Cartoonisation d'image

Samuel Helye, Sylvain Leclerc

29 Novembre 2022

1 Cartoonisation par méthodes classiques

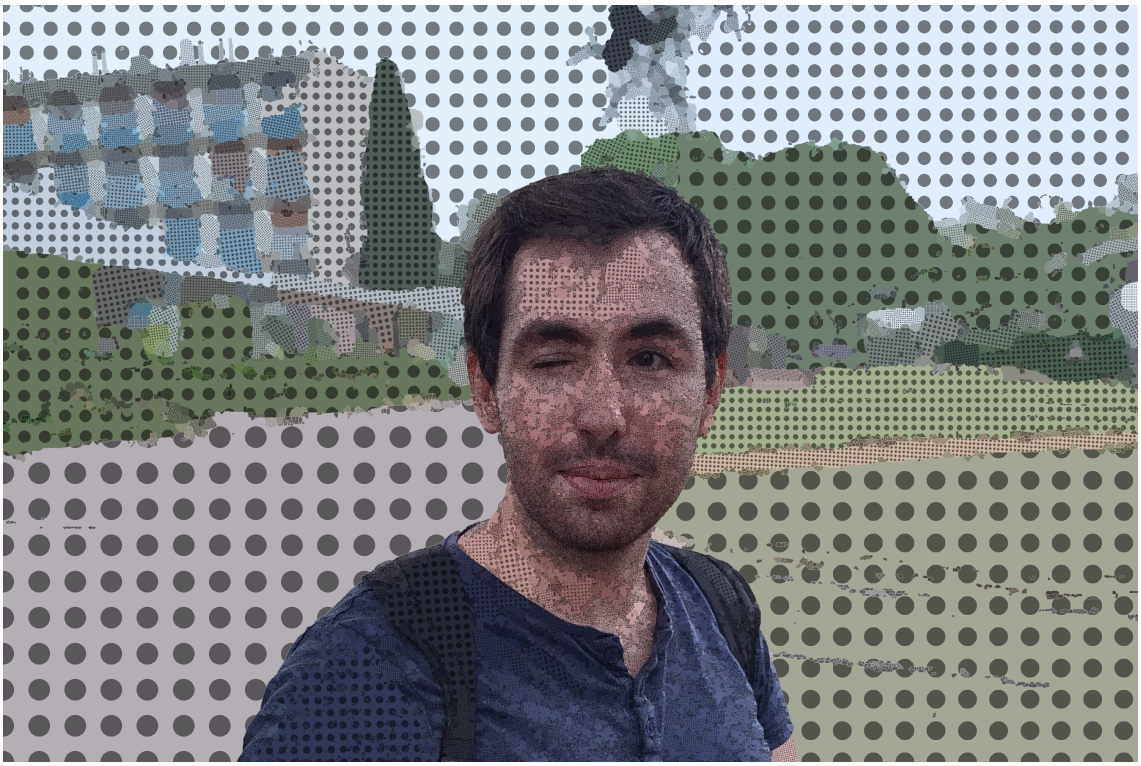
1.1 Création de gifs

Cette semaine, nous avons appliqué notre méthode de cartoonisation classique à la cartoonisation de gifs.
Nous avons pour se faire écrit un script bash, celui-ci aura plusieurs objectifs.
Tout d'abord, il doit séparer le gif d'entrée en un ensemble d'images, puis il remplace la transparence (qui n'est pas prise en compte par notre programme) par du noir. Ensuite, il lance la cartoonisation de l'ensemble des images, le script lance plusieurs cartoonisation à la fois, pour profiter des différents cœur du processeur, il ne lance pas toutes les cartoonisation en même temps pour ne pas saturer la RAM. Nous avons défini la limite expérimentalement, sur nos machines, à 16 processus en parallèles.
Enfin, les images sont assemblées pour recréer un gif.

On ne peut pas integrer un gif dans un PDF, vous pouvez en voir sur notre git.

1.2 Création de texture

Après avoir rempli les zones avec des dégradés, nous explorons une méthode de remplissage par textures générées procéduralement.



Ces textures consistent en une trame point, la taille et l'écartement des point dépendent le la taille de la zone. La direction dépend de la direction du gradient, et la couleur dépend de la couleur de la zone.
Par la suite, il sera intéressant de déterminer différemment ces paramètres, notamment la couleur des points.

2 Generation d'image par IA

Cette semaine a été surtout consacrée pour de la recherche (en partie du à des problèmes technique de mise en place de l'environnement Python pour pouvoir exploiter un réseau de neuronne de segmentation plus pertinent). Nous avons notamment retenu un article faisant l'état de l'art des différentes technique de Deep learning pour la segmentation sémantique [3]. Il énumère les différents modèles et datasets avec des points de comparaison pour donner un maximum d'outil théorique pour notre choix. Cela soulève une question qu'il nous faut encore examiner, les datasets sont présentés avec un Objectif (purpose) et nous offre donc différentes possibilités pour différents résultats. Par exemple, le dataset Stanford 2D-3D-S [4] offre un jeu de donnée pour la segmentation d'intérieur, il reconnait donc les murs, chaises, bureau, etc. Cela peut être un choix intéressant, comme pour les datasets plutôt orienté "Urbain" (Urban driving), c'est une pure question de choix. Dans l'idéal il serait intéressant de pouvoir exploiter tout ses datasets et faire un choix en fonction de l'image d'entrée, mais étant limité par le temps nous sommes encore en train de décider vers où nous nous orientons.

Références

[1] Bharath K. Object Detection Algorithms and Libraries (July 2022).

[2] OpenAI. Language Models are Few-Shot Learners (July 2020)

[3] A Review on Deep Learning Techniques Applied to Semantic Segmentation (April 2017)

[4] Joint 2D-3D-Semantic Data for Indoor Scene Understanding (April 2017)