# Flight Tracker Pro

## Dashboard de Surveillance du Trafic Aérien

Rapport de Projet - UE OpenData

Aness RABIA Georgios STEPHANOU

Année Universitaire 2024-2025

**Résumé**

Ce rapport présente Flight Tracker Pro, un dashboard web de visualisation du trafic aérien mondial en temps réel exploitant les données ouvertes de l'API OpenSky Network. Le projet démontre l'utilité des données ouvertes pour la recherche et l'analyse environnementale, en permettant de suivre plus de 10 000 vols simultanés et d'analyser leur impact écologique.

# 1 Introduction

## 1.1 Contexte

Le trafic aérien génère quotidiennement des millions de données de vol. OpenSky Network, un réseau collaboratif de plus de 20 000 récepteurs ADS-B à travers le monde, collecte et rend publiques ces données en temps réel. Ces données ouvertes permettent des analyses approfondies sur le trafic aérien, la congestion, et surtout l'impact environnemental du secteur aérien.

## 1.2 Problématique

Comment exploiter les données ouvertes du trafic aérien pour créer un outil d'analyse accessible permettant de visualiser, comprendre et évaluer l'impact environnemental de l'aviation mondiale ?

## 1.3 Objectifs

— Explorer l'API OpenSky Network (open data)
— Développer un dashboard web interactif
— Analyser la densité du trafic aérien mondial
— Évaluer les possibilités d'analyse environnementale (pollution, émissions CO)
— Créer une solution gratuite et open-source

## 1.4 Méthodologie

**Durée :** 4 semaines (novembre 2024)
**Équipe :** 2 personnes (Aness RABIA, Giorgios STEPHANOS)
**Technologies :** Python 3.13, Streamlit, Plotly, Pandas, API OpenSky Network

# 2 OpenSky Network : Open Data pour la Recherche

## 2.1 Présentation

OpenSky Network est une initiative à but non lucratif qui met à disposition gratuitement des données de vol en temps réel. Cette ressource constitue une opportunité unique pour la recherche académique, permettant d'étudier le trafic aérien sans coût d'infrastructure.

## 2.2 API REST

L'API OpenSky offre plusieurs endpoints :

| Endpoint | Description |
|---|---|
| `/states/all` | État actuel de tous les vols détectés |
| `/tracks/all` | Trajectoire complète d'un vol spécifique |
| `/flights/all` | Historique des vols sur une période |

## 2.3 Exemple de requête

```python
import requests

# Récupérer tous les vols dans une zone (Europe)
response = requests.get(
    "https://opensky-network.org/api/states/all",
    params={
        'lamin': 40.0,  # Latitude min
        'lomin': -10.0, # Longitude min
        'lamax': 55.0,  # Latitude max
        'lomax': 30.0   # Longitude max
    }
)

data = response.json()
print(f"Vols détectés : {len(data['states'])}")
# Résultat : ~3000 vols en Europe
```

## 2.4 Exemple de données retournées

Chaque vol est représenté par 17 champs :

```
{
  "time": 1701432000,
  "states": [
    [
      "3c6444",           // icao24 (ID unique avion)
      "BAW123",           // callsign (numéro de vol)
      "United Kingdom",   // origin_country
      -0.4619,            // longitude
      51.4706,            // latitude
      10668,              // altitude barométrique (m)
      false,              // on_ground (au sol ?)
      250.5,              // velocity (vitesse m/s)
      180.5,              // true_track (direction degrés)
      5.2                 // vertical_rate (taux montée m/s)
    ]
  ]
}
```

**Observation :** Ces données brutes permettent des analyses variées : statistiques de vol, densité de trafic, corrélations, et estimations d'émissions polluantes.

# 3 Développement du Projet : Avancement Hebdomadaire

## 3.1 Semaine 1 : Exploration et Configuration

**Objectif :** Comprendre l'API OpenSky et mettre en place l'environnement de développement.

**Réalisations :**

— Exploration de la documentation OpenSky Network

— Tests de requêtes API avec différentes zones géographiques

— Configuration de l'authentification OAuth2

— Installation de l'environnement Python (Streamlit, Plotly, Pandas)

— Premiers tests de récupération de données

**Résultats :**

— Compréhension de la structure des données (17 champs par vol)

— Configuration réussie avec limites de requêtes élevées (OAuth2)

— Détection moyenne de 10 000+ vols sur zone mondiale

## 3.2 Semaine 2 : Traitement des Données et Première Carte

**Objectif :** Nettoyer les données et créer une première visualisation cartographique.

**Réalisations :**

— Développement du pipeline de traitement des données

— Gestion des valeurs manquantes et données invalides

— Conversion des unités (m/s → knots, mètres → feet)

— Calcul du statut de vol (montée/descente/croisière/sol)

— Création de la carte interactive avec Plotly

— Reconnaissance automatique des compagnies aériennes (24+ compagnies)

**Résultats :**

— Pipeline fonctionnel avec nettoyage automatique

— Première carte interactive avec 4 couleurs par statut

— Identification de compagnies : Air France, Lufthansa, Emirates, etc.

## 3.3 Semaine 3 : Analyses Statistiques et Optimisations

**Objectif :** Implémenter les analyses statistiques et améliorer les performances.

**Réalisations :**

— Création de 7 onglets d'analyse (Carte, Analyses, Heatmap, Compagnies, Liste, Détails, Statistiques)

— Développement de la heatmap de densité du trafic

— Analyses statistiques : distributions altitude/vitesse, top pays, corrélations

— Optimisation de la carte : migration Scattergeo → Scattermapbox ($\times$ 10 plus rapide)

— Implémentation de filtres (région, altitude, vitesse, pays, compagnie)

**Résultats :**

— Application fluide avec 10 000+ vols (60 FPS)

— Heatmap avec grilles de 2°×2° (zones de  200 km)

— Identification des zones les plus denses (Amsterdam, Paris, Londres)

### 3.4 Semaine 4 : Design, Tests et Documentation

**Objectif :** Finaliser le design, corriger les bugs et documenter le projet.
**Réalisations :**

— Design moderne avec CSS personnalisé (bleu/blanc minimaliste)

— Corrections de bugs (filtres, conflits de variables, timeout API)

— Amélioration de l'ergonomie (légende visible, explications claires)

— Tests de performance et validation fonctionnelle

— Documentation complète du code

— Préparation de la présentation

**Résultats :**

— Application complète avec 7 onglets fonctionnels

— Design professionnel et responsive

— Performance : rafraîchissement $<$ 3 secondes

— Export CSV pour analyses externes

## 4 Fonctionnalités Principales

### 4.1 Vue d'ensemble

L'application propose 7 onglets d'analyse :

1. **Carte Interactive** : Visualisation mondiale en temps réel (10 000+ vols)
2. **Analyses** : Distributions statistiques (altitude, vitesse, pays)
3. **Heatmap** : Densité du trafic aérien par zones géographiques
4. **Compagnies** : Classement et analyse par compagnie aérienne
5. **Liste des Vols** : Tableau interactif avec export CSV
6. **Détails Vol** : Recherche d'un vol spécifique
7. **Statistiques Globales** : Métriques agrégées et records

### 4.2 Carte Interactive

Visualisation temps réel avec 4 couleurs par statut :

— Rouge : Montée

— Noir : Descente

— Bleu : Croisière

— Gris : Au sol

### 4.3 Heatmap de Densité

La heatmap divise le globe en grilles de 2°×2° (environ 200 km × 200 km) et calcule le nombre de vols par zone. Cela permet d'identifier les couloirs aériens les plus fréquentés et d'estimer les zones à forte pollution atmosphérique.

### 4.4 Filtres Avancés

— **Région** : 8 zones prédéfinies (Monde, Europe, USA, Asie, etc.)
— **Altitude** : Min/Max (0-50 000 ft)
— **Vitesse** : Minimum (0-600 knots)
— **Pays** : Filtrage par pays d'origine
— **Compagnie** : Sélection d'une compagnie spécifique

## 5 Difficultés Rencontrées et Solutions

### 5.1 Difficulté 1 : Performance de la Carte

**Problème :** Avec Plotly Scattergeo, la carte était très lente ($< 5$ FPS) lors de l'affichage de milliers de vols.
**Cause :** Pas d'accélération matérielle GPU, rendu uniquement CPU.
**Solution :** Migration vers Plotly Scattermapbox qui utilise Mapbox GL avec hardware acceleration → 60 FPS.

### 5.2 Difficulté 2 : Timeout de l'API OAuth

**Problème :** Erreurs de connexion fréquentes au serveur d'authentification OpenSky.
**Cause :** Timeout par défaut trop court (10 secondes).
**Solution :**

— Augmentation du timeout à 30 secondes
— Implémentation d'un mode anonyme de secours
— Message d'avertissement à l'utilisateur en mode dégradé

### 5.3 Difficulté 3 : Conflits de Variables Streamlit

**Problème :** Le filtre par compagnie dans la sidebar ne fonctionnait pas correctement.
**Cause :** Deux `st.selectbox` partageaient la même variable (`selected_airline`).
**Solution :**

— Renommage des variables avec noms uniques
— Ajout de paramètre `key` unique pour chaque widget Streamlit

### 5.4 Difficulté 4 : Compréhension de la Heatmap

**Problème :** Les utilisateurs ne comprenaient pas ce que représentait la heatmap.
**Cause :** Manque d'explications contextuelles.
**Solution :**

— Réduction de la taille des grilles (5°×5° → 2°×2°) pour plus de précision
— Ajout de légendes explicatives
— Identification géographique des zones denses (Europe de l'Ouest, etc.)
— Calcul de dispersion pour quantifier la concentration

# 6 Impact Environnemental et Pollution

## 6.1 Problématique Environnementale

L'aviation représente environ **2-3% des émissions mondiales de CO**, avec une croissance annuelle de 5% avant la pandémie. Les données ouvertes comme celles d'OpenSky permettent d'analyser ce phénomène.

## 6.2 Données Exploitables pour l'Analyse Environnementale

Avec OpenSky Network, il est possible d'estimer :

1. **Densité de trafic par zone** → Zones à forte pollution atmosphérique
2. **Nombre de vols par pays** → Contribution nationale aux émissions
3. **Altitudes de croisière** → Efficacité énergétique
4. **Vitesses moyennes** → Consommation de carburant
5. **Distances parcourues** → Émissions totales estimées

## 6.3 Exemple d'Analyse : Émissions CO

**Formule simplifiée :**
Un avion commercial moyen émet environ **3 kg de CO par km parcouru**.

```
# Calcul d'émissions pour un vol de 1000 km
distance_km = 1000
emission_factor = 3  # kg CO per km
total_co2 = distance_km * emission_factor

print(f"Émissions estimées : {total_co2} kg CO")
# Résultat : 3000 kg CO (3 tonnes)
```

**Application à notre heatmap :**
Les zones les plus denses de notre heatmap (Amsterdam, Paris, Londres, Francfort) sont aussi les zones à plus forte concentration de pollution aérienne. Un vol dans ces couloirs aériens contribue significativement aux émissions locales de CO, NOx et particules fines.

## 6.4 Analyse des Compagnies

Notre outil révèle que certaines compagnies volent plus haut et plus vite :

| **Compagnie** | **Alt. moy. (ft)** | **Vit. moy. (kts)** |
|---|---|---|
| Emirates (long-courrier) | 38 500 | 485 |
| Air France | 36 500 | 470 |
| Ryanair (low-cost) | 32 100 | 445 |

**Interprétation environnementale :**

— Les vols long-courriers (Emirates) volent plus haut → plus efficaces énergétiquement

— Les low-cost (Ryanair) volent plus bas → consommation légèrement supérieure

— Corrélation altitude-vitesse = 0.72 → voler haut permet de voler vite avec moins de résistance

## 6.5 Perspectives d'Amélioration Environnementale

Avec les données OpenSky, il serait possible d'approfondir :

1. **Estimation précise des émissions** : Utiliser les modèles BADA (Base of Aircraft Data) pour calculer consommation et émissions réelles
2. **Optimisation des routes** : Identifier les routes inefficaces et suggérer des alternatives
3. **Analyse temporelle** : Comparer le trafic pré/post-pandémie pour quantifier l'impact environnemental
4. **Prédictions** : Machine Learning pour anticiper les pics de pollution aérienne
5. **Comparaisons modales** : Comparer l'empreinte carbone avion vs train pour certains trajets

# 7 Résultats et Validation

## 7.1 Métriques de Performance

| **Métrique** | **Résultat** |
|---|---|
| Vols simultanés affichés | 10 000+ |
| Temps de rafraîchissement | $<$ 3 secondes |
| FPS carte interactive | 60 |
| Compagnies reconnues | 24+ |
| Zones heatmap | 16 200 (90×180 grilles) |

## 7.2 Fonctionnalités Livrées

7 onglets d'analyse fonctionnels

Carte interactive temps réel

5 types de filtres avancés

Export CSV pour analyses externes

Heatmap de densité

Analyses statistiques complètes

Design moderne et responsive

### 7.3 Limites Identifiées

— Données temps réel uniquement (pas d'historique intégré)

— Reconnaissance limitée à 24 compagnies principales

— Pas de calcul automatique d'émissions CO

— Dépendance à la connexion Internet

## 8 Améliorations Possibles

### 8.1 Améliorations Techniques

1. **Base de données** : PostgreSQL pour stocker l'historique des vols
2. **Trajectoires** : Visualisation des routes aériennes empruntées
3. **API propre** : Développer une REST API pour chercheurs
4. **Mode dark** : Thème sombre pour réduire fatigue oculaire

### 8.2 Améliorations Environnementales

1. **Module Émissions CO** : Calcul automatique basé sur distance, type d'avion, altitude
2. **Comparateur Modal** : Comparer empreinte carbone avion vs train
3. **Alertes Pollution** : Notifications zones à forte concentration
4. **Analyse Temporelle** : Évolution du trafic et des émissions dans le temps
5. **Machine Learning** : Prédiction de trafic et optimisation de routes

### 8.3 Exemple d'Amélioration : Calculateur d'Émissions

Un module pourrait estimer les émissions d'un vol en combinant :

— Distance calculée via coordonnées GPS

— Type d'avion (via base ICAO24)

— Consommation moyenne par type (base BADA)

— Phase de vol (décollage/croisière/atterrissage)

Cela permettrait de créer un **dashboard environnemental** montrant :

— Émissions totales mondiales en temps réel

— Classement des pays/compagnies par empreinte carbone

— Évolution journalière/mensuelle

# 9 Conclusion

## 9.1 Synthèse

Ce projet a démontré l'intérêt des données ouvertes pour la recherche et l'analyse du trafic aérien. En 4 semaines, nous avons développé Flight Tracker Pro, un dashboard web complet permettant de visualiser plus de 10 000 vols simultanément avec des analyses statistiques avancées.

## 9.2 Apport Pédagogique

Ce projet illustre concrètement :

— L'exploitation d'APIs REST publiques (open data)
— Le traitement de données massives en temps réel
— L'importance des visualisations interactives
— Les possibilités d'analyse environnementale avec données ouvertes

## 9.3 Perspectives

OpenSky Network constitue une ressource précieuse pour la recherche académique. Au-delà de la simple visualisation, ces données peuvent servir à :

— Étudier l'impact environnemental de l'aviation
— Optimiser les routes aériennes pour réduire les émissions
— Analyser les tendances de trafic et prédire la congestion
— Comparer l'efficacité énergétique des compagnies et types d'avions

Le secteur aérien représentant 2-3% des émissions mondiales de CO, disposer d'outils d'analyse accessibles et gratuits est essentiel pour la recherche climatique et la transition écologique.

# Annexes

## Annexe A : Installation

```
# Cloner le projet
git clone https://github.com/username/flight-tracker-pro

# Créer environnement virtuel
python3 -m venv venv
source venv/bin/activate

# Installer dépendances
pip install -r requirements.txt

# Lancer l'application
streamlit run main.py
```

## Annexe B : Dépendances

```
streamlit==1.39.0
pandas==2.2.3
requests==2.32.3
plotly==5.24.1
numpy==2.3.5
scipy==1.16.3
```

## Annexe C : Ressources

— OpenSky Network : `https://opensky-network.org/`
— Documentation API : `https://openskynetwork.github.io/opensky-api/`
— Streamlit : `https://streamlit.io/`
— Plotly : `https://plotly.com/python/`