DAUPHINE
UNIVERSITÉ PARIS

INTELLIGENCE ARTIFICIELLE

Approximer une image

Etudiants:
Shahram Mahdavi
Samir Amrani

2018/2019

Rapport du Projet :Approximer une image

Intelligence Artificielle

I. Introduction

LE but de notre projet est de réussir à recréer une image donnée à partir de polygones convexes colorés et opaques, notamment l'image du célèbre tableau de Léonard de Vinci, la Joconde. Nos recherches au problème posé nous ont amené à trouver que nous devions utiliser un algorithme génétique. Son fonctionnement s'inspire de la génétique humaine et de l'évolution de la nature en se basant sur les principes suivants : croisement/mutation/sélection pour se rapprocher de plus en plus de la solution finale.

II. Notre Démarche

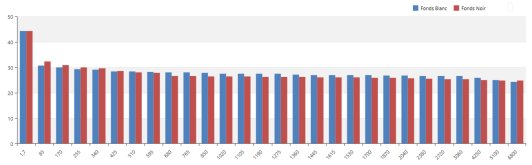
En première tentative nous avons travaillé sans croisement mais en utilisant uniquement le principe de mutation et sélection. Nous avions initié un tableau d'individu de taille 2 avec une image de 50 polygones générés au hasard chacun. Puis nous avons écrit une fonction de mutation qui pouvait muter un polygone choisis au hasard comme : changer un paramètre de sa couleur / changer son opacité / ajouter un point / enlever un point / modifier un point du polygone / le supprimer et en ajouter un nouveau générer aléatoirement / en permuter 2 au hasard etc. Ensuite, nous avons appliqué la fonction de fitness au deux images et nous clonions celle qui avait le fitness la plus basse. Puis nous recommencions le processus.
Nous avions obtenu comme résultat une descente très rapide de la fitness de l'image à environs 43/42. Nous avons laissé tourner le programme toute la nuit mais la fitness restait convergente à ce niveau-là.
Nous avons ensuite réessayé par exemple en utilisant un tableau de beaucoup plus grande taille mais le résultat était à peu près similaire.
Après avoir fait plusieurs autres tentatives de code nous avons finalement aboutis à la solution suivante :
- Créer une population d'individu, chaque individu est constitué d'une liste de polygones ,de son image et de sa fitness.
- initialiser la population à 100 individus généré au hasard.
- Créer une nouvelle liste ≪ listri ≫ qui trie la population d'Individu en fonction de leur score de fitness par ordre croissant.
- On prend dans ≪ listri ≫ deux Individus parent au hasard dans les 30 premières.
- On les croise et on génère deux fils et on prend celui avec un meilleur fitness.
- On insère le fils dans une position entre 30 et 100.
- on mute la liste et on garde la mutation que si elle est meilleur .
- On tri a nouveau la liste puis on affiche la meilleure fitness
- On recommence le même processus n fois .

III. Differentes Executions

A. le tableau ci-dessous presente differentes executions avec differents parametres.

Nbr de génération	50 polygon et le fonds blanc		50 polygon et le fonds noir	
	Le temps(en s)	La fitness	Le temps(en s)	La fitness
1	1.7	44.31	1.7	44.31
50	85	30.65	85	32.24
100	170	29.95	170	30.94
150	255	29.24	255	30.05
200	340	29.02	340	29.57
250	425	28.43	425	28.59
300	510	28.37	510	28.11
350	595	28.23	595	27.88
400	680	28.09	680	26.62
450	765	28.02	765	26.52
500	850	27.93	850	26.46
600	1020	27.5	1020	26.38
650	1105	27.49	1105	26.37
700	1190	27.46	1190	26.32
750	1275	27.46	1275	26.31
800	1360	27.07	1360	26.29
850	1445	27.02	1445	26.07
900	1530	27.01	1530	26.07
950	1615	27.01	1615	26.01
1000	1700	26.93	1700	25.89
1100	1870	26.8	1870	25.81
1200	2040	26.76	2040	25.74
1400	2380	26.69	2380	25.49
1600	2720	26.66	2720	25.33
1800	3060	26.58	3060	25.3
2500	4250	25.87	4250	24.92
3000	5100	25.02	5100	24.87
4000	6800	24.32	6800	24.79

B. le temps d'execution et la difference entre fond noir et fond blanc



.

IV. Observations et Analyse

-Notre première version n'était pas optimal car les mutations faites étaient trop forte. De ce fait , l'image avait fini par rejeter toutes les mutations car elle avait déjà convergé en grande partie. Les mutations ne changeaient pas assez la photo pour avoir un meilleur score de fitness. C'est ceci qui nous a poussé à introduire les croisements entre individu.

- Nous avons créé une classe Population composée d'une ArrayList d'Individu. Nous croisions 2 Individus au hasard en prenant les 25 premiers et 25 derniers de chaque. Puis nous mutons l'individu créé.Nous avions remplacé la fonction de mutation par juste changer la couleur ou remplacer un polygone par un nouveau car nous avions trouvé cela suffisant. Le programme fonctionnait et la fitness baissait jusqu'à 43 en 25min puis s'était stabilisée pendant plus de 4h.

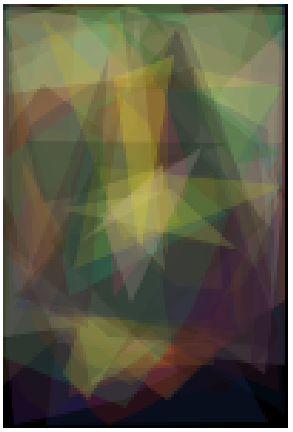
- Nous avons alors modifié la méthode de croisement en prenant un random au lieu de 25 polygones de chaque parent. De plus, au lieu de générer un fils, nous générions deux fils en même temps puis nous choisissions le meilleur(pour correspondre au mieux au principe de l'algorithme génétique). Puis nous comparions le meilleur fils avec ses deux parents. Si sa fitness était meilleure , nous remplacions le parent. Puis nous mutions toute la liste d'individu et nous gardions la mutation que si elle était meilleure. Nous avions obtenu alors de meilleurs résultats. Le score de fitness baissait à 34 en 35min puis décroissait de 0.1 chaque 10 à 20 min , puis s'était stabilisé à 30-31 pendant plus de 4h. Le problème était que notre population n'était pas assez diversifié. Les individus étaient trop similaires entre eux. Cela était dû au fait que nous ne gardions pas les parents.

- Du coup l'image n'était pas encore satisfaisante. Il fallait encore modifier le programme.

Nous prenons alors 2 individus au hasard parmi les 30 meilleurs Individu pour générer toujours un fils ce que nous avons changé est qu'aprés la génération des fils nous ajoutons directement le fils aprés les 30 meilleur individu puis retrions la liste et affichons l'image de la meilleur fitness.
le score de fitness alors baissait jusqu'a 26 en 20 min puis décroissait lentement.
Aussi nous avons améliorer l'algorithme en ajoutant deux autres differantes methodes de croisement, et chaque fois que le score de fitness se stabilise plus de 250 fois sur une valeur on génére 70 nouveau individu en gardant les 30 meilleur.
l'algorithme foctionne et le score de fitness baisse a 25 en 15 a 20 min.

V. image obtenu

ci-dessous on a l'image obtenu avec un score de fitness 26



VI. Ressenti du projet

La première difficulté a été de trouver comment commencer car nous n'avions aucune notion d'algorithme génétique dans le cours.
Ensuite, nous pouvons cité la compréhension du code donné. En effet , nous n'étions pas familier de javaFX. Comprendre nos erreurs nous ont pris un certain temps.
De plus , même en ayant réussis à résoudre les erreurs de compilations , il fallait attendre un certains temps avant de voir si notre code aboutissait à l'image souhaité ou si la fonction de fitness finissait par converger sans aboutir à une image acceptable et comprendre pourquoi dans ce dernier cas. Ce qui impliquait de devoir réfléchir à nouveau à un nouvel algorithme.

REFERENCES

https://stackoverrun.com/fr/q/1930238
https://fr.wikipedia.org/wiki/Algorithme_gntique
https://sausheong.github.io/posts/
a-gentle-introduction-to-genetic-algorithms/
https://o7planning.org/en/11157/
javafx-transformations-tutorial
http://tech.nitoyon.com/en/blog/2009/
02/17/50-polygons-mona-lisa-in-as3/
https://skyduino.wordpress.com/2015/07/16/
tutorielpython-les-algorithmes-genetiques-garantis-sans-ogm/
https://khayyam.developpez.com/articles/algo/
genetic/ http://home.iitk.ac.in/~aniketmt/Project

VII. ANNEXE

A. *les differentes images de l'execution*