Université de Bordeaux

Traitement d'image Avancée

Assignment 1

Texture Synthesis

Author:
Jimmy Gouraud

Supervisor:
Aurélie Bugeau

October 5, 2017

1 Introduction

Nous allons voir et implémenter la méthode de synthèse de texture par "Non-parametric Sampling" d'Alexei A. Efros et de Thomas K. Leung. Cette méthode permet de synthétiser une texture pixel par pixel en analysant le voisinage local du pixel, à l'aide de l'utilisation des patchs.

2 Texture Synthesis

2.1 Algorithme de synthèse de texture

L'algorithme se découpe en 3 étapes :

1. Initialisation de l'image de destination à partir d'un bout de la texture (appelé «seed»)
2. Recherches des pixels à traiter
3. Traitement des pixels trouvés

```
Data: texture, dest_image, patch_size
Result: dest_image fill
fill_seed(texture, dest_image);
while dest_image not filling do
    pixels = find_pixels_unfilled(dest_image);
    for pixel in pixels do
        best_patch = find_best_patch(pixel, dest_image, texture,
         patch_size);
        pixel = best_patch.pixel;
    end
end
```

Algorithm 1: Pseudo Code - Texture Synthesis

Lors de l'initialisation, on va créer une image de destination de la taille désirée, puis on va "coller" en son centre une "seed", c'est-à-dire un bloc de pixels de 3×3 issue de la texture initiale (fonction fill_seed(texture, dest_image)) que l'on va prendre aléatoirement.

Ensuite, tant que l'image de destination n'est pas remplit entièrement, on va chercher les pixels à traiter et les traiter.

2.2 Recherche des pixels à traiter

Pour trouver les pixels à traiter, on va, lors de l'initialisation, créer un tableau 2D appelé "masque" de la taille de l'image de destination que l'on va initialiser à 0. On va ensuite remplir les pixels correspond à la seed à 1. Ainsi, dans lorsque l'on cherchera les pixels à traiter, il suffira de faire une dilatation de ce masque et de faire la différence avec le masque précédent (voire figure 1).

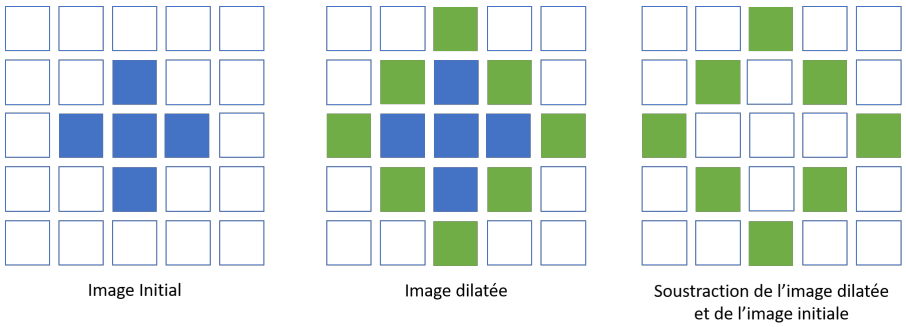

Figure 1: Dilatation et soustraction

On pourrait améliorer le procédé en maximisant le nombre de voisins pour chaque pixel par exemple.

2.3 Traitement des pixels

Pour traiter un pixel, nous allons analyser son voisinage à l'aide des patchs et essayer de trouver un environnement le plus proche possible dans notre texture initiale. Pour comparer deux patchs entre eux, nous allons avoir recours à la SSD (*Sum of Squared Differences*). La SSD dans notre cas se fera sur les canaux RGB. On aurait pu de plus faire passer un noyau de convolution gaussien pour avantager les pixels proche du centre, mais les résultats ne sont pas forcément meilleurs.
Après avoir parcouru l'ensemble des patchs contenue dans notre texture et les avoir comparé avec notre patch centré autour du pixel que l'on est en train de traiter, on sélectionne le meilleur patch. Le meilleur patch est simplement celui avec lequel on obtient la plus petite SSD.
Il suffit ensuite de recopier le pixel du centre du patch sélectionné, dans le pixel de notre image de destination.

Afin d'obtenir plus de diversité dans notre génération de texture, on pourrait sélectionner aléatoirement un patch parmi les 5 meilleurs patchs par exemple, parmi tous les patchs qui possède une $SSD < min(SSD) * 1.1$.

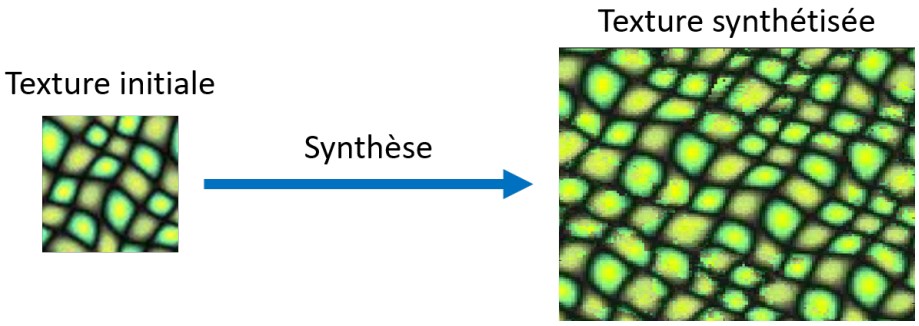

Figure 2: Synthèse de Texture

2.4 Modification pour faire du *in painting*"

La méthode d'*in painting* permet de combler les occlusions d'une image et/ou de supprimer des éléments indésirables (comme par exemple une personne sur une photo).
Le procédé pour faire du "in painting" est exactement le même que pour la synthèse de texture. On va simplement passer l'image que l'on souhaite retoucher, un masque de la taille de l'image avec comme valeur 1 les pixels que l'on souhaite garder et 0 ce que l'on souhaite traiter, et la taille d'un patch. Ensuite, au lieu de rechercher les patchs dans notre texture comme on faisait pour la méthode *texture synthesis*, on va rechercher les patchs contenu directement dans notre image de destination. Ainsi on va ainsi pouvoir remplir les occlusions identifiés par le masque.

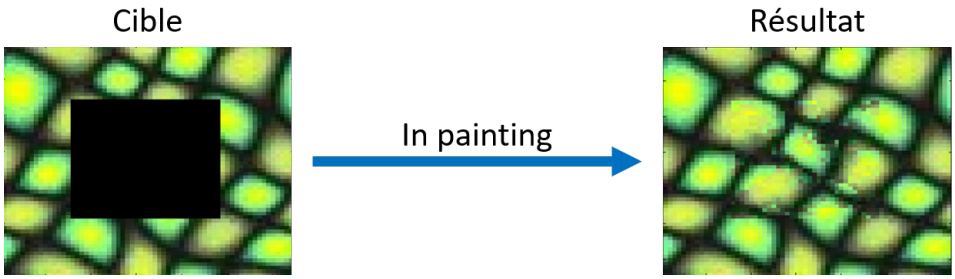

Figure 3: In painting

3 Conclusion

Pour obtenir des résultats cohérents, c'est-à-dire une texture à la fois différente mais visuellement ressemblante, il faut réussir à trouver une bonne taille de patch. Plus le patch est grand, plus la texture synthétisée ressemblera à la texture initiale, mais en contrepartie, cela entraînera du bruit dû à une plus grande diversité dans le choix du meilleur patch et donc provoquera du bruit.[1] [2]

References

[1] Antonio Criminisi, Patrick Perez, and Kentaro Toyama. Region filling and object removal by exemplar-based image inpainting. September 2004.

[2] Alexei A. Efros and Thomas K. Leung. Texture synthesis by non-parametric sampling. *IEEE International Conference on Computer Vision*, September 1999.