



Rapport d'élève ingénieur

Projet de 2<sup>e</sup> année

Filière : F3 et F4

## Simulateur d'ouragan

*Présenté par : Ivan Le Teuff et Elisa Drouot*

Responsable isima : Aurélien Mombelli  
Référent isima : Jacques Laffont

Soutenance le vendredi 24 mars  
Projet de 4 mois

Campus des Cézeaux . 1 rue de la Chébarde . TSA 60125 . 63178 Aubière CEDEX

## Remerciements

Nous tenons à remercier Monsieur Aurélien MOMBELLI en tant que référent de projet pour toute l'aide et les conseils qu'il a pu nous donner tout au long de ce projet.

# Table des figures et illustrations

Figure 1 : Composants radiales et transverses de la vitesse .....	3
Figure 2 : Gantt prévisionnel .....	5
Figure 3 : Gantt réel .....	5
Figure 4 : Diagramme des cas d'utilisations du logiciel .....	6
Figure 5 : Template de l'onglet graphique .....	7
Figure 6 : Template de l'onglet multiple .....	7
Figure 7 : Diagramme de classe du logiciel .....	8
Figure 8 : Effet de la force de Coriolis sur le sens de rotation des tempêtes .....	10
Figure 9 : Effet de la force de Coriolis sur la force de pression .....	11
Figure 10 : Fonction polynomiale de degrés 2 passant par deux points A et B .....	12
Figure 11 : Fonction polynomiale de degrés 2 avec un minimum existant .....	13
Figure 12 : Chaîne de Markov à 3 états .....	14
Figure 13 : Points d'impact à gauche et trajectoires à droite des ouragans de la base HUR-DAT2 .....	17
Figure 14 : Un QHBoxLayout contenant cinq boutons .....	19
Figure 15 : Image du tracé d'un ouragan .....	20
Figure 16 : Image des paramètres du logiciel .....	20
Figure 17 : Diagramme de classe d'un singleton .....	21
Figure 18 : Onglet graphique du logiciel avec différents ouragans modélisés .....	22
Figure 19 : Directions et trajectoires des ouragans .....	23
Figure 20 : Résultats d'ouragans partant de différents endroits .....	23
Figure 21 : Visualisations au cours d'une simulation .....	24
Figure 22 : Fenêtre avec l'onglet de simulation multiple .....	24

## Table des tableaux

Tableau 1 : Correspondances position et coordonnées (latitude, longitude) .....	15
Tableau 2 : Allure de la base HURDAT2 .....	16

## Résumé

Le but de ce projet est de créer un logiciel avec une interface graphique permettant à l'utilisateur de pouvoir simuler le rayon et la trajectoires d'ouragans dans l'océan atlantique dans un rectangle de 60° de latitude et 110° de longitude.

Pour cela nous avons utilisé Qt Creator et codé en C++. Pour le rayon nous avons utilisés des équations de thermodynamique\* dépendant de paramètres extérieurs et pour la trajectoire la modélisation a été faite à l'aide d'une chaîne de Markov.

A ce jour nous avons le logiciel fonctionnel qui permet de mettre un ouragan à un endroit sur la carte et de lancer la simulation avec en temps réel la trajectoire et le rayon qui s'actualise.

**Mots clés :** C++, Chaîne de Markov, ouragan, Modélisation, Météorologie, Qt Creator, Simulation, Thermodynamique

## Abstract

The goal of this project was to develop a software with a graphical interface allowing the user to be able to simulate the hurricane radius and the hurricane trajectory in the atlantic ocean in a area of 60° latitude and 110° longitude.

For this purpose we used Qt Creator and coded in C++. For the radius we have used thermodynamic equations that depends on external parameters and for the trajectory, modelisation has been made with a Markov chain.

Currently our software is working and the user can put a hurricane somewhere on the map and throw the simulation with a hurricane radius and ouragan trajectory that change in real time.

**Keywords :** C++, hurricane, Markov Chain, Modeling, Meteorology, Qt Creator, Simulation, Thermodynamics

# Table des matières

Remerciements .....	i
Table des figures et illustration .....	ii
Résumé .....	iii
Abstract .....	iii
Table des matières .....	iv
Introduction .....	1
I. Contexte du projet .....	2
1. Description du projet .....	2
2. Étude du problème .....	2
3. Présentation des outils de développement .....	4
II. Réalisation et Conception .....	5
1. Répartition du travail .....	5
2. Conception du logiciel .....	6
2.1 Définition des cas d'utilisations .....	6
2.2 Création des templates .....	6
2.3 Création du diagramme de classe .....	8
3. Modélisation du rayon .....	9
3.1 Définition des limites .....	9
3.2 Méthode de résolution .....	9
3.3 Fonctionnement dans le code .....	12
4. Modélisation de la trajectoire des ouragans .....	13
4.1 Les hypothèses et le périmètre du modèle .....	13
4.2 Fonctionnement de la modélisation .....	14
4.3 Codage de la trajectoire .....	17
5. Programmation du logiciel .....	18
5.1 Mise en place du projet .....	18
5.2 Les bases du code .....	19
5.3 Implémentation du modèle mathématique .....	19
5.4 Utilisation de patrons de conception .....	20
III. Résultats et discussions .....	22
1. Onglet de lancement de simulation .....	22
2. Onglet de simulation multiple .....	24
Conclusion .....	25
Références Bibliographiques .....	v
Lexique .....	vi
Annexes .....	vii

## Introduction

Au cours de cette dernière année, il y a eu une augmentation du nombre d'ouragans dû au changement climatique, ce qui fait que de plus en plus d'endroits sont affectés et notamment des terres ou des gens vivent. Ce qui crée un enjeu au niveau de la modélisation des ouragans.

Ce projet se mariait bien aux compétences que l'on voulait développer, le développement de logiciel et la modélisation de systèmes. Une fois le projet proposé et accepté par M. Mombelli, nous sommes partis dans une phase de conception et de recherche d'informations.

Les ouragans sont des phénomènes compliqués à comprendre, ils impliquent de nombreux paramètres, notre objectif de simulation est de rester un maximum réaliste tout en gardant la complexité de ces phénomènes.

Notre projet se veut très accessible pour l'utilisateur, avec le choix total sur les paramètres de simulation.

Pour présenter ce projet, nous aborderons l'organisation globale et le déroulement du projet avant de présenter les outils utilisés et les méthodes que nous avons utilisés. Nous ferons un point sur la conception du logiciel puis nous verrons la modélisation du rayon des ouragans et ensuite leur déplacement. Puis on continuera sur la réalisation du logiciel avec l'implémentation des modèles réalisés. Avant de conclure, nous ferons un point sur les résultats obtenus.

# I. Contexte du projet

## 1. Description du projet

Le but de notre projet est de réaliser un simulateur d'ouragans. Nous nous sommes assez vite fixés sur deux grands axes : un axe interface graphique qui se présente sous la forme d'une application et un autre axe de modélisation. Dans ce deuxième axe nous avons eu pour objectif de pouvoir modéliser la trajectoire et le rayon d'un ouragan.

Dans les débuts du projets nous voulions pouvoir simuler la trajectoire et le rayon d'un ouragan depuis un point sur une carte imaginaire. Cependant nous nous sommes recentrés sur une simulation dans un cadre plus précis : nous avons simulé les ouragans dans l'océan atlantique au dessus de l'équateur.

Pour la partie logiciel, on avait pour idée d'avoir une interface libre pour l'utilisateur, où il avait beaucoup de contrôle disponible pour les simulations. Comme le placement des ouragans, ou le choix des paramètres.

## 2. Étude du problème

Du point de vue de la modélisation dans le cadre du calcul du rayon on peut utiliser différents modèles dont les deux principaux sont soit considérer l'ouragan comme une onde soit comme une gigantesque machine thermique.

Dans le modèle où l'on envisageait l'ouragan comme une onde on résout deux équations différentielles pour obtenir le système suivant [1] :

$$aa'\xi + a^2 - b^2 - (k - \mu)a + \omega b = 0, \quad \text{et} \quad ab'\xi + 2ab - (k - \mu)b + \omega a = 0$$

Avec  $\xi = \sqrt{X^2 + Y^2}$  tel que  $X = x - u * t, Y = y - v * t$   
où  $t$  est le temps et  $x'(t) = u(t), y'(t) = v(t)$ .

On note  $u$  et  $v$  les composantes de la vitesse\*.

De plus on a

$$u* = -\frac{\mu p x + \omega p y}{\mu^2 + \omega^2} \quad \text{et} \quad v* = \frac{\omega p x + \mu p y}{\mu^2 + \omega^2}$$

On a ici  $px$  et  $py$  les composantes du gradient de pression\*.

Les variables  $k, \mu$  et  $\omega$  (coefficient de Coriolis) sont des constantes.

Dans le cas où on prend  $\mu = 0$  et  $\omega = 0$  on a alors une solution non nulle  $a = k(1 - \frac{R}{\xi})$ ,  $b = 0$ .

On peut alors résoudre nos équations et obtenir la figure suivante :

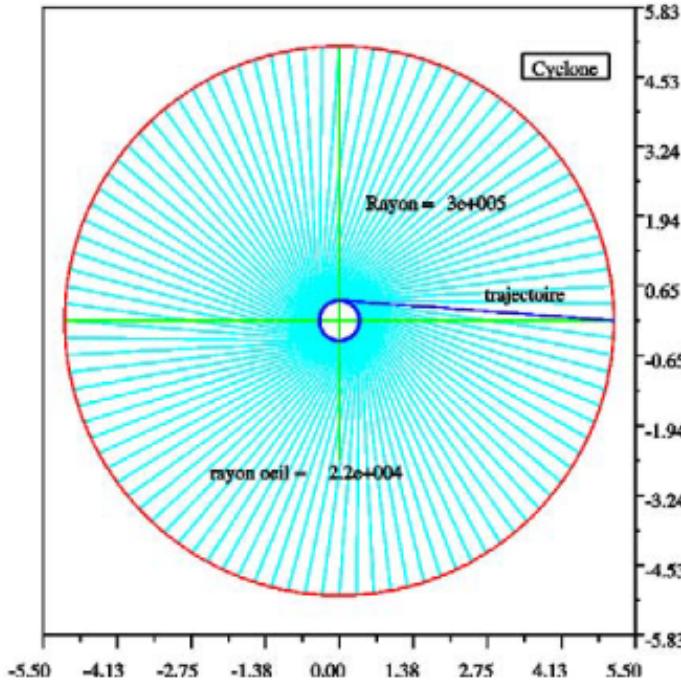


FIGURE 1 – Composants radiales et transverses de la vitesse

On voit que ce modèle est assez complexe dans le sens où en plus de calculer le rayon de l'ouragan il calcule aussi le rayon de l'œil de l'ouragan (zone de vents calmes et de temps clément siégeant en général au centre de la circulation de l'ouragan) ainsi que la direction du vent au sein de l'ouragan. C'est pour cela que pour calculer le rayon nous avons choisi un modèle plus simple : celui qui modélise l'ouragan comme gigantesque machine thermique à partir des paramètres environnants.

Pour la modélisation de la trajectoire des ouragans on fait face à deux catégories de modèles :

- Les modèles statistiques : ce type de prévisions repose sur la répétitivité dans l'espace et dans le temps des trajectoires des ouragans tropicaux. On utilise alors une base de donnée pour faire nos prédictions. Par exemple dans les modèles CLIPER, HURRAN et MOCCANA on va alors rechercher dans la base de données des anciens ouragans ayant des caractéristiques proches de l'ouragan étudié, en intensité, position, déplacement, au cours de la même partie de la saison, et on observe leur évolution. On attribue alors à l'ouragan en cours le même type de comportement.

- Les modèles dynamiques : le principe est d'étudier l'ouragan uniquement grâce aux paramètres météorologiques à un moment donné. Son environnement connu grâce aux différents paramètres analysés (vents, températures, humidité, pression atmosphérique, vitesse verticale, etc ...) il existe différents modèles (AVN, NOGAPS, MRF, ect).

- Les modèles statistico-dynamiques : allient les avantages des 2 types précédents. Ils combinent la statistique, par analogie aux comportements d'ouragans répertoriés dans une base, et l'aspect dynamique, prenant en compte les différents aspects de l'environne-

ment météorologique de l'ouragan étudié. Ce sont des modèles ayant la réputation d'être de bonne qualité, et ont fait leurs preuves durant les dernières années.

Dans notre projet nous avons choisi d'utiliser un modèle statistique, d'une part par sa simplicité (en effet il n'y a pas énormément de paramètres à prendre en compte) d'autre part car nous l'avons trouvé intéressant avec cet aspect de traitement de nombreuses données.

### 3. Présentation des outils de développement

**Qt :** Qt est une interface de programmation d'application(API) qui est une surcouche du langage de programmation C++. Qt permet de créer des logiciels graphiques en simplifiant les procédés de liens entre la partie graphique et la partie logique du logiciel. Par exemple en simplifiant l'affectation de l'action d'appui d'un bouton graphique à une fonction logique. La documentation fournie est complète et facile à utiliser.

**Qt Creator :** C'est l'environnement de développement associé à Qt, il permet de créer des projets Qt et de les développer. Il contient un onglet debugger très bien fait et aussi un onglet Design permettant de créer facilement l'aspect graphique du logiciel en création.

**GitLab :** C'est un logiciel de forge qui héberge l'outil de versionnement Git, il permet de gérer efficacement le travail de groupe sur des ressources partagées ainsi que la gestion des versions du projet. On utilise celui hébergé par l'ISIMA.

**Balsamiq :** C'est un logiciel de création de template\*, pour des logiciels de tous formats, appareils mobiles ou ordinateur. Il permet aussi de lier les différents templates\* entre eux, très utile pour faire des démonstrations d'utilisation. On a utilisé la version d'essai.

**draw.io :** C'est un site internet de création de diagrammes, tels que des diagrammes de cas d'utilisation ou de classe\*.

## II. Réalisation et Conception

### 1. Répartition du travail

La répartition du travail était clair dès le début du projet. Ivan sur la partie de la conception et développement du logiciel et Élisa sur la partie mathématique et modélisation des ouragans. Cette répartition nous a amené à détacher les principales tâches du projet. Cela nous a permis d'établir le planning prévisionnel suivant.

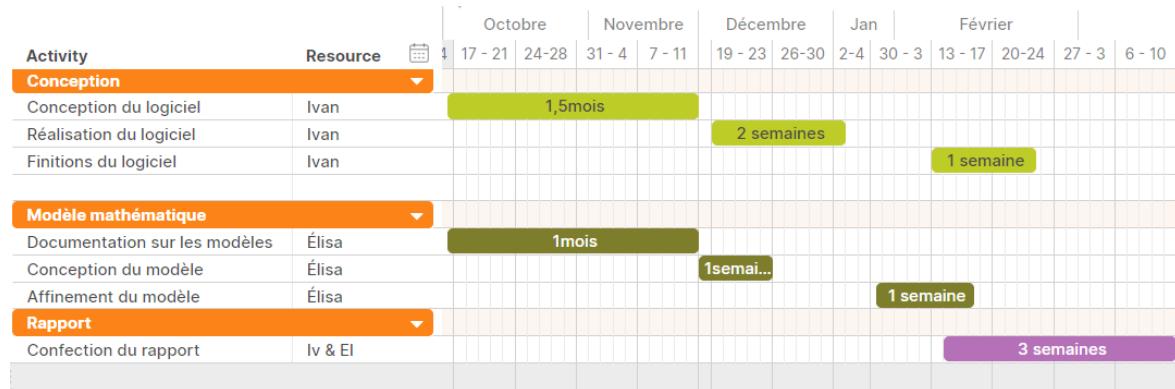


FIGURE 2 – Gantt prévisionnel

Le projet était prévu pour être travaillé sur 3 périodes :

Octobre à décembre : période de conception et de recherche.

Vacances de décembre : programmation du logiciel et conception du modèle mathématique.

Vacances de février : Mise au propre du logiciel avec implémentation des modèles mathématiques.

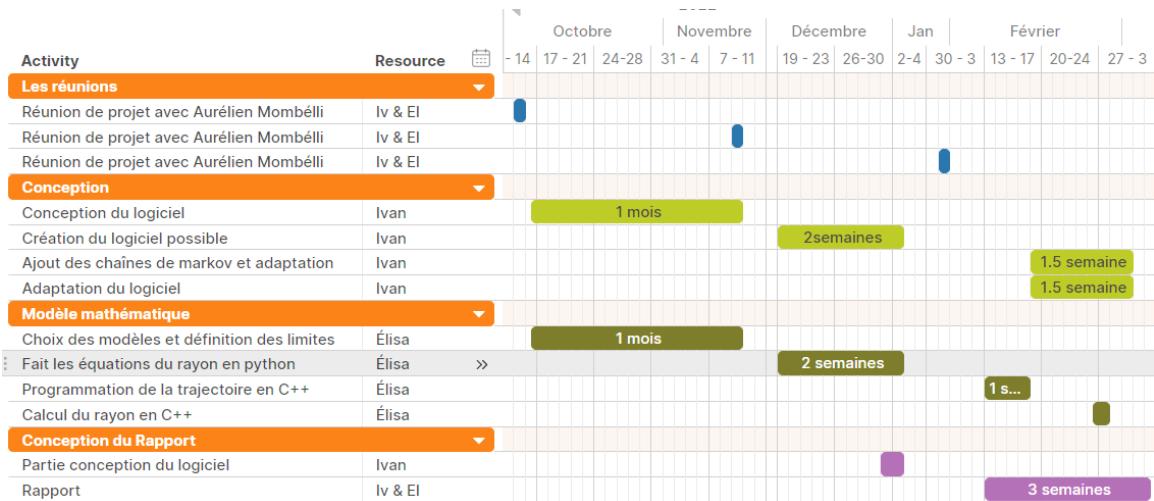


FIGURE 3 – Gantt réel

Les temps de travail ont été globalement bien prévu. La principale différence entre le planning prévisionnel et réel est la modélisation des équations qui a été plus longue que

prévu et donc pas adapté en C++ pour les vacances de Noël. Cela a engendré un retard pour les vacances de février. Nous n'avions pas programmé les réunions.

## 2. Conception du logiciel

### 2.1 Définition des cas d'utilisations

La conception du logiciel est la première chose sur laquelle on s'est penché, notamment avec la définition des cas d'utilisations.

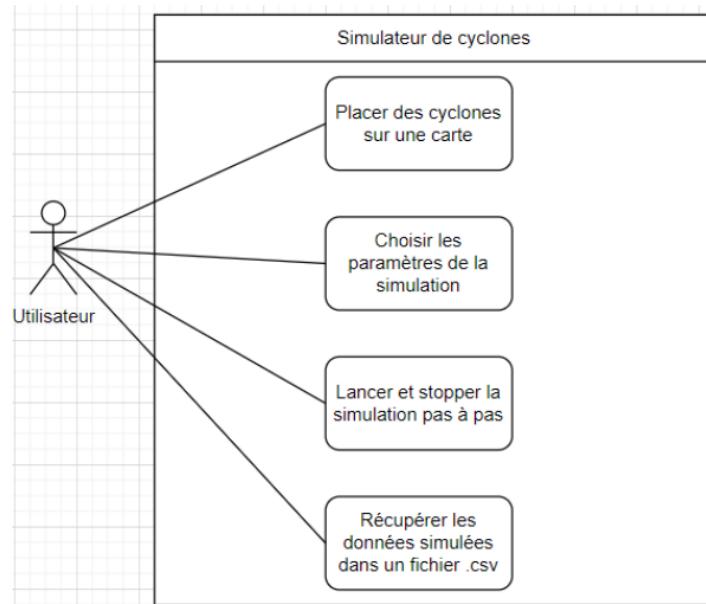


FIGURE 4 – Diagramme des cas d'utilisations du logiciel

Il permet de fixer les principaux objectifs du projet.

### 2.2 Crédit des templates

Nous avons enchaîné avec les attendus graphiques, pour cela nous avons regardé ce qu'il était possible de faire avec Qt, nous avons donc créé deux templates\*, correspondant aux deux onglets que l'on voulait :

Le template\* simulation graphique, qui permet de voir graphiquement le déplacement des ouragans. Il contient les choix des différents paramètres, le tableau des ouragans et la carte graphique.

Le template\* de simulation multiple, qui permet de faire plusieurs simulations et d'enregistrer les données obtenues dans un fichier au format csv. Il contient les choix de paramètres, le choix du nom de fichier et le bouton simuler.

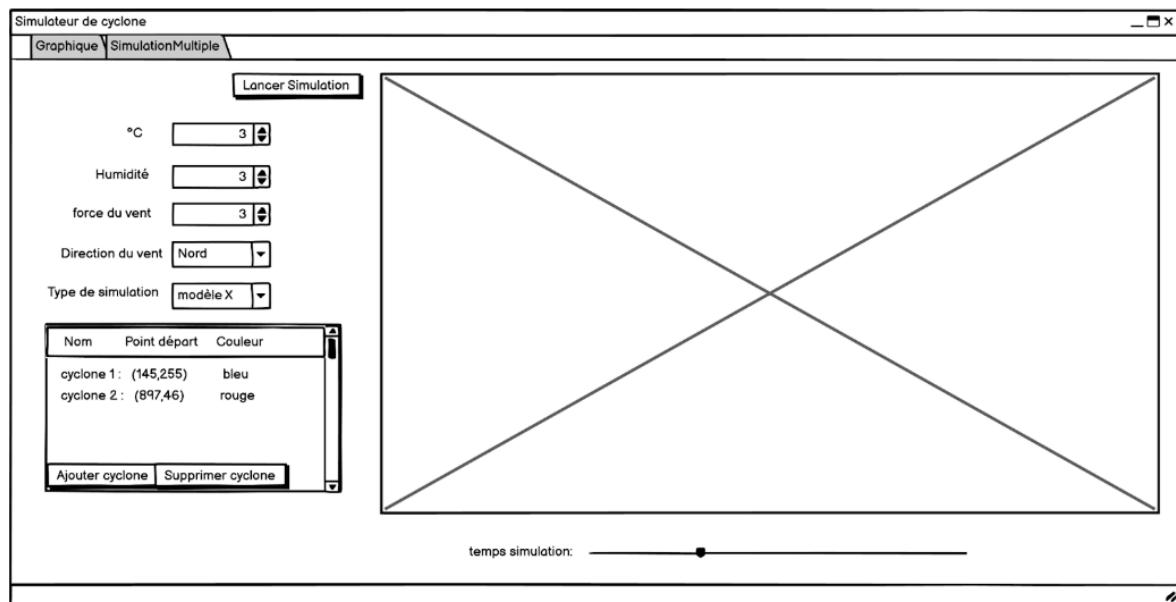


FIGURE 5 – Template de l'onglet graphique

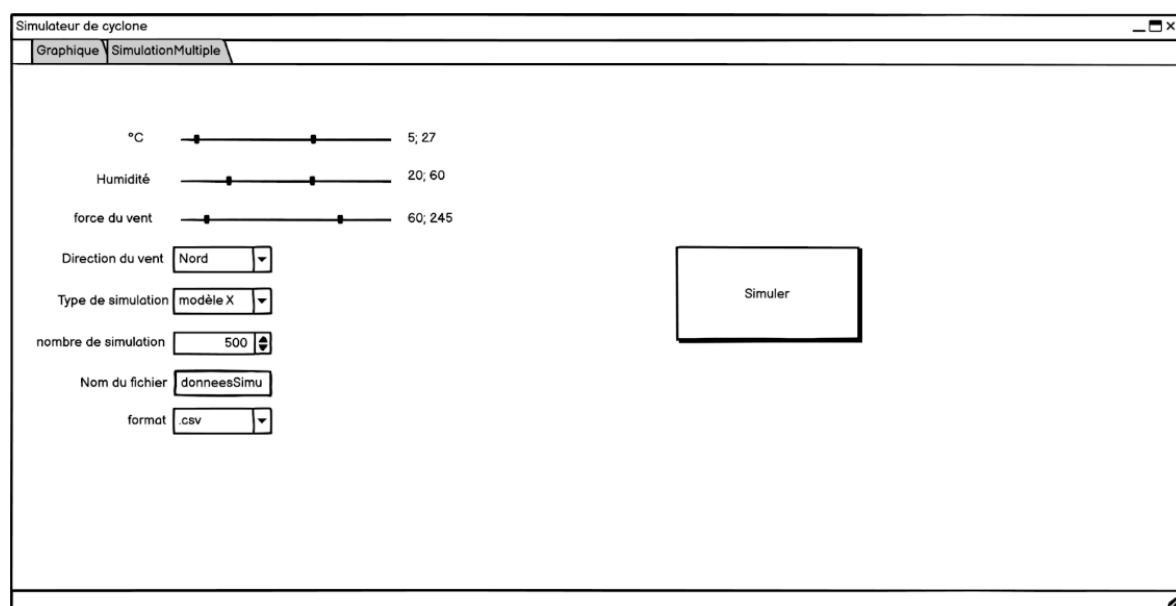


FIGURE 6 – Template de l'onglet multiple

## 2.3 Création du diagramme de classe

Cette étape était compliquée car on ne savait pas encore comment allaient être les différents paramètres, s'il y allait avoir plusieurs types de simulations et comment elles seraient. Malgré ce manque d'informations, on a réalisé un diagramme de classe\* général, pour obtenir la base nécessaire à la programmation de notre logiciel.

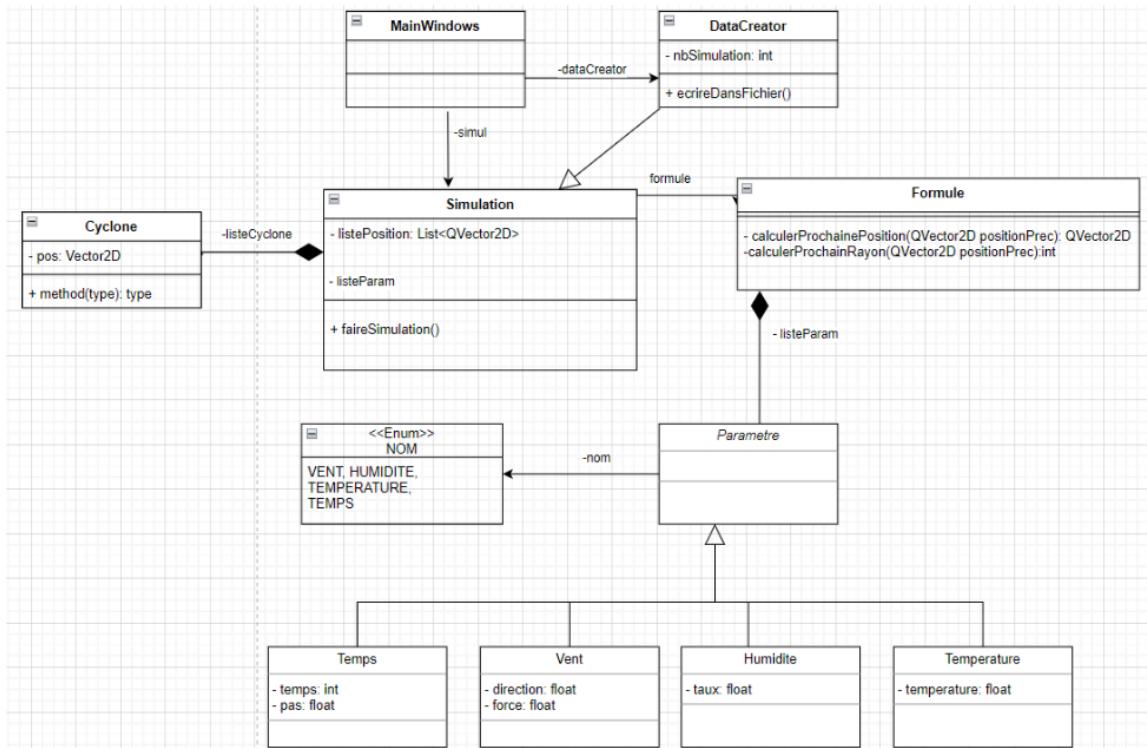


FIGURE 7 – Diagramme de classe du logiciel

### 3. Modélisation du rayon

#### 3.1 Définition des limites

Avant la formation de l'ouragan il y a ce que l'on appelle la formation d'une dépression tropicale, à ce moment la vitesse des vents est inférieure à 17 m/s, cette dépression tropicale devient ensuite un tempête tropicale où la vitesse des vents est comprise entre 17 m/s et 32 m/s, puis en fin lorsque la vitesse des vents devient supérieure à 32 m/s dès lors la formation devient un ouragan.

L'ouragan dépend de plusieurs paramètres notamment :

- La température : celle de l'eau sur au moins 60 mètres de profondeur doit être supérieure à 26 °C, sur une très large étendue maritime. Ces eaux chaudes vont en fait servir de carburant et de source de chaleur au système.
- La pression atmosphérique : l'atmosphère doit posséder une instabilité suffisamment élevée pour permettre aux particules de monter facilement vers les hautes couches de l'atmosphère.
- L'humidité : elle doit être d'au moins 70 % soit être entre 700 hPa et 400 hPa.
- Contraintes dynamiques : il faut un cisaillement vertical\* et l'existence d'une perturbation\*.

Dans notre projet les paramètres intervenants sur le rayon sont la température initiale de l'air, la pression atmosphérique, la vitesse du vent et la latitude.

Une fois formé, un ouragan se comporte comme une gigantesque machine thermique qui puise son énergie du déséquilibre thermodynamique\* entre l'atmosphère et l'océan qui jouent respectivement le rôle de source froide et de source chaude. On peut simplifier le bilan d'énergie en considérant que le système transforme l'énergie thermique (qui est l'énergie cinétique due aux mouvements des atomes ou molécules d'un corps) extraite de l'océan en énergie cinétique\*. Ainsi, plus un ouragan se déplace dans des eaux chaudes et plus il gagne en énergie et devient dévastateur alors qu'au contraire il perd en intensité lorsqu'il traverse des eaux plus froides. Pour donner un ordre d'idée le taux de chauffage moyen de l'atmosphère dû à la condensation de vapeur d'eau est de l'ordre de 25 °C/ jour, il y a environ 10 cm de pluie par jour.

#### 3.2 Méthode de résolution

La grande hypothèse que nous aurons sera de considérer les ouragans comme des disques parfaits, ce qui est rarement vrai dans la réalité. En effet, du fait de son déplacement et de sa composition nuageuse enroulée en spirale le phénomène est très souvent asymétrique.

Pour décrire le fonctionnement d'un ouragan tropical on peut modéliser l'ouragan par une machine thermique avec une source chaude et une source froide on obtient alors

l'équation qui suit [2] :

$$\epsilon = \frac{TS - T0}{Ts}$$

avec

- $TS$  et  $T0$  respectivement les températures des sources chaude (l'océan) et froide (l'atmosphère).
- $\epsilon \simeq \frac{1}{3}$

Quand on a ni  $VTs/r >> f$  ni  $V_{Ts}/r << f$  :

$$V_{Ts}(f + \frac{V_{Ts}}{r}) = C_p(\theta_{\nu 0} + \theta_{CS}) \frac{\partial \pi_S}{\partial r} = \frac{\partial h_S}{\partial r}$$

L'équilibre hydrostatique s'écrit :

$$g \frac{\theta_{CS}}{\theta_{\nu 0}} = C_p(\theta_{\nu 0} + \theta_{CS}) \frac{\partial \pi_S}{\partial z} = \frac{\partial h_S}{\partial z}$$

avec

- $V_{Ts}$  le vent tangentiel
- $h$  la hauteur géopotentielle
- $f$  la force de Coriolis
- $r$  la distance depuis le centre
- $\theta_{\nu 0}$  la température potentielle de l'environnement
- $\theta_{CS}$  la perturbation\* de température potentielle =  $\lambda * T$
- $\pi_S$  la perturbation\* de pression réduite

L'indice (s) représente les composantes symétriques, et l'indice (0) les valeurs de l'environnement.

La force de Coriolis est une force qui s'exerce sur les ouragans elle vient du fait que en remontant vers le nord on se rapproche de l'axe de rotation terrestre passant du pôle sud au pôle nord. En se rapprochant du pôle Nord à la surface de la terre, la vitesse réelle en rotation diminue car on se rapproche de cet axe.

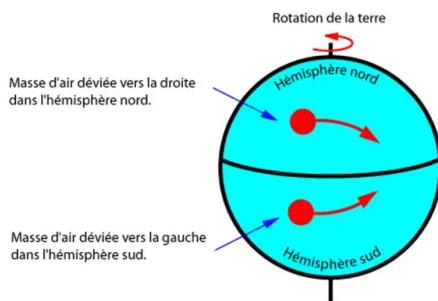


FIGURE 8 – Effet de la force de Coriolis sur le sens de rotation des tempêtes

On voit donc bien que la force de coriolis va affecter le sens de rotation des ouragans selon l'hémisphère.

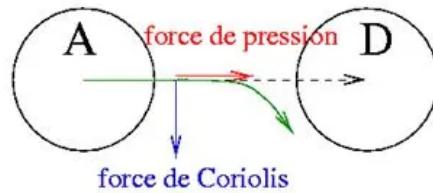


FIGURE 9 – Effet de la force de Coriolis sur la force de pression

Ce sens de rotation est dû au sens de la force de Coriolis qui s'applique sur la force de pression.

En combinant les 2 équations on obtient l'équation suivante :

$$(f + \frac{V_{Ts}}{r})(\frac{\partial V_{Ts}}{\partial z}) = (\frac{g}{\theta_{\nu 0}})(\frac{\partial \theta_{Cs}}{\partial r})$$

ce qui donne :

$$r = \frac{1}{f}(g \frac{\theta_{Cs}}{\theta_{\nu 0}} \frac{1}{V_{Ts}} \partial z - V_{Ts})$$

si on considère  $g \frac{\theta_{Cs}}{\theta_{\nu 0}} \frac{1}{V_{Ts}}$  constant on obtient :

$$r = \frac{1}{f}(g \frac{\theta_{Cs}}{\theta_{\nu 0}} \frac{1}{V_{Ts}} z - V_{Ts})$$

Avec le document [3] on trouve que :

$$f = 1,454 \cdot 10^{-4} \cdot \sin(\text{latitude})$$

Or la température potentielle est définie par :

$$\theta = T_0 \left( \frac{P_{std}}{p} \right)^{R_a/C_p} = T_0 \left( \frac{P_{std}}{p} \right)^{2/7}$$

Avec :

- $P_{std}$  fixée à 1 000 hPa
- p la pression en hPa
- $T_0$  la température de l'air exprimée en Kelvins

On obtient finalement :

$$r = \frac{1}{f} \left( \frac{g \lambda T}{T_0} \left( \frac{p}{P_{std}} \right)^{2/7} \frac{z}{V_{Ts}} - V_{Ts} \right)$$

### 3.3 Fonctionnement dans le code

Nous avons codé le rayon en c++ pour ensuite l'intégrer au reste du code.

Pour implémenter le rayon dans le code on a tout d'abord les constantes  $P_{std} = 1000 \text{ hPa}$ ,  $g = 9,80665 \text{ m/s}$ ,  $\lambda = 0,00706$  (lambda est la constante de la convection) et pour la hauteur  $z = 15000 \text{ m}$  (sachant que z doit être entre 10000 et 15000m). Pour les paramètres initiaux nous les avons fixés à  $P_0 = 1015 \text{ hPa}$ ,  $T_0 = 27 + 273.15 \text{ K}$  et  $V_{Ts} = 32 \text{ m/s}$ .

Les paramètres qui changent dans cette équation à chaque itération sont les paramètres T et P (la température et la pression). Pour les faire changer on utilise une fonction qui aura la forme d'une parabole (fonction en forme de U) qui change à chaque itération.

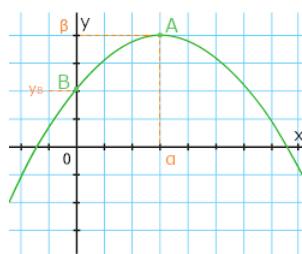


FIGURE 10 – Fonction polynomiale de degrés 2 passant par deux points A et B

La fonction  $f$  a alors l'allure montrée par la figure ci dessus, c'est une fonction polynomiale de degrés deux, c'est à dire que l'on a  $f(x) = ax^2 + bx + c$ .

Pour trouver comment calculer  $f$  on la met sous la forme  $a(x - \alpha)^2 + \beta$  avec  $\alpha$  et  $\beta$  les coordonnées du sommet (représenté par le point A sur la figure).

Or on a  $f(0) = y_B$  donc  $y_B = f(0) = a(0 - \alpha)^2 + \beta$ .

Finalement on obtient :

$$a = -\frac{y_B - \beta}{\alpha^2}$$

Dans le cas de la modélisation du rayon la température et la pression atteignent les valeurs extrêmes  $30^\circ \text{C}$  et  $999 \text{ hPa}$ . Donc pour la température on aura  $\alpha = 30 + 273.15$  et  $y_B = T_0$  avec  $T_0$  défini plus tôt, de même pour la pression on aura  $\alpha = 999$  et  $y_B = P_0$  avec  $P_0$  aussi défini plus haut.

Pour la pression la parabole sera inversée par rapport à l'axe x et aura donc l'allure de la figure suivante :

La fonction  $f$  aura alors bien pour minimum  $999 \text{ hPa}$ .

Le problème restant étant de trouver  $\alpha$  : on remarquera que la variable  $x$  vaudra le nombre d'itérations déjà faites à chaque temps de pas. Donc dans l'idéal :

$$\alpha = \frac{\text{Nombre final d'itérations}}{2}$$

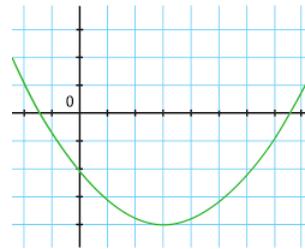


FIGURE 11 – Fonction polynomiale de degrés 2 avec un minimum existant

Or on ne peut pas connaître le nombre d’itérations à l’avance, on a donc eu une solution pour ce problème : fixer  $\alpha$  au nombre moyen d’itérations faites que l’on divise ensuite par deux, ce qui en soit n’est pas optimal. Une autre solution possible qui serait plus juste serait de calculer le rayon sur un ouragan dont la trajectoire a déjà été calculé et dont on connaît déjà le nombre d’itération, cependant nous n’avons pas eu le temps d’implémenter cette solution.

## 4. Modélisation de la trajectoire des ouragans

### 4.1 Les hypothèses et le périmètres du modèle

Pour la compréhension du modèle il est nécessaire de décrire les hypothèses que nous avons faites :

- On considérera qu’un pas de temps symbolise six heures.
- On considérera la déplacement d’un ouragan entre deux points comme étant un mouvement de translation à vitesse constante c’est à dire qu’il reste parallèle à lui-même au cours du mouvement (cf annexe 1 : Représentation des mouvements de translation) et que la vitesse n’augmentera pas ou ne diminuera pas au cours du mouvement.

On en vient ensuite aux périmètres :

- On a d’une part le périmètre temporel : la base de donnée utilisée allant de 1851 à 2014 on se limite donc à 163 années de données. Une des remarque que l’on pourrait faire sur ce périmètre c’est que la qualité des données avant 1970 est moins bonne car cette année marquera l’utilisation de satellites pour les mesures, les ouragans étant jusque là mesurés par des bateaux et des stations météorologiques. Il n’est donc pas exclu que des portions de trajectoires soient manquantes voir même que certaines tempêtes n’aient jamais été détectées.
- D’autre part on posera un périmètre spatial : on restera focalisé sur les études d’ouragans se situant entre  $0^\circ$  et  $60^\circ$  de latitude et  $0^\circ$  à  $110^\circ$  de longitude au niveau de l’océan Atlantique.

Il est intéressant de noter que avec notre périmètre spatial on peut voir que d’après l’annexe 2(cf annexe 2 : Nom du phénomène selon le lieu) on modélise bien des ouragans et non pas des cyclones ou des typhons : on parle de cyclone lorsque le phénomène à lieu au niveau de l’océan Indien et au Pacifique sud. On parle en revanche d’ouragan en Atlantique nord et dans le Pacifique nord-est et enfin de typhon dans le Pacifique nord-ouest.

## 4.2 Fonctionnement de la modélisation

Pour déterminer la trajectoire des ouragans on utilisera un modèle statistique se basant sur une base de données. Le principe étant d'à partir de cette base de données aboutir à une chaîne de Markov.

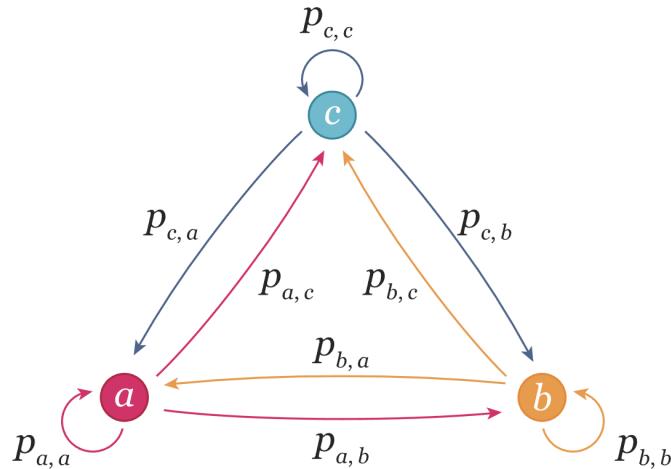


FIGURE 12 – Chaîne de Markov à 3 états

Une chaîne de Markov possède des états ici a, b et c. Les flèches indiquent les probabilités de transition d'un état à un autre par exemple si on se trouve à l'état c on a la probabilité  $p_{c,c}$  de rester en c, la probabilité  $p_{c,a}$  de passer à l'état a et la probabilité  $p_{c,b}$  de passer à l'état b. Ce qu'il faut retenir c'est qu'une chaîne de Markov est un modèle permettant d'étudier un phénomène aléatoire au cours du temps (soit un processus stochastique) et que son passage d'un état à un autre dépendant que de l'état présent (pas de ceux antérieurs), on dit aussi que le système n'a pas de «mémoire».

Dans le cas de notre modélisation on utilise donc une chaîne de Marcov, chaque état représentant une position (une latitude et une longitude) et nous avons donc dans un premier temps chercher les probabilités de transition qui permettent de passer d'un état à un autre, ce qui veut dire d'une position à l'autre, c'est ce qui permet donc de passer d'une position à l'autre.

On numérote alors les positions comme sur la figure qui suit :

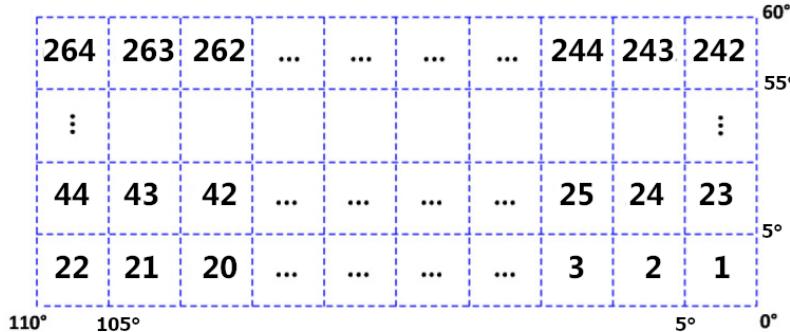


TABLEAU 1 - Correspondances position et coordonnées (latitude, longitude)

Ce tableau illustre comment on obtient des positions à partir d'une latitude et d'une longitude. On peut voir que nous avons choisi de d'avoir une précision de  $5^\circ$ , l'idéal serait d'avoir une précision de  $1^\circ$  mais cela ajoute des ressources et de la complexité. On observe que nous avons 264 positions.

Pour obtenir la position à partir d'une latitude et d'une longitude on le fait à l'aide des équations suivantes :

$$\text{Position} = 22f(\text{Latitude}) + f(\text{Longitude}) + 1$$

$$\text{avec } f(x) = \text{quotient de } \frac{x}{5}$$

De même pour ensuite obtenir la latitude et la longitude à partir d'une position on utilise :

$$\text{Longitude} = (\text{Position} - 1) \equiv 22x_5 + 2,5$$

$$\text{Latitude} = ((\text{Position} - 1)/22)x_5 - 2,5$$

On a  $a \equiv b$  qui est le reste de la division de  $a$  par  $b$ . Ces équations nous renverrons les coordonnées du milieu de la case par exemple la position 1 aura une latitude de 2,5 et une longitude de 2,5.

Une chaîne de Markov possède une matrice de transition, si on reprend la figure 8 on obtient la matrice suivante :

$$\begin{pmatrix} p_{a,a} & p_{a,b} & p_{a,c} \\ p_{b,a} & p_{b,b} & p_{b,c} \\ p_{c,a} & p_{c,b} & p_{c,c} \end{pmatrix}$$

C'est de cette manière que l'on va stocker notre chaîne de Markov : sous la forme d'une matrice. Il faut ajouter que dans notre matrice de transition un ouragan ne peut rester sur lui-même donc  $\forall i \in [1, 264]$  on a  $p_{i,i} = 0$  notre matrice ressemblera donc à ça :

$$\begin{pmatrix} 0 & p_{1,2} & \dots & p_{1,264} & p_{1,265} \\ p_{2,1} & 0 & \dots & p_{2,264} & p_{2,265} \\ \dots & & & & \\ p_{264,1} & p_{264,2} & \dots & 0 & p_{264,265} \end{pmatrix}$$

La matrice est de taille 264x265 car on a 264 états et la dernière colonne contient la probabilité que l'ouragan «meurt», c'est à dire que le phénomène s'arrête.

Il faut aussi noter que notre matrice de transition a la propriétés de base :

$$\forall i \in [1, 264], \quad \sum_{j=1}^{265} p_{i,j} = 1$$

C'est-à-dire que la somme des termes de n'importe quelle ligne est toujours égale à 1.

Par contre nous avons apporté une modification à notre chaîne de Markov, en effet normalement elle ne devrait dépendre que de l'état actuel mais elle dépend aussi de l'état précédent ce qui veut dire que si à l'instant d'avant l'ouragan était à la position 24 et qu'il est actuellement à la position 25 il ne peut pas retourner en position 24. Nous avons posé cela pour éviter les phénomènes de demis tours qui ne sont pas réalistes.

On peut montrer avec t un temps donné :

$$P[Position_{t-1} = position_i | Position_t = position_j | Position_{t+1} = position_i] = 0$$

Avec A|B qui veut dire B sachant A.

Pour constituer la matrice de transition nous avons utilisé la base de données HUR-DAT2 qui a les données d'ouragan s'étant produits de 1851 à 2014.

StormID	StormName	Year	Month	Day	Hours	ReIdentifier	Lat	Lon	MaxWind	MinPress	Cat
AL011950	ABLE	1950	8	12	0		17.1	-55.5	35.0	-999.0	0
AL011950	ABLE	1950	8	12	0		17.7	-56.3	40.0	-999.0	0
AL011950	ABLE	1950	8	12	1		18.2	-57.4	45.0	-999.0	0
AL011950	ABLE	1950	8	12	1		19.0	-58.6	50.0	-999.0	0
AL011950	ABLE	1950	8	13	0		20.0	-60.0	50.0	-999.0	0

TABLEAU 2 - Allure de la base HURDAT2

Dans cette base de données on s'intéressera uniquement au numéro identifiant l'ouragan, à sa latitude et à sa longitude. Une chose à remarquer est par ailleurs le fait qu'entre chaque mesure on a 6h d'écart. Il peut aussi être intéressant de voir la répartition des points de cette base de donnée.

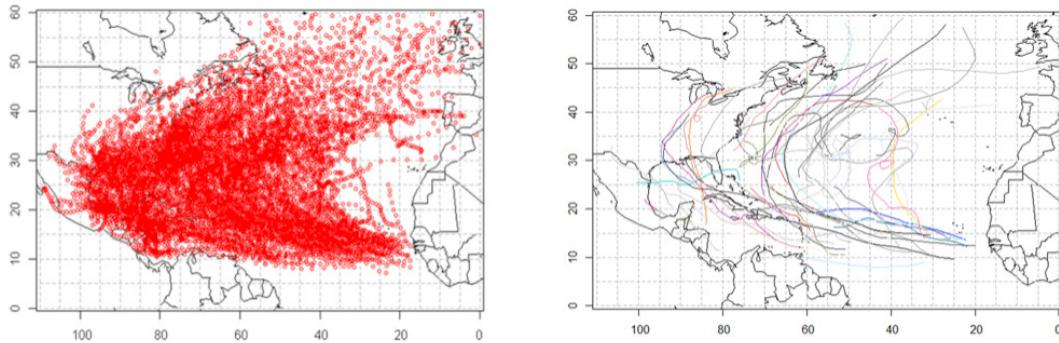


FIGURE 13 – Points d’impact à gauche et trajectoires à droite des ouragans de la base HURDAT2

On observe une grande concentration des points dans une certaine zone : au dessus de l’océan et sur la côte ouest des États Unis et d’Amérique Latine. Les îles dans les caraïbes aussi sont sous cette masse de points. Pour la trajectoire on voit quelle présente la caractéristique d’être arrondie et dit de manière grossière de partir d’en bas à droite pour monter jusqu’en haut à droite.

Pour en revenir à la création de la matrice de transition on utilise une matrice creuse, on parcours la base de donnée. Dès que l’on passe d’une position i à une position j on ajoute 1 à la case  $i,j$  de la matrice. Une fois que l’on a parcourut toute la base de données on fait ces opérations ligne par ligne :

$$\forall i \in [1, 264], \quad somme = \sum_{j=1}^{265} p_{i,j}$$

$$p_{i,j} = \sum_{k=1}^j p_{i,k}$$

Pour finir on divise toute la ligne par le terme *somme* et on obtient alors notre matrice de transition. Il faut cependant pas oublier de traiter le cas de «mort» de l’ouragan en vérifiant que l’ouragan n’est toujours pas mort.

### 4.3 Codage de la trajectoire

Le code a ici aussi été réalisé entièrement en c++, en commençant par le codage d’un objet\* matrice creuse. L’usage du matrice creuse est très intéressant ici car de nombreuses valeurs de notre matrices sont nulles, cela permet donc d’optimiser l’espace utilisé. Par exemple si on prend une matrice avec beaucoup de valeur nulles :

$$\begin{pmatrix} 4 & 2 & 0 & \dots & 0 \\ 0 & 3 & 0 & \dots & 0 \\ 0 & 0 & 6 & \dots & 0 \\ \dots & & & & \\ 0 & 0 & 0 & \dots & 0 \end{pmatrix}$$

Il sera plus simple de ne stocker que les indices et les valeurs non nulles comme suit :  
 1 1 4 (ici on retrouve la valeur 4 qui était dans la ligne 1 colonne 1 dans la matrice )  
 1 2 2  
 2 2 3  
 3 3 6

Pour simplifier la manipulation de la matrice nous avons choisi de trier dans l'ordre croissant les lignes et de créer en parallèle un tableau contenant le nombre d'élément pour chaque ligne. Pour illustrer on peut prendre cet exemple :

1 1 4      donnerait le tableau [2,0,1,1]  
 1 2 2  
 3 3 6  
 4 2 5

Cela rend par exemple l'insertion plus facile car si on veut insérer un élément de la ligne 4 il suffit de récupérer les 3 premières valeurs du tableau et de les additionner pour savoir à quel indice insérer l'élément.

Un objet\* construction est ensuite utilisé pour construire la matrice. On parcourt la base de données avec la position précédente et la position suivante, si on n'a pas encore d'élément à l'indice (position précédente, la position suivante) on crée l'élément, sinon on va lui ajouter 1. On compare également les numéros d'ouragan, si il diffère on ajoute un à l'élément (position précédente, 265) de la matrice. La création de la matrice sera faite avec des Vectors ce qui en C++ permet de faire des tableaux à taille variable.

Ensuite on fera les opérations décrites dans la partie précédente pour obtenir la matrice de transition.

La dernière étape consiste à trouver la position suivante à l'aide de la position courante et de la matrice de transition. Pour ce faire on génère un nombre aléatoire entre 0 et 1 (notons le r) puis on parcourra toute la ligne dont l'indice est la position courante jusqu'à ce que l'on trouve h tel que  $p_{courante,h-1} \leq r$  et  $p_{courante,h} \geq r$ . La position suivante devient alors la position h. On refait cette manipulation jusqu'à ce que notre ouragan « meurt ».

## 5. Programmation du logiciel

### 5.1 Mise en place du projet

Nous avons commencé par créer un projet vierge pour apprendre à utiliser Qt Creator, on a testé les différentes fonctionnalités. Par exemple on a disposé différents boutons grâce à l'onglet design de Qt Creator (cf annexe 3 : Onglet design de Qt Creator) et on les a ensuite lié à des fonctions logiques.

Ensuite nous avons créé le vrai projet, que nous avons déposé sur le gitlab du projet afin de sauvegarder les différentes versions du projet et partager facilement les fichiers du projet.

## 5.2 Les bases du codes

Pour l'aspect graphique du logiciel, on a reproduit les templates\* graphiques fait pendant la conception à l'aide de l'onglet design de Qt Creator(cf annexe 3 : Onglet design de Qt Creator), il permet de disposer les éléments graphiques tels que les boutons, les champs sélecteur de valeurs pour les paramètres demandé à l'utilisateur ou encore du texte. Pour cela il faut glisser déposer les éléments graphiques choisis de la colonne à gauche dans la zone centrale. Il y a deux types d'éléments, les contenants et les contenus. Les contenants sont les Layouts\*, les contenus sont les éléments graphiques visibles.

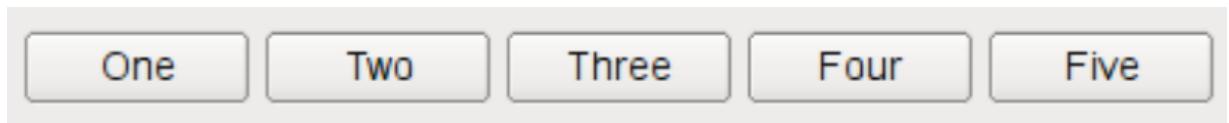


FIGURE 14 – Un QHBoxLayout contenant cinq boutons

Sur cette figure, le contenu est un layout\* horizontal, les contenus sont les cinq boutons.

Malgré son côté intuitif, c'est compliqué d'avoir des répartitions non équitables entre les contenus, si on veut une disposition non équitable entre les éléments, il faut fixer leurs politiques d'étirements un à un, ce qui permet de les laisser fixe ou alors de les étendre à notre convenance. Une fois cette astuce maîtrisée, on a complété le côté graphique du projet en suivant correctement les templates\* (cf Annexe 4 : Premier visuel du logiciel).

Pour la logique du logiciel, nous avons suivi le diagramme de classes\* fait lors de la conception, on a créé les différents fichiers et les liens entre les objets\*. On veut pouvoir avoir plusieurs ouragans en parallèle, donc chaque ouragan sera représenté par une instance\* de la classe ouragan. L'objet Map contiendra les ouragans et s'occupera de les afficher correctement sur l'espace associé. L'objet mainwindows contient tous les objets, c'est cette classe qui contient les objets graphiques et qui peut les associer aux fonctions logiques. Nous avons réalisé différentes tâches :

- Gérer les listes d'ouragans.
- Gérer l'affichage des ouragans avec des glisser/déposer.
- Lier le tableau des ouragans et les ouragans affichés.
- Lier les boutons à des fonctions pour ajouter/supprimer des ouragans.
- Ajouter la flèche de la direction du vent.

## 5.3 Implémentation du modèle mathématique

Une fois le modèle fini, il a fallu l'implémenter dans le logiciel, cela a causé quelques changements.

Le plus gros changement est le fait que le nombre d'emplacement des ouragans est limité à 264 places, donc il a fallu quadriller la map, et adapter le fait que l'utilisateur peut glisser/déposer un ouragan où il veut, maintenant l'ouragan est déposé dans la zone correspondant à l'endroit du lâché. Ce n'est plus au pixel près comme avant.

Il a fallu ajouter les cycles de simulation, comme le fait de continuer tant que tous les ouragans ne sont pas morts. Pour garder une trace du parcours des ouragans, il a fallu ajouter un tracé des positions comme sur la figure ci-dessous.

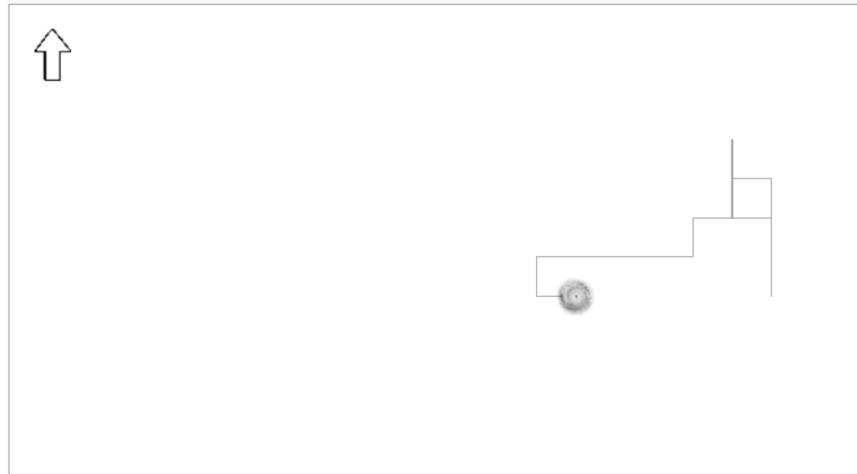


FIGURE 15 – Image du tracé d'un ouragan

On a pu mettre les vrais paramètres nécessaires au modèle en choix à l'utilisateur. On a lié les éléments graphiques à des variables dans le code.

Lancer la simulation	Remise à zéro
Température (°C)	26
Pression atmosphérique (hPa)	1000
Vitesse du vent (km/h)	116
Direction du vent °	0
Type de simulation	type1

FIGURE 16 – Image des paramètres du logiciel

l'ajout d'un bouton de remise à zéro a été nécessaire pour nettoyer le tableau des ouragans.

#### 5.4 Utilisation de patrons de conception

“Un patron de conception est la meilleure solution connue à un problème de conception récurrent.” -M. Cédric Bouhours.

Le patron Observateur :

Le problème récurrent est comment faire pour notifier un changement d'état d'un objet\* à un autre objet alors qu'ils ne se connaissent pas ou qu'ils n'ont pas accès à l'autre.

Par exemple dans la fenêtre principale, on a deux objets qu'on veut connecter :  
 Objet 1 : Mon tableau contenant les informations sur les ouragans.  
 Objet 2 : La carte contenant la représentation graphique des ouragans.  
 Les deux objets sont contenues dans l'objet mainwindows, mais ils ne se connaissent pas, pourtant il faudrait que quand l'un est modifié par l'utilisateur, l'autre se modifie. Si l'utilisateur bouge un ouragan sur la carte avec sa souris, il faut mettre à jour le tableau, et inversement, si le tableau est modifié, il faut mettre à jour l'ouragan sur la carte.  
 C'est là que Qt intervient, il redéfinit le type objet, les QObject ont des Slots et des Signals en plus de leurs attributs et de leurs méthodes. Les Slots sont des méthodes qui peuvent être connectées à des signaux (Signals). Une fois connecté, quand un objet émet un signal, alors tous les objets qui sont connectés à ce signal reçoivent l'information et activent leur slot associé.

Le patron singleton :

Le problème récurrent est comment faire pour que plusieurs objets différents contiennent des données communes.

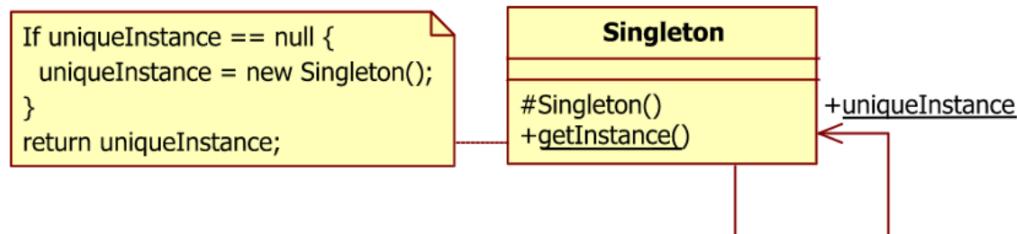


FIGURE 17 – Diagramme de classe d'un singleton

Dans le projet, on a les objets Position pour chaque ouragans, ils contiennent tous un objet Construction. Cet objet Construction charge les données pratiques comme les matrices de Markov.

Afin d'éviter le chargement de ces données à chaque création d'ouragan, on applique le patron de conception Singleton à la classe Construction.

La classe Construction devient donc une "classe globale", elle ne peut exister qu'une seule et unique instance\* et elle ne charge donc les données qu'une seule fois. Et tous les objets Position contiendront cette même instance\* de Construction et se la partageront.

## III. Résultats et discussions

### 1. Onglet de lancement de simulation

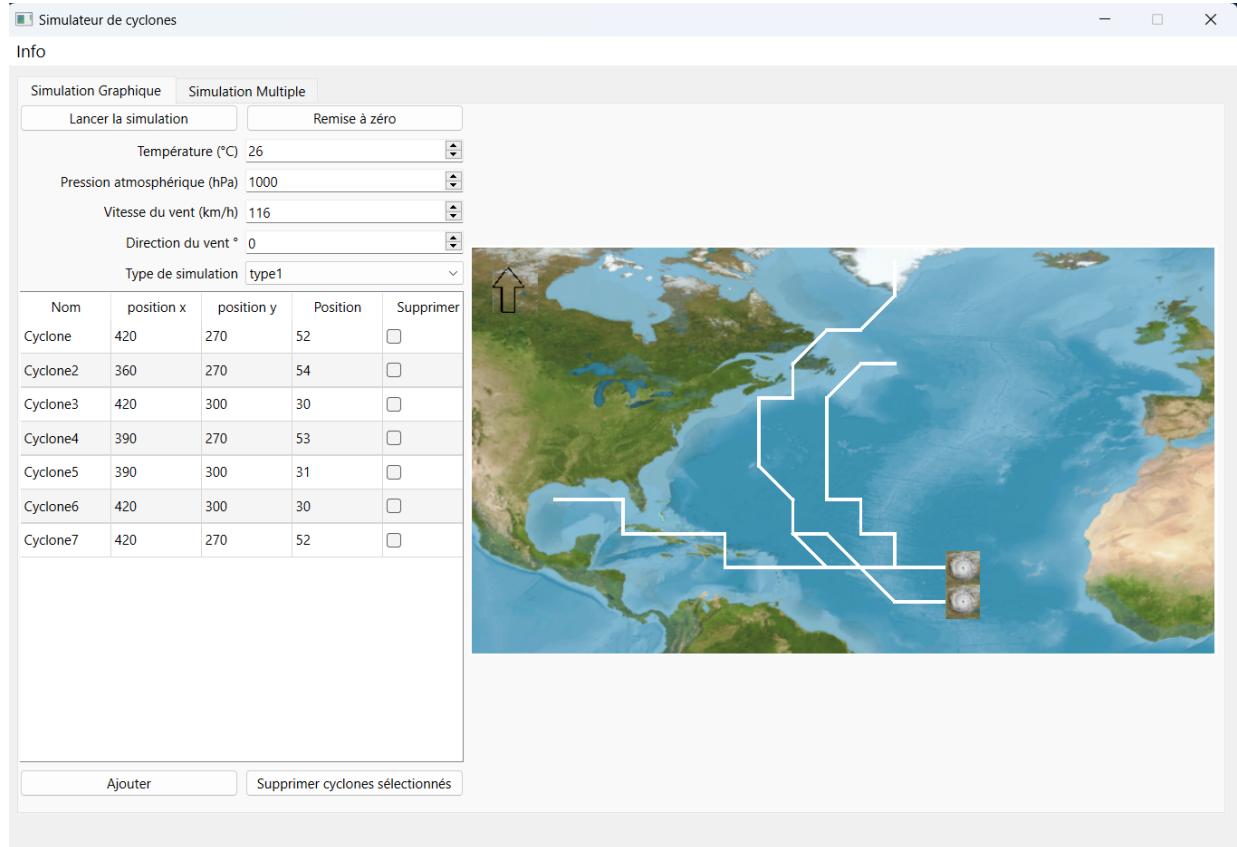


FIGURE 18 – Onglet graphique du logiciel avec différents ouragans modélisés

On voit sur la figure ci-dessus un aperçu de la fenêtre de simulation, dans la simulation ci-dessus on a lancé la modélisation pour 7 ouragans simultanément.

Le résultat se rapproche bien des plans de départ, avec le bouton Lancer la simulation, les champs pour laisser le choix des paramètres à l'utilisateur. Le tableau d'ouragan avec ses boutons ajouter et supprimer. Ce tableau est lié à la carte à droite, l'utilisateur peut glisser déposer les ouragans sur la carte et cela mettra à jour le tableau, et inversement. Tous les ouragans sont indépendants.

Le tracé blanc est la trajectoire des ouragans, leur point de départ étant là où les ouragans sont placé actuellement. Si l'on veut comparer avec les trajectoires réelles il est possible de comparer avec les informations sur cette figure :

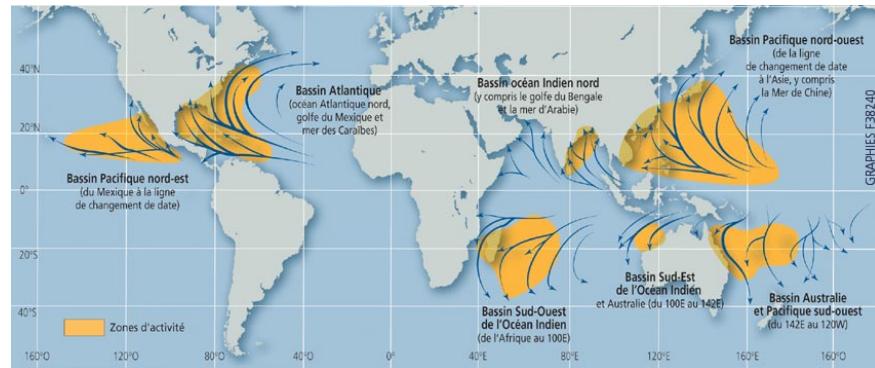


FIGURE 19 – Directions et trajectoires des ouragans

Le résultat obtenu semble assez convaincant, nos ouragans vont eux aussi adopter cette trajectoire arrondie qui va monter et revenir vers la longitude de départ. Pour venir appuyer cela on peut aussi observer la trajectoire sur le continent :

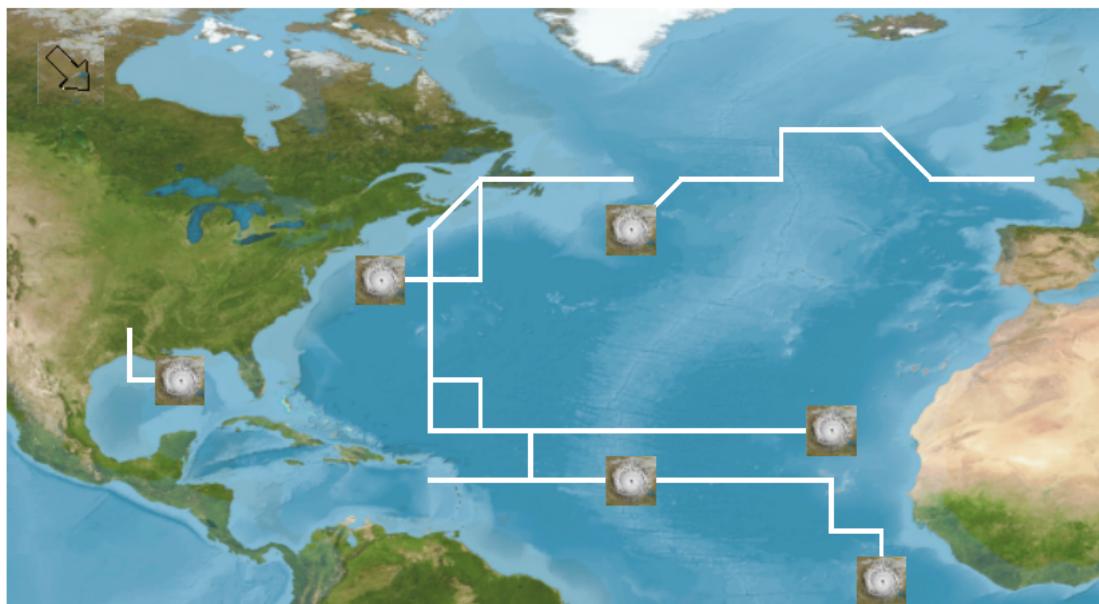


FIGURE 20 – Résultats d'ouragans partant de différents endroits

En effet on voit bien le fait que les ouragans «meurent» vite quand ils arrivent au niveau des terres comme le montre également la figure 19.

On peut aussi voir le résultat obtenu avec un grand nombre d'ouragans qui partent du même endroit. (cf annexe 5 : plusieurs simulations du même point de départ).

Avec l'interface graphique on peut néanmoins dire que la trajectoire n'est pas lisse du tout, cela est du à un manque de précision et pourrait être pallié avec un lissage des trajectoires.

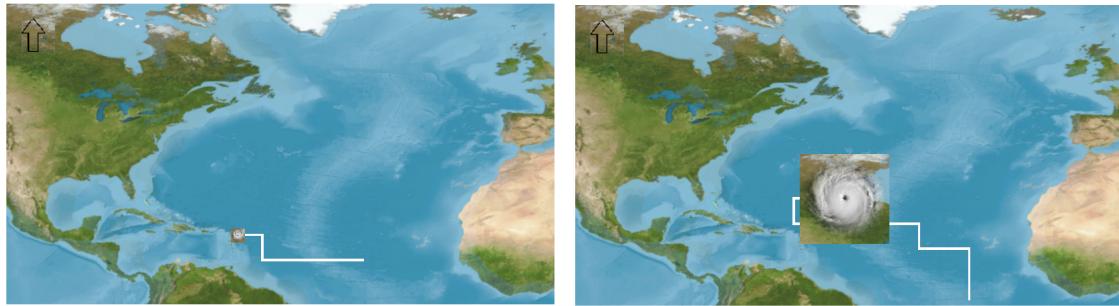


FIGURE 21 – Visualisations au cours d'une simulation

On peut voir sur la figure ci-dessus deux visualisations de deux ouragans différents durant leur simulation. Leur rayon change bien, cependant il faut noter que l'échelle du rayon n'est pas fidèle à la réalité, la taille affichée a été choisie de manière à bien voir les changements (par exemple l'ouragan de droite ne fait en réalité pas la taille d'un pays).

## 2. Onglet de simulation multiple

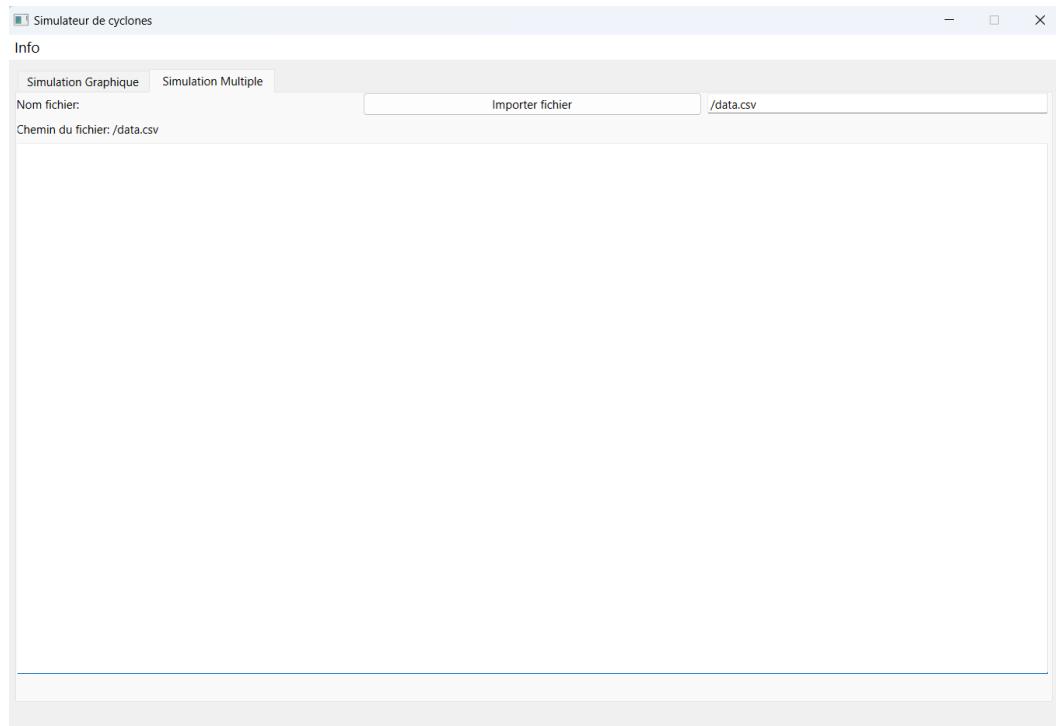


FIGURE 22 – Fenêtre avec l'onglet de simulation multiple

On peut voir l'apparence de l'onglet de simulation multiple même si ce dernier n'est pas fonctionnel.

Sur l'interface graphique nous avons encore mention de cyclone, or nous travaillons bien sur des ouragans (nous l'avons décidé plus tard), cela peut donc être une piste d'amélioration.

## Conclusion

Notre objectif était de réaliser un logiciel permettant de simuler l'évolution d'un ouragan à partir d'un point donné avec des paramètres fixés. Le logiciel devait comprendre un modèle réaliste de simulation de déplacement et de taille des ouragans. Il devait aussi contenir une interface utilisateur simple et efficace.

Nous avons eu l'occasion de développer en Qt, une surcouche du langage C++ qui offre des possibilités intéressantes pour la programmation logicielle. Nous avons pu acquérir de nouvelles connaissances à propos des chaînes de Markov et de la modélisation de phénomènes physiques en général.

L'étape de modélisation a été plus longue que prévu, ce qui a engendré des retards sur l'implémentation du modèle dans le logiciel. Donc le choix de se concentrer sur l'onglet principal a été fait. L'onglet de simulation multiple n'est donc pas fonctionnel.

L'objectif du projet est tout de même atteint, le logiciel est fonctionnel et on retrouve un comportement des ouragans cohérent. Cependant il reste des perspectives d'améliorations. Comme avoir un modèle avec plus de précision et linéariser la trajectoire, ou avoir l'onglet de simulation multiple fonctionnel pour pouvoir étudier les données simulées. Ainsi que des options d'affichages comme des couleurs différentes par ouragan ou le choix du traçage. De plus on pourrait améliorer le rayon en appliquant la solution de le calculer une fois l'ouragan déjà modélisé.

On peut aussi ajouter que l'augmentation des phénomènes météorologiques tel que les cyclones, les ouragans et les typhons donnent de l'importance à leur modélisation. En effet les modéliser peut permettre de prévenir les populations et dans des cas extrêmes de les évacuer. Un domaine notable d'application de simulateur d'ouragan est le domaine des assurances, les assurances veulent savoir quel risque est associé à chaque zone.

## Références Bibliographiques

- [1] A.-Y. LeRoux, M.-N. LeRoux, J.-A. Marti « Un modèle mathématique de cyclone » Compte Rendu de mathématiques de la mécanique, Académie des Sciences de Paris, 2004 [Online]. Disponible sur : [https://www.sciencedirect.com/science/article/pii/S1631073X04002730?ref=pdf\\_download&fr=RR-2&rr=7a2218ca3ae93627](https://www.sciencedirect.com/science/article/pii/S1631073X04002730?ref=pdf_download&fr=RR-2&rr=7a2218ca3ae93627) [Consulté le : 28-fév-2023]
- [2] S. Jolivet « Modélisation mésoéchelle des cyclones tropicaux dans le Sud-Ouest de l’Océan Indien avec Méso-NH » Thèse de doctorat de Physique de l’atmosphère, Université de la Réunion, Réunion, 2008 [Online]. Disponible sur : <https://theses.hal.science/tel-00462517/document> [Consulté le : 01-fév-2023]
- [3] C. Denise-Baillon « Modélisation statistique du risque de tempêtes tropicales dans le bassin Atlantique Nord » Mémoire de maîtrise de Statistiques et Finances ,Université de Lyon, Lyon, 2013 [Online]. Disponible sur : [http://www.ressources-actuarielles.net/EXT/ISFA/1226-02.nsf/0/59a9a2b1746f19c5c1257c72004b99eb/\\$FILE/Me%CC%81moire%20DENISE-BAILLON%20Christophe.pdf](http://www.ressources-actuarielles.net/EXT/ISFA/1226-02.nsf/0/59a9a2b1746f19c5c1257c72004b99eb/$FILE/Me%CC%81moire%20DENISE-BAILLON%20Christophe.pdf)
- [4] © 2023 The Qt Company« Qt documentation » Documentation de Qt [Online]. Disponible sur : <https://doc.qt.io/> [Consulté le : 01-mar-2023]

## Lexique

Cisaillement du vent : Le cisaillement du vent est une différence de la vitesse ou de la direction du vent entre deux points suffisamment proches dans l'atmosphère.

Classe : En programmation orientée objet, une classe déclare des propriétés communes à un ensemble d'objets. La classe déclare des attributs représentant l'état des objets et des méthodes représentant leur comportement. Une classe représente donc une catégorie d'objets. Elle apparaît aussi comme un moule ou une usine à partir de laquelle il est possible de créer des objets.

Composantes d'un vecteur : Les composantes d'un vecteur sont les déplacements selon un premier axe et selon un deuxième axe entre son origine et son extrémité (cf annexe 6 : Composantes d'un vecteur) (la composante du vecteur AB selon l'axe x c'est a et la composante selon l'axe y c'est b).

Énergie Cinétique : L'énergie cinétique est l'énergie que possède un corps du fait de son mouvement.

Gradient de pression : Le gradient de pression est la quantité utilisée en mécanique pour représenter la variation de la pression dans un fluide.

Instance : Une instance d'objet ou de classe est la mise en mémoire d'un individu de la classe ou du type objet. Il peut y avoir plusieurs instances d'une même classe.

Layout : Un layout est un objet graphique qui permet de disposer les éléments qu'il contient, par exemple en empilant plusieurs boutons.

Objet : Un objet est la représentation du comportement d'un individu d'une classe. Par exemple on peut imaginer qu'un objet de type Lapin peut avoir des attributs comme une couleur, un âge ou bien une taille, et des actions comme manger des carottes ou se reproduire.

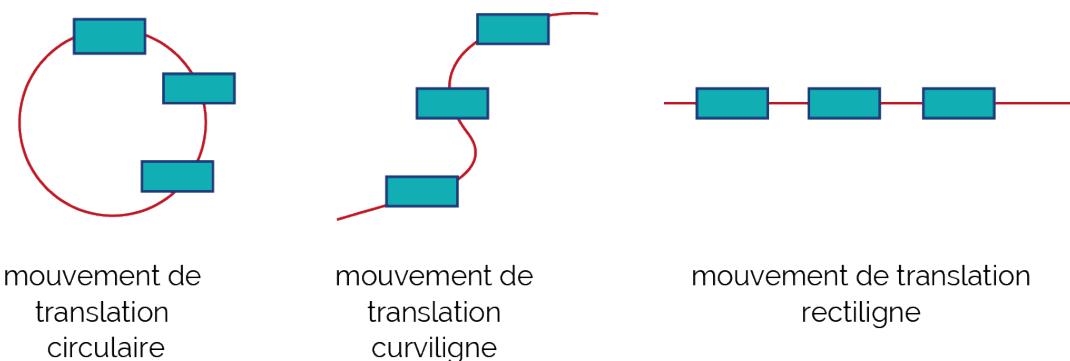
Perturbation : Évènement qui crée une irrégularité dans le fonctionnement d'un système.

Template : Un template est une représentation graphique du produit attendu.

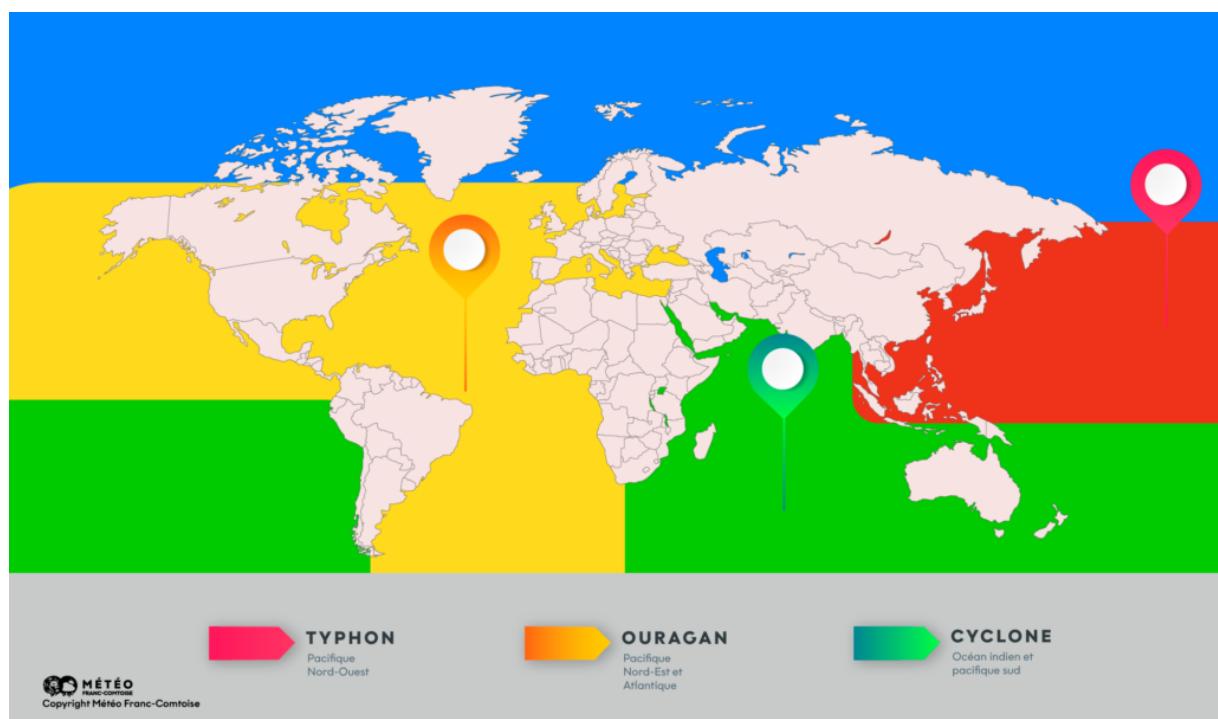
Thermodynamique : Branche de la physique qui étudie les relations entre phénomènes thermiques (qui impliquent des modifications de température) et mécaniques.

## Annexes

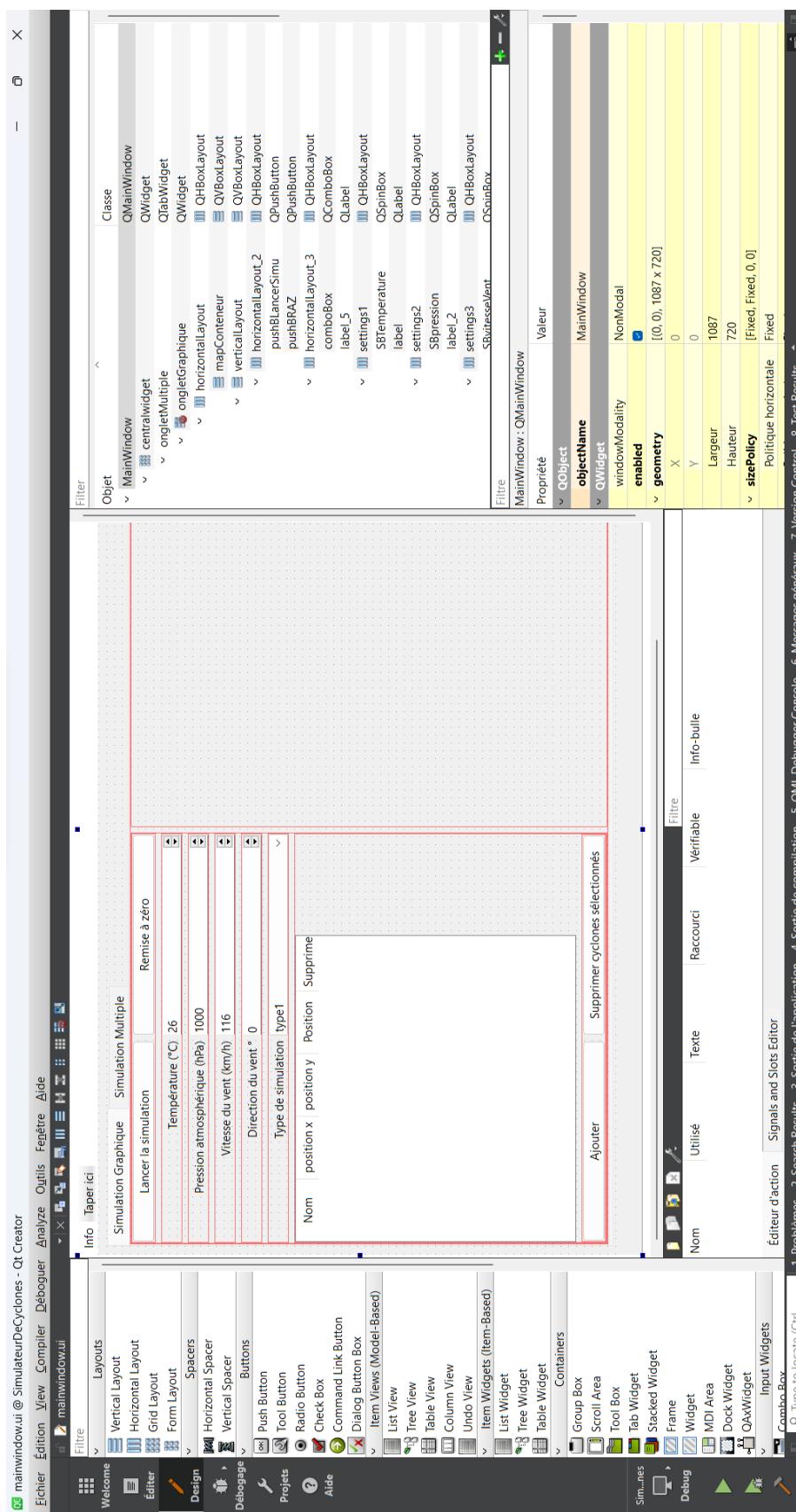
### Annexe 1 : Représentation des mouvements de translation



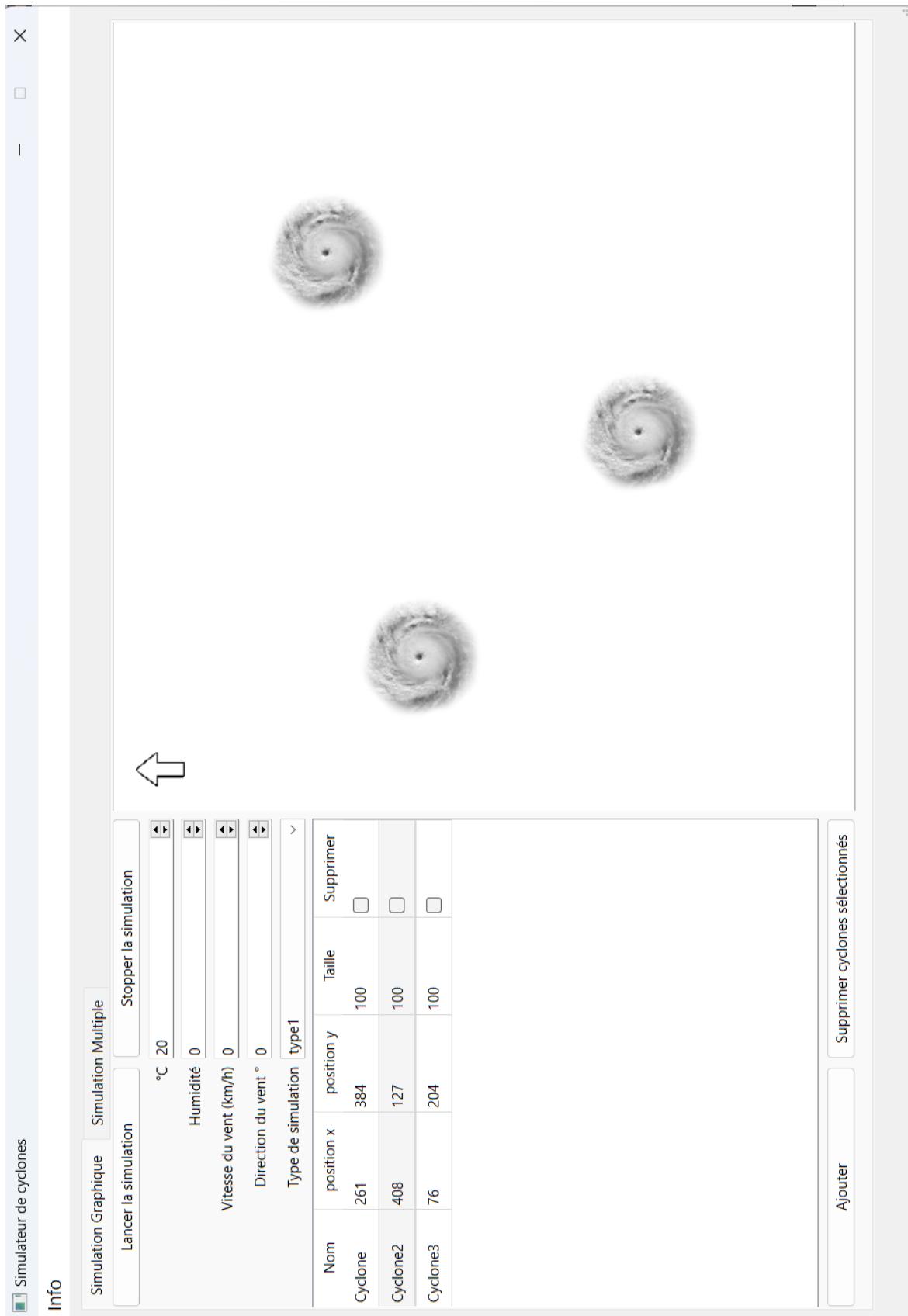
### Annexe 2 : Nom du phénomène selon le lieu



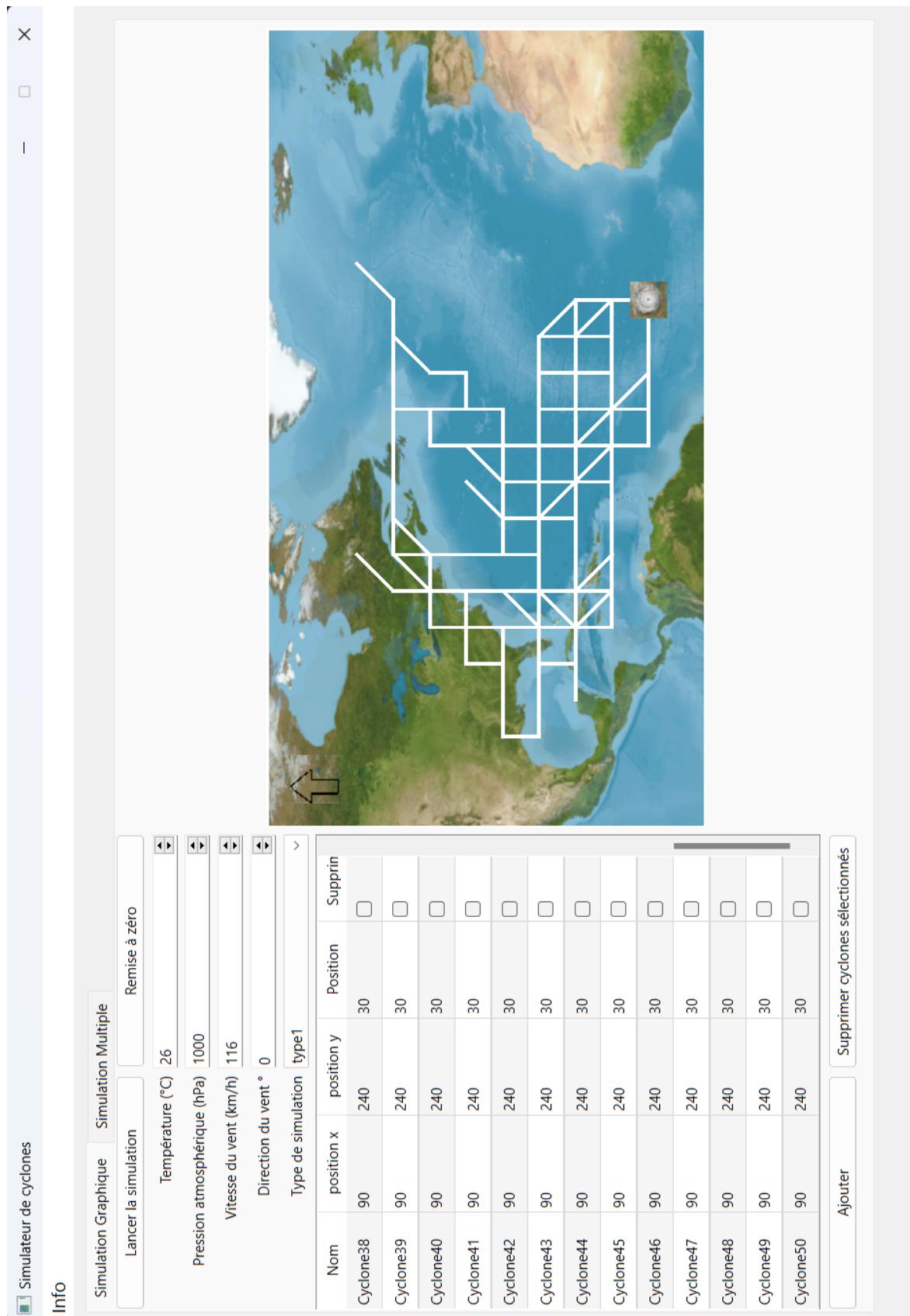
### Annexe 3 : Onglet design de Qt Creator



## Annexe 4 : Premier visuel du logiciel



## Annexe 5 : plusieurs simulations du même point de départ



## Annexe 6 : Composantes d'un vecteur

