Master SETI - UE Vision Robotique

TP 1 - Seam Carving

THOMAS BOULANGER JULES FARNAULT

24 NOVEMBRE 2024

Table des matières

Introduction

L'objectif de ce TP est d'implémenter un algorithme de seam carving, c'est à dire un algorithme de réduction de taille d'image qui minimise la déformation dû au changement de taille. Le TP est composé de deux parties avec une première implémentation basé sur les graph cuts. Dans un second temps, l'algorithme sera amélioré pour prendre en compte les objets d'intérêt afin de limiter leur déformation.

1 | Seam carving basique

Dans cette partie, on implémente l'algorithme développé par Rubinstein et al., 2008. L'implémentation de l'algorithme est découpé plusieurs étapes :

- Conversion de l'image RGB en une image en niveau de gris
- Calcul du gradient de l'image
- Calcul de la couture (seam) optimal selon les lignes et les colonnes
- Suppression de la couture selon soit de la ligne, soit de la colonne selon un critère d'optimisation
- Itération jusqu'à l'obtention de la taille voulue

1.1 - Calcul de l'énergie

Pour le calcul de l'énergie de l'image, nous allons utiliser le gradient au niveau de chaque pixel. La norme du gradient donnera l'énergie de ce pixel.

On commence par implémenter un filtre de Sobel avec une matrice de dimension 3. Les résultats ne sont pas très concluants. En effet, il a fallu mettre en place la gestion des bords, car sinon l'énergie minimale était toujours à cet endroit. De plus, la couture semble rester "collé" aux bords ce qui rend l'algorithme très peu performant.

On cherche donc à améliorer cette partie en utilisant le filtre Laplacien, toujours en dimension 3. Nous passerons la matrice en valeur absolue pour avoir une énergie. Les effets de bords disparaissent et les coutures se font bien au cœur de l'image en évitant les objets d'intérêts.

Le filtre Laplacien semble meilleur dans le calcul de l'énergie, le filtre de Sobel semble rester toujours proche du bord même en le forçant à s'en détacher. Dans la suite, nous utiliserons donc un filtre Laplacien.

1.2 - Calcul de la couture

Le but est de trouver un chemin dans le sens suivant les lignes ou les colonnes qui minimisent la suppression d'énergie.

Pour ce faire, on implémente un algorithme de type Graph Cut/ Max Flow avec d'abord une matrice accumulatrice de l'énergie puis, avec un backtracking, on remonte cette matrice accumulatrice afin de trouver la couture minimisant globalement l'énergie.

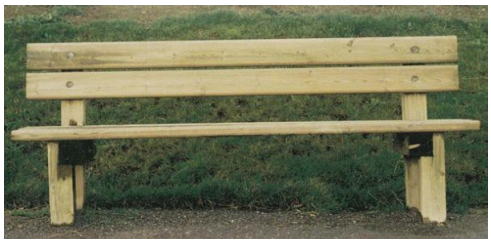
Les résultats sont positifs, mais un problème subsiste : comment choisir si l'on doit supprimer une ligne ou une colonne ?

1.3 - Suppression de la couture et itération

Ainsi, nous avons deux chemins possibles, l'un dans le sens des lignes, l'autre dans le sens des colonnes. Pour choisir, on prend le chemin qui minimise l'énergie normalisée. Cette normalisation permet de s'abstraire du ratio de l'image qui biaiserait le choix.

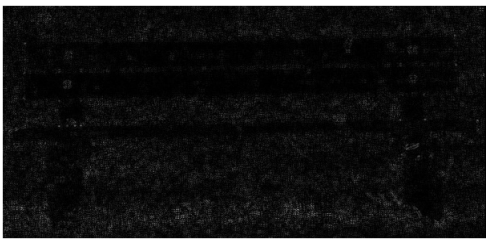
On procède avec cette méthode de façon itérative. On choisira au début du programme une taille voulu, et on réduira sur les lignes et les colonnes selon le critère décrit précédemment. De plus, on ajoute un blocage permettant de ne par réduire plus que la taille voulut.

1.4 - Résultats

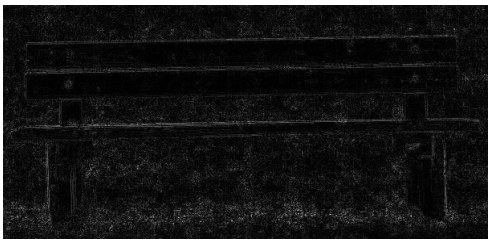


(a) Image d'origine (290x604)

(b) Passage en niveau de gris



(c) Filtre de Sobel



(d) Filtre Laplacien

(e) Coutures suivant les lignes et les colonnes (en rouge)

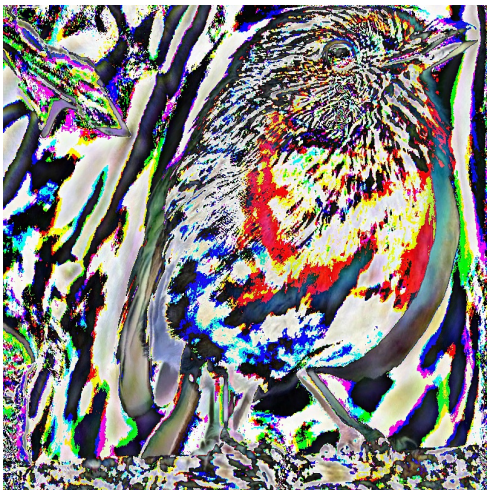
(f) Image finale (200x200) (Laplacien)

FIGURE 1.1 – Différentes étapes de l'algorithme

(a) Image d'origine (1920x1080)

(b) Image finale avec Sobel (800x800)

(c) Image finale avec le Laplacien (800x800)



(d) Image de la différence de l'image finale avec le filtre de Sobel et le filtre Laplacien (800x800)

On remarque que le filtre à un gros impact sur l'image finale. L'image résultant du filtre laplacien est moins déformée. On choisira d'utiliser ce filtre dans la suite.

(a) Image d'origine (230x307)

(b) Image finale avec le Laplacien (150x150)

FIGURE 1.3 – Application du seam carving basique à plusieurs images

Comme on peut le constater, l'algorithme fonctionne bien, mais il peut grandement altérer les objets d'intérêts (jambe de la fille sur la plage). C'est pourquoi l'on va modifier notre algorithme afin qu'il évite les objets d'intérêts.

2 | Seam carving amélioré

Dans la première partie, nous arrivons à réduire notre image, mais nous avons tendance à modifier fortement les zones importantes de l'image (personne, animaux...). Nous allons donc chercher à réduire cela.

Pour cela, nous allons mettre en place une segmentation qui va différentier les zones de l'image. Nous choisirons ensuite les clusters que l'on veut garder le plus possible et nous augmenterons considérablement leur énergie afin de les rendre moins intéressant à supprimer.

Si on se retrouve avec trois pixels avec une énergie maximale, on choisira de regarder les pixels en i-2 et i+2, en ainsi de suite jusqu'à ne plus avoir de maximum.

On utilise un algorithme de k-means pour segmenter l'image. Pour des raisons de temps de calcul, on utilisera seulement l'image de la plage qui est dans un petit format. On va essayer de ne pas déformer l'enfant et le pigeon.

(a) Image d'origine

(b) Image segmentée avec K-means

(c) Résultat sans la segmentation

(d) Résultat avec segmentation

On définit le cluster noir comme étant le cluster important. Pour chaque pixel de ce cluster, on va augmenter très fortement son énergie dans la matrice énergie. Cela aura pour effet de ne pas rendre optimale la couture qui passe par ses pixels.

Le résultat est plutôt bon, on modifie moins l'image quand on segmente l'image.

(a) Image banc segmentée avec K-means (4 clusters)

(b) Image oiseau segmentée avec K-means (2 clusters)

On remarque que notre segmentation n'est pas très bonne et ne permet pas de réduire l'image. Cela vient du fait que les zone que l'on ne veut pas réduire ne sont pas séparable :

— l'oiseau à plusieurs couleurs qui sont les couleurs de l'image

— le banc prend toute l'image et le but est de réduire le banc donc ne correspond pas au besoin

Ainsi, il n'est pas toujours utile de segmenter l'image pour la réduire.

Conclusion

Lors de ce TP, nous avons implémenter deux algorithmes de seam carving permettant de réduire la taille d'une image en minimisant la déformation des objets d'intérêts à l'aide des graph cuts. Le premier algorithme, bien que basique, permet déjà de réduire des images en gardant les objets d'intérêts. Cependant il coupe parfois ces objets d'intérêts ce qui introduit des déformations indésirable. C'est pourquoi l'on a implémenté un second algorithme faisant de la segmentation afin d'identifier les objets d'intérêts en amont du seam carving ce qui permet de limiter la découpe de ces objets et ainsi d'amélioré les résultats.