LES TROPHÉES NSI

Édition 2023

DOSSIER DE CANDIDATURE
PRÉSENTATION DU PROJET

Ce document est l'un des livrables à fournir lors du dépôt de votre projet : 4 pages maximum (hors documentation).

Pour accéder à la liste complète des éléments à fournir, consultez la page **Préparer votre participation**.

Vous avez des questions sur le concours ? Vous souhaitez des informations complémentaires pour déposer un projet ? Contactez-nous à **info@trophees-nsi.fr**.

NOM DU PROJET : Fire Aspect

> PRÉSENTATION GÉNÉRALE :

Le projet Fire Aspect consiste à simuler la propagation d'un incendie sur une zone donnée grâce à une capture d'écran d'une carte et un algorithme codé en Python en prenant en paramètre la vitesse, la force du vent et l'indice de percolation de l'environnement.

> ORGANISATION DU TRAVAIL :

Notre équipe est composée de quatre membres : BEC-CLEMENTE Elio, MILLET-LEITE ANTUNES Mattéo, SEILLER Mathieu et MERCIER Yohan. La répartition des tâches durant le déroulement de la création du projet a été attribuée de la manière suivante :

- BEC-CLEMENTE Elio : Réalisation de la partie graphique grâce au module Python TKINTER.
- MILLET-LEITE ANTUNES Mattéo : Réalisation du système de propagation du feu.
- SEILLER Mathieu : Réalisation de la rose des vents.
- *MERCIER Yohan : Réalisation du processus de pixellisation de la carte Map Quest.*

LES ÉTAPES DU PROJET :

La programmation de la simulation s'est déroulée en quatre parties :

1. Tout d'abord, nous avons procédé à la pixellisation de la carte. Pour cela, nous avons séparé l'image en carrés de cinq pixels de côté. Ensuite, nous avons regardé la couleur majoritaire de chaque carré pour définir la nature de celui-ci : bleu pour l'eau, vert pour la végétation et blanc pour les zones urbaines.
2. Deuxièmement, nous avons affiché la carte à l'aide du module tkinter. Ensuite, nous avons ajouté tous les boutons pour enlever les cases bleues et vertes. Nous avons également affiché la rose des vents et tracé les lignes entre le centre et l'endroit où nous cliquons.
3. Troisièmement, nous avons simulé la propagation de l'incendie en calculant la probabilité que chaque case autour d'une case en feu prenne feu. Nous avons également mis en place une fonctionnalité permettant de déclencher

un feu en cliquant sur une case de la carte, qui apparaît alors en rouge. Lorsque l'incendie évolue, nous pouvons observer cette évolution en temps réel sur la carte grâce à l'utilisation de tkinter.

4. Quatrièmement, nous avons finalisé la Rose des vents en la faisant fonctionner : en augmentant la probabilité de propagation de l'incendie dans la direction sélectionnée et en diminuant la probabilité de propagation de l'incendie dans la direction opposée en fonction de la distance entre le point de clic et le centre de la rose.

> FONCTIONNEMENT ET OPÉRATIONNALITÉ :

Le projet est parfaitement conforme aux attentes du début même si le temps nous a rattrapé, nous aurions aimé rajouter plus de paramètres météorologiques, de données sur l'état du sol et sur l'incendie en lui-même pour affiner la précision de l'indice de percolation car ces paramètres contribuent à l'influence de la propagation.

Tout d'abord, ce projet a été codé en Python et de façon Orienté Objet.
Pour procéder, nous avons chargé différents modules :

- Pour les modules externes (installé à partir d'internet) : PIL et tkinter mais on n'utilisera seulement les objets Image et ImageTk pour le module PIL.
- Pour les modules que nous avons créés : plan et RoseDesVents

Ensuite, on instancie l'objet Tk du module tkinter, ce qui va créer la fenêtre dans laquelle on pourra créer le modèle, puis on instancie l'objet Plan du module Plan, ce qui va créer la zone où l'on pourra manipuler le plan, et l'objet Rose du module RoseDesVents pour créer la zone où la direction et la vitesse du vent pourront être affectés.
Nous créons ensuite des boutons réalisant les actions suivantes :

- Quitter : Ferme la fenêtre tkinter à l'aide de la méthode 'destroy' appliqué à l'objet instancié auparavant : fen
- Evolution : Fait avancer les carrés rouges d'une case en fonction des indices de percolations qui sont calculés à partir de la direction du vent.
- Isoler les Bleus / Verts : Supprime les cases Bleues / Vertes étant parasite.
- Réinitialisation du Plan : Remet le Plan dans son état initial.
- Réinitialisation de la Rose des Vents : Supprime les traits placés sur la Rose des Vents.

Pour obtenir le plan de façon pixéliser, nous avons diviser l'image (qui est une capture d'écran du site 'https://www.mapquest.com/') en cases de 5x5 pixels auxquelles nous avons attribués les valeurs suivantes : 'blue' , 'green' , 'None'.

Ces valeurs ont été attribués en fonction de la moyenne des couleurs des pixels présents dans une cases. Ainsi, nous avons pu déterminer si une zone était une zone boisée (‘green’), une zone d’eau (‘blue’) ou une zone urbaine (‘None’). De plus, pour une meilleure visibilité, nous avons décidé de recolorer les cases des zones boisées et des zones d’eau.

Nous avons décidé le site Map Quest pour sélectionner notre plan car sur les cartes de ce site, il y a beaucoup moins d’éléments parasite tel que les noms des routes ou les icônes.

Le programme a été fait de façon à ce que l’incendie arrête de se propager s’il rencontre de l’eau (cases bleues) et se propage de manière plus importante lorsqu’il rencontre une zone boisée (cases vertes). De plus, le feu se propagera dans la direction du vent sélectionner sur la Rose des Vents (Si une direction est sélectionnée, car ce n’est pas obligatoire).
Pour calculer, la chance qu’une case à proximité brûle, on calcule l’indice de percolation dans la direction de cette case. Pour calculer cet indice de percolation, on relève la direction du vent (l’angle sur la Rose des Vents) et on calcule l’indice de percolation (grâce à la fonction : exp(-0.022x) ; où x est l’angle entre la direction du vent et la direction dont on cherche l’indice de percolation) dans chacune des directions possibles (Si aucune direction n’est sélectionnée, les indices de percolations dans chacune des directions sont par défaut paramétré sur 40% de chance). La force du vent est aussi calculé en fonction de la longueur du tracé fait sur la Rose des Vents.

Enfin, pour utiliser le programme, il suffit simplement de placer un ou plusieurs départs de feu en cliquant sur le plan, de choisir ou non une direction pour le vent et de cliquer sur le bouton ‘Evolution’.

> OUVERTURE :

Le projet fonctionne très bien, cependant, nous pourrions le rendre encore plus précis et plus utile pour comprendre et trouver des solutions pour limiter la propagation des incendies :

1. Ajouter plus de précision et de différences sur l'environnement, car pour l’instant nous avons seulement la végétation, l’eau et le milieu urbain. Nous pourrions, par exemple, définir plus précisément la végétation pour différencier la forêt plus ou moins dense, l’herbe, etc.
2. Ajouter des événements qui modifieraient le comportement de la propagation de l’incendie, tels que la pluie qui diminuerait la propagation, la sécheresse qui augmenterait la propagation, ou encore l’orage qui ajouterait des départs de feu aléatoirement sur la carte.

3. Pouvoir modifier la carte en ajoutant des cases d’eau, de végétation ou de zone urbaine pour trouver comment limiter la propagation.

Enfin, pour diffuser notre projet, nous pourrions le rendre disponible en ligne, en créant une interface utilisateur pour que les utilisateurs puissent facilement utiliser notre programme sans avoir à installer Python. Nous pourrions également ajouter des fonctionnalités de visualisation des données, telles que des graphiques pour représenter les taux de propagation de l'incendie en fonction des différents paramètres.

DOCUMENTATION

Dans toute la documentation on considère qu'on a importé les modules suivants :

«from PIL import Image,ImageTk
import tkinter as tk
import plan as pln
import RoseDesVents as rs
from Carre import *»

Objet Plan du module plan :

Objet permettant de contrôler toute la partie du plan

Attributs :

image : méthode open de l'objet Image du module PIL, Identification de l'image importé

__fen : tk.Tk, Instance de l'objet Tk() fenêtre Tkinter

tailleCarre : int, Taille en pixel d'un carré de la grille du plan

imageTkinter : tk.ImageTk.PhotoImage, conversion de l'attribut image pour la manipuler avec tkinter

canvas : tk.Canvas, instance de l'objet Canvas du module tkinter, identification du canvas du plan

grille : list, Liste d'objets Carre modélisant la subdivision du plan en carré

positionCouleur : dict, clé = couleur et valeur = lise de coordonnées des cases qui ont été colorés sous forme d'un tupple

roseVent : rs.Rose, Instance de l'objet Rose

Méthodes :

__creationCanvas : Creation du canvas et affichage de l'image Retourne l'objet Canvas du plan

__creationGrille : Creation de la grille qui découpe le plan en plus gros pixel

Retourne cette même grille sous forme de tableau d'objets

remplissageCouleurs : Remplissage de couleur dans la grille tkinter et la grille sous forme de tableau a deux dimensions

Parametres:

caseX:int, coordonnees x de la case

caseY:int, coordonnees y de la case

couleur:str,couleur a appliquer pour le remplissage

__supprimerCase : Supprime dans l'algorithme et sur le plan la case de coordonnees : (caseX,caseY)

Parametres:

caseX:int, coordonnees x de la case

caseY:int, coordonnees y de la case

__colorationDesPixels : Choisie si un pixel doit etre colore en vert,en bleu ou ne pas etre colore

affinnerBleu : Supprime les pixels bleus parasite/seuls

affinnerVert : Supprime les pixels verts parasite/seuls

evolution : Permet de faire évoluer l'incendie en fonction du vent et du terrain

effacerTout : Permet de reinitialiser le plan (supprimer l'incendie)

Objet Rose du module RoseDesVents:
Rose des vents pour ajouter du vent a la simulation

Attributs :

__fen : Instance de l'objet Tk(), fenêtre Tkinter

image : Image.open, Identification de l'image importée

imageTkinter : tk.ImageTk.PhotoImage, conversion de l'attribut image pour la manipuler avec tkinter

canvas : tk.Canvas, instance de l'objet Canvas du module tkinter, identification du canvas de la Rose des vents

trait : tk.Canvas, instance de l'objet Canvas du module tkinter, identification du canvas du trait rouge

indicesPercolation : dict, clé = direction du vent ; valeur = indice de percolation, direction du vent = les deux premiers caractères representent le cosinus arrondi a 1, les deux derniers le sinus arrondi a 1

angleDuVent : float, Angle du vent

vitesseVent : float, distance entre le milieu et les coordonnées cliqués sur la rose

Méthodes :

__creationCanvas : Creation du canvas et affichage de l'image

Retourne l'objet Canvas du plan

__effacerTrace : Efface le tracé sur la Rose

trace : Trace un trait entre le milieu et le point de coordnees x,y

Parametres:

x:int,Numero de la colone du pixel

y:int,Numero de la ligne du pixel

toutEffacer : Efface en entier le canvas qui contient la Rose des Vents et remet a la valeur initiale les variables

distanceMilieu : Calcul de la distance entre le milieu et le point de coordnees x,y

Parametres:

x:int,Numero de la colone du pixel

y:int,Numero de la ligne du pixel

Retourne la distance

angleVent : Clacul d'angle du vent par le biais d'un cercle trigonometrique

Parametres:

x:int,Numero de la colone du pixel

y:int,Numero de la ligne du pixel
Retourne l'angle du vent en degre

calculPercolation : Calcul de l'indice de percolation en fonction du vent
Parametres:
x:Coordonnees x du pixel concerne
y:Coordonnees y du pixel concerne

Objet Carre :
Objet permettant de creer graphiquement les cases sur le plan
Attributs :
canvas : tk.Canvas, Canvas du carré
couleur : str, couleur du carre

Le fichier main.py permet de lancer le programme et de vérifier si l'utilisateur clique sur l'un des canvas