et des retards

V/ Simulation

VI/ Conclusion

VII/ Bibliographie

VIII/ Annexes

# I/ Introduction et présentation du projet

Le système EGTS (Electric Green Taxiing System) est un système électrique implanté sur le train d’atterrissage principal d’avions, tels que l’Airbus A320 et ses dérivés, et qui permet à ceux-ci de circuler depuis leurs parkings jusqu’à la piste sans utiliser leurs moteurs principaux. Son but est de réduire la quantité de carburant utilisé par un avion, ainsi que de diminuer la quantité de gaz à effet de serre émis lors des opérations au sol.

Il présente de nombreux avantages, tel que la réduction des coûts dus à l’utilisation des réacteurs, car ceux-ci sont très consommateurs en carburant et peu efficaces au sol. Ce système étant 100% électrique, il est donc aussi respectueux de l’environnement, ce qui est un enjeu crucial dans le contexte mondial actuel. De plus ce système permettrait de ne plus utiliser de camions pour sortir les avions de leur emplacement.

L’EGTS a été conçu en 2003 par l’entreprise Delos Aerospace et est actuellement fortement développé par le groupe SAFRAN.

Cependant, ce système est très sensible aux pentes rencontrées sur le terrain aéroportuaire, contrairement aux réacteurs. Si le terrain est plat, il sera tout aussi efficace. Mais, la puissance du moteur électrique ne lui permet pas de maintenir une vitesse constante si la pente est trop importante. Ainsi, son utilisation peut fortement ralentir la circulation des avions au roulage. La réduction du trafic et les pertes d’argent qui en découleraient effraient les aéroports qui refusent donc l’utilisation de ce système.

L’objectif de notre projet est, donc, d’étudier l’impact du système EGTS sur l’aéroport de Roissy – Charles De Gaulle, pour ainsi déterminer si ce système est viable sur cet aéroport.

Pour se faire, nous allons tout d’abord déterminer l’altitude en tout point de l’aéroport, grâce à un algorithme de triangulation de Delaunay et au calcul des équations des plans de chacun des triangles obtenus. Puis nous implémenterons un système d’accélération qui nous permettra de déterminer les trajectoires des avions équipés de cette technologie. Enfin, nous utiliserons un algorithme de backtrack pour résoudre les conflits qui pourraient apparaître lors d’une journée « normale », si certains avions sont équipés du système EGTS.

# II/ Calcul de l'altitude en tout point de l'aéroport

## 1/ Algorithme de triangulation de Delaunay

Pour calculer l’altitude de tous les points de l’aéroport, nous avons commencé par séparer le terrain en différentes « zones », grâce à un algorithme de triangulation de Delaunay.

La triangulation de Delaunay est une méthode géométrique, inventée en 1934 par le mathématicien russe Boris Delaunay, qui permet de séparer un plan en triangles, de sorte qu’aucun point du plan n’est à l’intérieur du cercle circonscrit d’un des triangles de la triangulation. Cela permet de maximiser le plus petit angle de l'ensemble des angles des triangles, évitant ainsi les triangles « allongés ».

Pour effectuer cette méthode, nous avons choisi de mettre en œuvre l’algorithme de Bowyer-Watson, un algorithme incrémental qui implémente la triangulation de Delaunay. Son principe est d’ajouter des points un par un à une triangulation de Delaunay valide d’une sous-liste de points. Après chaque insertion, les triangles dont le cercle circonscrit contient le point ajouté sont supprimés, laissant alors un polygone, entourant le nouveau point, qui est alors triangulé à nouveau en utilisant ce point pour former des triangles avec les différentes arêtes du polygone.

Pour écrire cet algorithme, nous avons trouvé une version en pseudo-code, que nous avons traduite en OCaml. Pour simplifier le code, nous avons écrit cinq sous-fonctions :

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* Une qui calcule le centre et le rayon d’un cercle circonscrit à un triangle
* Une qui teste si un point est dans un cercle
* Une //edge//
* Une qui supprime un triangle d’une liste de triangles
* Une //removePointCommunSuperTri//

🡺 Pas de listing 🡪 donc je vire ?? oui

Il nous a été fourni une liste de points, correspondant à l’aéroport de Roissy - Charles de Gaulle, dont nous connaissons les coordonnées et à partir de laquelle nous avons effectué la triangulation.

Pour cela nous avons dû définir un type triangle qui contient les trois sommets du triangle et le tuple des coefficients de l’équation du plan de celui-ci.

## 2/ Calcul du plan d'un triangle et test d'appartenance à un triangle

Maintenant que l’aéroport est divisé en triangles, nous allons pouvoir calculer l’équation de plan de chacun.

Pour ce faire, nous avons récupéré un algorithme implémentant la méthode d’élimination de Gauss.

Ainsi, pour chaque triangle, à partir des coordonnées de ses 3 sommets, nous avons pu déterminer les coefficients a, b et c de l’équation de plan associée : z = a\*x +b\*y + c en résolvant un système 3 équations à 3 inconnues.

// Détailler module solve //

Puis nous avons créé un test, pour vérifier si un point donné appartient ou non à un triangle donné. En effet, cela sera utile par la suite pour déterminer quelle équation de plan choisir lors du calcul de l’altitude de ce point et pouvoir calculer des pentes entre 2 points qui devront nécessairement appartenir au même triangle.

// Détailler **geometrieTriangle.ml** //

## 3/ Calcul des pentes sur chaque portion d'une trajectoire

Il nous a été fourni une liste des avions circulant sur l’aéroport Roissy – Charles de Gaulle lors d’une journée type. Ces avions sont notamment décrits par un ensemble de coordonnées : ces coordonnées sont celles des points formant la trajectoire de chaque avion.

Nous avons précédemment réussi à déterminer l’équation de tous les triangles formant le secteur de l’aéroport. Il nous est donc maintenant possible de déterminer l’altitude de tous les points de l’espace à l’aide de ces équations.

Pour un point donné d’une trajectoire, s’il est prouvé comme appartenant à un triangle T (grâce au test décrit dans le point II/ 2/) d’équation z = a\*x + b\*y + c, alors il vérifie lui-même l’équation et il est donc aisé de déterminer sa coordonnée verticale en utilisant les coefficients a, b et c du triangle T et ses coordonnées x et y.

Lorsque les z de tous les points de toutes les trajectoires ont été déterminées, on peut alors déterminer les pentes tout au long d’une trajectoire. Si 2 points appartiennent à un même triangle, il suffit de diviser la différence d’altitude entre les points par la distance entre ces 2 points pour obtenir un pourcentage de pente. Dans le cas où les 2 points n’appartiendraient pas au même triangle, il faudra trouver le ou les points d’intersection et calculer les pentes résultantes entre chaque point. En effet, le but est que la nouvelle trajectoire de l’avion colle au mieux à la modélisation 3d de l’aéroport. // pente.ml ?? Détermine la distance entre 2 points d’une trajectoire, mais pas la pente. Manque une fonction pente() qui calcule la pente = distance3D / deltaZ) //.

# III/ Implémentation du modèle d'accélération

## 1/ Calcul du modèle d’accélération

Notre but étant de déterminer les éventuels retards que pourraient causer le système EGTS, il nous faut donc calculer les vitesses des avions avec un tel système et les comparer au système classique avec l’utilisation des réacteurs.

Nous avons donc calculé le modèle d’accélération, pour déterminer la vitesse 5 secondes plus tard à partir de la vitesse courante.

Pour le système classique, le modèle est simple : la vitesse augmente de 0.9 (0.9 quoi ???) toutes les 5 secondes.

Pour le système EGTS, le modèle est plus complexe : la vitesse dans 5 secondes (V\_5) dépend de la vitesse courante (V), mais aussi de la masse (M) de l’avion et de la pente du terrain et bien entendu des capacités du moteur électriques.

// Détail du modèle d’accélération //

## 2/ Calcul des nouvelles trajectoires en fonction du modèle d'accélération

Nous sommes donc à un stade où nous possédons des points, avec altitude, qui forment une trajectoire et un modèle 3d de notre aéroport. Il faut donc adapter notre trajectoire pour qu’elle suive au mieux les différents plans, définis par des triangles, de notre aéroport. C’est pourquoi, il a fallu vérifier que 2 points consécutifs appartenaient au même triangle. En effet, le but est que la nouvelle trajectoire de l’avion colle au mieux à la modélisation 3d de l’aéroport et cela passe par suivre les différents plans de notre modélisation. Ainsi, en calculant des points d’intersection entre les différents triangles et nos segments composant la trajectoire d’un avion, on peut récupérer une liste de points conformes à nos attentes. Cette liste constitue notre nouvelle trajectoire qui est sensible à la modélisation 3d de notre aéroport.

Ensuite, à partir de cette liste, il est facile d’appliquer notre modèle d’accélération entre chaque point : nous pouvons calculer facilement les différentes pentes associées et donc les vitesses.

## 3/ Mesure et comparaison des temps de roulage

# IV/ Résolution des conflits

## 1/ Algorithme de backtrack

## 2/ Mesure des temps de roulage et des retards

# V/ Simulation

En nous inspirant de l’application Pyairport, vue en TP l’année dernière, nous avons

# VII/ Bibliographie

[*https://en.wikipedia.org/wiki/EGTS*](https://en.wikipedia.org/wiki/EGTS)

[*https://fr.wikipedia.org/wiki/Triangulation\_de\_Delaunay*](https://fr.wikipedia.org/wiki/Triangulation_de_Delaunay)

[*https://en.wikipedia.org/wiki/Bowyer%E2%80%93Watson\_algorithm*](https://en.wikipedia.org/wiki/Bowyer–Watson_algorithm)

<https://rosettacode.org/wiki/Gaussian_elimination>