Rapport de Projet
Traitement de nuage de points

RANSAC

Jean Fechter
Thibault Boutet

Promotion Epita Image 2024
Janvier 2024

1 Ransac

1.1 Détéction d'un plan unique

L'algorithme de Ransac (RANdom SAmple Consensus) nous permet d'identifier un plan dans un nuage de point 3D. Le principle de l'algorithme est de déterminer plusieurs plans puis de conserver celui qui obtient le meilleur concensus. Pour obtenir de bon résultats, il est importants d'ajuster les paramètres suivants :

— n : Le nombre d'itérations de l'algorithme (le nombre de plan que l'on va calculer)

— ϵ : La distance qui définit si un point apparatient au plan trouvé

Une fois que l'on a obtenu le plan avec le plus du points du nuage 3D, on peut colorier tous les points du plan pour l'isoler visuellement.

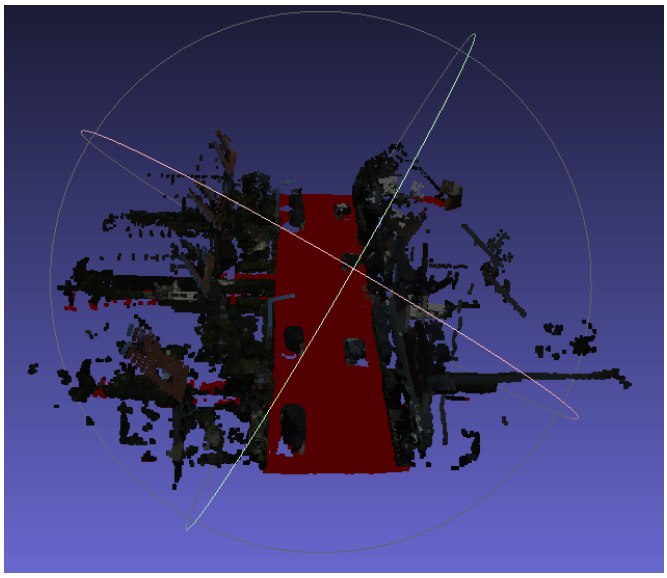

FIGURE 1 – Isolation de la route du modèle *road_small.obj*

Le résultat précédent à été obtenu avec n=100 et ϵ=0.15. On remarque que le plan de la route est correctement isolé du reste du nuage de points. Nous avons rafiné les paramètres afin d'obtenir le meilleur résutlat pour la route sans "déborder" sur le reste des points du nuage. Intuitivement, on comprends que le plan sélectionné soit la route car il s'agit de la plus grande concentration de points. Les probabilités de selectionné un point initial appartenant au plan de la route sont très élévées. C'est pourquoi nous avons choisis une valeur pour n assez faible.

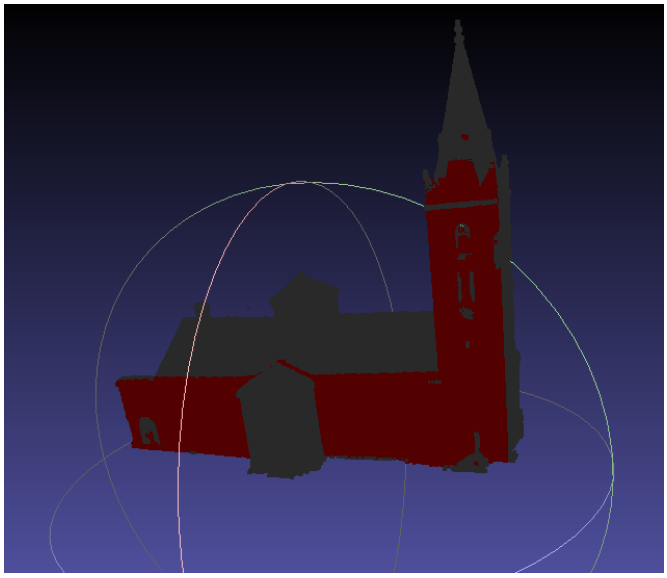

FIGURE 2 – Isolation d'un des murs principaux sur *church.obj*

Le résultat précédent à été obtenu avec n=2000 et ϵ=0.3. En applicant notre algorithme sur le modèle *church.obj*, on va isoler l'un des murs principaux du batiments. En effets le murs lateraux sont des plans avec la plus grosses concentration de points du nuage.

1.2 Détéction de plusieurs plans

Plusieurs étapes ont été nécessaires afin d'adapter notre premier algorithme pour détecter plusieurs plans.

Dans un premier temps, le nombre de plans est fixé. La seule information à rajouter est de garder trace des points qui ont déjà été assignés à un plan, afin de ne pas les réutiliser dans un autre. Pour cela, nous appelons N fois notre fonction RANSAC qui détecte un plan unique, en retirant à chaque appel les points conservés par le plan précédent.

Ensuite, afin de ne pas sélectionner arbitrairement le nombre de plans, nous avons ajouté un critère d'arrêt. Ce dernier est assez simple, l'algorithme s'arrête lorsqu'il reste moins d'un pourcentage des points originaux (par exemple 50%).

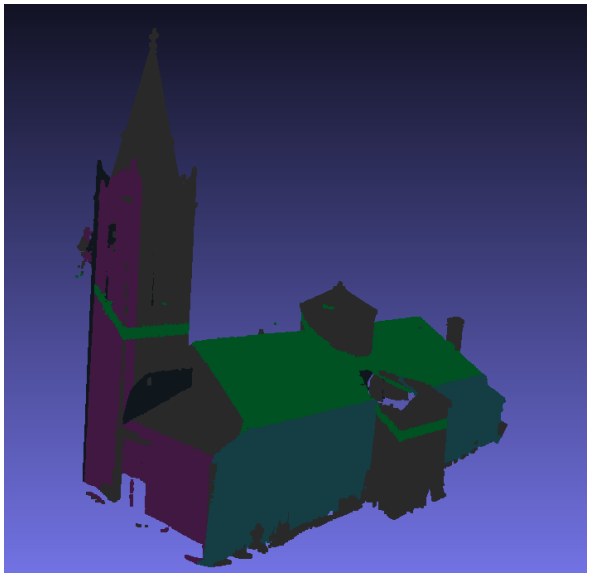

FIGURE 3 – Détéction de plusieurs plans sur la donnée church.obj

On remarque que le résultat pourrait être amélioré. En effet, jusqu'à présent, tous les points appartenant au plan infini choisi lui sont attribués, sans prendre en compte d'autres paramètres. Par exemple dans notre cas, l'objectif est de segmenter des plans de batiments, ici une église, dont on souhaite isoler les murs, faces du toit, etc... Cependant on peut voir que la méthode actuelle pose problème puisque le plan vert sur l'image possède des points qui ne devraient pas lui appartenir.
Nous avons donc besoin d'utiliser plus d'information que simplement la position des points.

2 Extentions

2.1 Filtre des normales

Une des améliorations possibles est l'utilisation des normales des points afin de prendre en compte l'orientation de ces derniers.
L'idée est de ne conserver que les points dont la normale est orientée dans la même direction que celle du plan étudié, en accordant une légère tolérance (par exemple 45 degrés).
Grâce à cet ajout, les points n'ayant pas la même orientation que le plan choisi, et n'étant donc probablement pas sur le même plan réel, ne seront pas conservés.

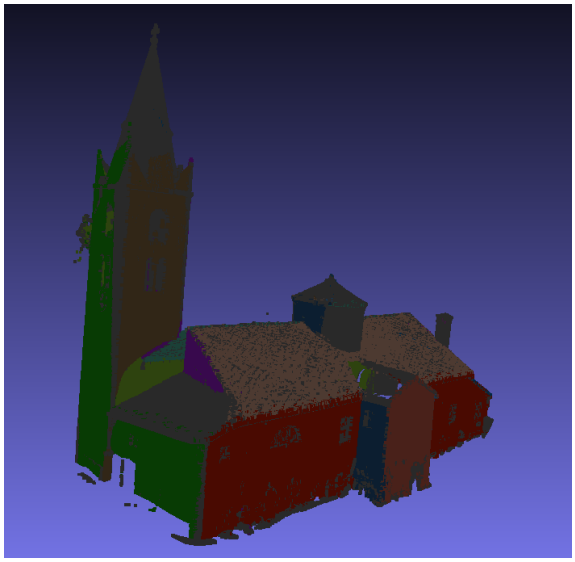

FIGURE 4 – Détéction de plusieurs plans sur la donnée church.obj en prenant en compte les normales des points

On peut voir que les plans semblent mieux segmenter que précédemment, cependant certains points appartenant bien au plan ne sont plus colorés.
Deux explications sont possibles :

— Le seuil de tolérance de l'orientation des normales est trop restrictif.

— Certaines normales sont mal définies dans l'objet original.

2.2 Accroissement de régions

Une autre amélioration possible est d'ajouter un algorithme d'acroissement de région après RANSAC, afin d'ajouter les points manquants au plan et de ne conserver qu'un ensemble de points connexes.
En utilisant le KD-tree, nous avons essayé de l'implémenter en parcourant récursivement l'ensemble des voisins du point de départ dans un rayon suffisamment proche. Cependant nous n'avons pas réussi à le faire fonctionner (voir code commenté).

3 Conclusion

Nous avons réussi à implémenter l'algorithme du RANSAC afin de détecter un plan dans un nuage de points 3D. Puis, nous avons étendu notre algorithme pour isoler les plans principaux d'un nuage. Enfin, nous avons ajouté des extensions à notre projet pour le rendre plus performant. Cependant, nous nous sommes tout de même confrontés à des difficultés. Particulièrement au niveau de l'implémentation de l'accroissement de région. La piste d'amélioration principale du projet serait d'obtenir une version fonctionnelle de la méthode d'accroissement dans le but d'améliorer les résultats. De plus, nous avons remarqué que les temps d'éxécution de nos algorithmes sont plutôt long. Il serait pertinent de paralléliser notre code.