Création d'images par algorithme génétique avec référence

Rapport de projet

Lucien Cartier-Tilet

March 24, 2019

Contents

1 Sujet

Le sujet de ce projet est la création d'un logiciel pouvant recréer une image fournie grâce à des générations aléatoires et successives de formes aux, positions, couleurs et taille aléatoires. L'algorithme commence par créer une image vide aux dimensions identiques à l'image de référence, puis applique une de ces formes aléatoires. Si la ressemblance de l'image ainsi générée augmente par rapport à sa version précédente par rapport à l'image de référence, alors cette modification est conservée, sinon elle est annulée. Répéter jusqu'à satisfaction.

2 Les méthodes utilisées

Plusieurs approches au problème sont possibles, allant de la simple implémentation naïve du problème à des moyen pouvant au moins décupler la vitesse de génération de l'image. Sauf indication contraire, j'ai utilisé dans l'implémentation de chaque méthode des carrés comme forme d'éléments appliqués aléatoirement à l'image.

Pour évaluer la ressemblance entre deux image, j'évalue une distance euclidienne entre le vecteur de leurs pixels qui peut se résumer à ceci :

$$\sqrt{\sum_{i=0}^{n} V_i^2 + W_i^2}$$

`V` étant le vecteur de pixels de l'image de référence, `W` étant le vecteur de pixels de l'image générée, et `n` la taille de ces deux vecteurs.

Les tests de temps sont réalisés sur un Thinkpad x220, disposant d'un processeur Intel® Core™ i5-2540M à 2.6GHz, composé de deux cœurs supportant chacun deux threads, et de 4Go de RAM. Le programme est compilé avec les options d'optimisation `-O3` et `-flto`.

Voici également ci-dessous la liste des options et arguments possibles concernant l'exécution du logiciel.

```
$ ./bin/genetic-image -h
Allowed options:
  -h [ --help ]           Display this help message
  -i [ --input ] arg      Input image
  -o [ --output ] arg     Image or video output path (default: input path +
                          "_output")
  -m [ --method ] arg     Method number to be used (default: 1)
  -n [ --iterations ] arg Number of iterations (default: 5000)
  -v [ --video ]          Enable video output
```

2.1 Méthode naïve

J'ai tout d'abord implémenté la méthode naïve afin d'avoir une référence en matière de temps. Cette dernière est implémentée dans `src/methods.cc` avec la fonction `method1()`. Comme ce à quoi je m'attendais, cette méthode de génération d'images est très lente, principalement dû au fait que l'algorithme en l'état essaiera d'appliquer des couleurs n'existant pas dans l'image de référence, voire complètement à l'opposées de la palette de couleurs de l'image de référence.

Voici la ligne de commande utilisée depuis le répertoire `build` afin de pouvoir obtenir un temps d'exécution :

```
perf stat -r nombreDExécutions -B ./bin/genetic-image \
     -i ../img/mahakala-monochrome.jpg -o output.png -n 200 -m 1
```

nombre d'itérations réussies	nombre d'exécutions	temps d'exécution moyen
10	100	0.09447s ($\pm$0.02%)
50	100	
100	50	
200	20	
500	10	
1000	5	