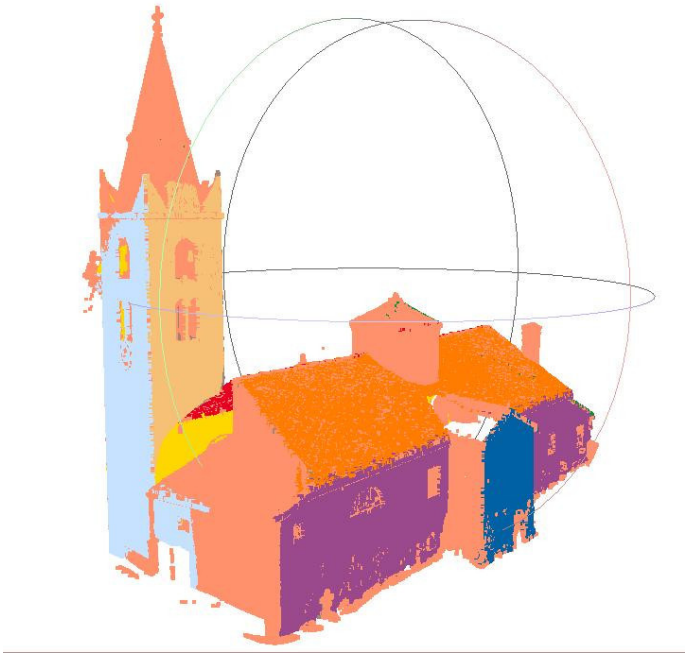
Algorithme RANSAC en 3D



Adrien ANTON LUDWIG Adèle PLUQUET

December 17, 2022

Table des matières

1 Introduction

L'objectif dans se travail est de mettre en place une première approche de segmentation en utilisant des primitives simples (plans, sphères, cônes, ...). Nous essayons ensuite de recaler ces primitives au sein du nuage de point afin de pouvoir segmenter des objets comme une route ou des façades (plans), des toits (cônes) ou autres.

2 Détection de plans

La détection de plans va nous permette de segmenter les routes et façades par exemple. Nous allons effectuer le recalage des primitives au sein du nuage de points grâce à l'algorithme RANSAC (RANdom SAmple Consensus). Cet algorithme est, comme son nom l'indique, aléatoire. Voici les étapes suivies pour cet algorithme :

- Répéter
  - On sélectionne aléatoirement trois points dans le nuage. Ces trois points permettent de définir un plan.
  - On calcule pour chaque point du nuage de points sa distance au plan. Si sa distance est inférieure à un certain **threshold**, alors on considère ce point comme un **inlier**. Dans le cas inverse, c'est un **outlier**.
  - On compare le nombre d'**inliers** de ce plan avec le maximum enregistré. Si nous avons plus d'**inliers** que le maximum enregistré, alors le maximum est remplacé par notre nombre d'**inliers** et on enregistre le plan actuel.

Une fois la version pour un plan fonctionnelle, nous l'avons améliorée pour détecter plusieurs plans. La base de l'algorithme utilisé est l'algorithme décrit précédemment. Cependant, pour chaque plan détecté, nous supprimons les **inliers** associés à ce plan du nuage de points. Ainsi, à la prochaine itération, c'est un plan différent qui sera trouvé. L'algorithme est répété tant que les plans trouvés comportent assez de points afin qu'ils restent pertinents et que la valeur donnée par le paramètre **max_objects** n'est pas dépassée.

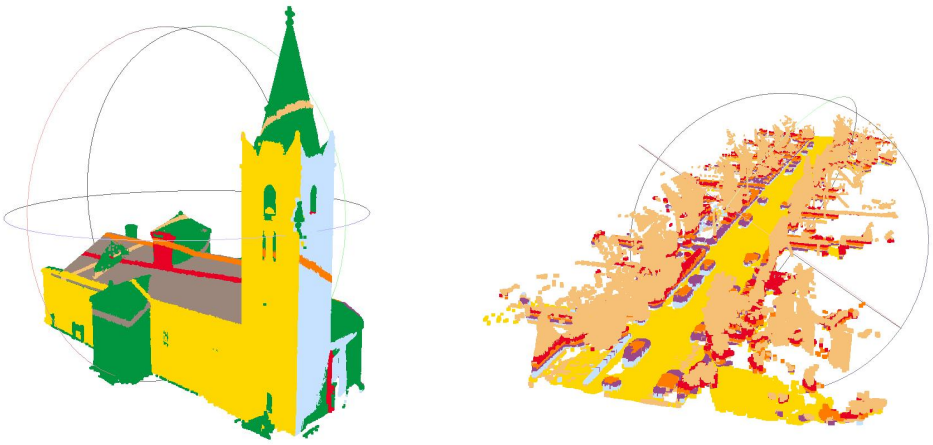


Figure 1: Détection de plusieurs plans grâce à RANSAC

La détection de plans fonctionne. Cependant, nous pouvons constater deux problèmes principaux. Premièrement, dans l'image de l'église, le plan détecté est infini et certains points lui sont associés alors qu'ils appartiennent à un autre plan, ce qui provoque les traits sut la tour et les toits. Secondement, pour la route, la route est bien détectée mais les plans suivants sont parallèles au sol et ne détectent pas les façades des maisons par exemple.

3 Filtrage des inliers en fonction des normales

Afin d'améliorer les résultats en profitant des informations fournies par les normales, nous avons ajouté à notre algorithme RANSAC l'option de conserver les points suffisamment proches du plan et dont la **normale est suffisamment alignée à celle de plan**. Pour cela, on calcule le produit scalaire des vecteurs normaux normalisés et on conserve les points dans 2 listes d'inliers différentes selon si le produit scalaire est positif ou négatif et absolument supérieur au threshold (0.75). Cette méthode permet de retirer des outliers dès les itérations random de RANSAC à faible coût de ressources. On peut voir sur l'amélioration de l'église entre les figures 1 et 2, la bande orange sur la face bleue a disparu. Pour la route, malgré plusieurs détections aberrantes, on commence à détecter les façades.

L'inconvénient majeur survient pour les faces texturées comme le toit de l'église. Les normales sont moins régulières donc des points légitimes sont retirés. Pour améliorer ça, on pourrait moyenner la normale des voisins de chaque point afin de lisser la surface.

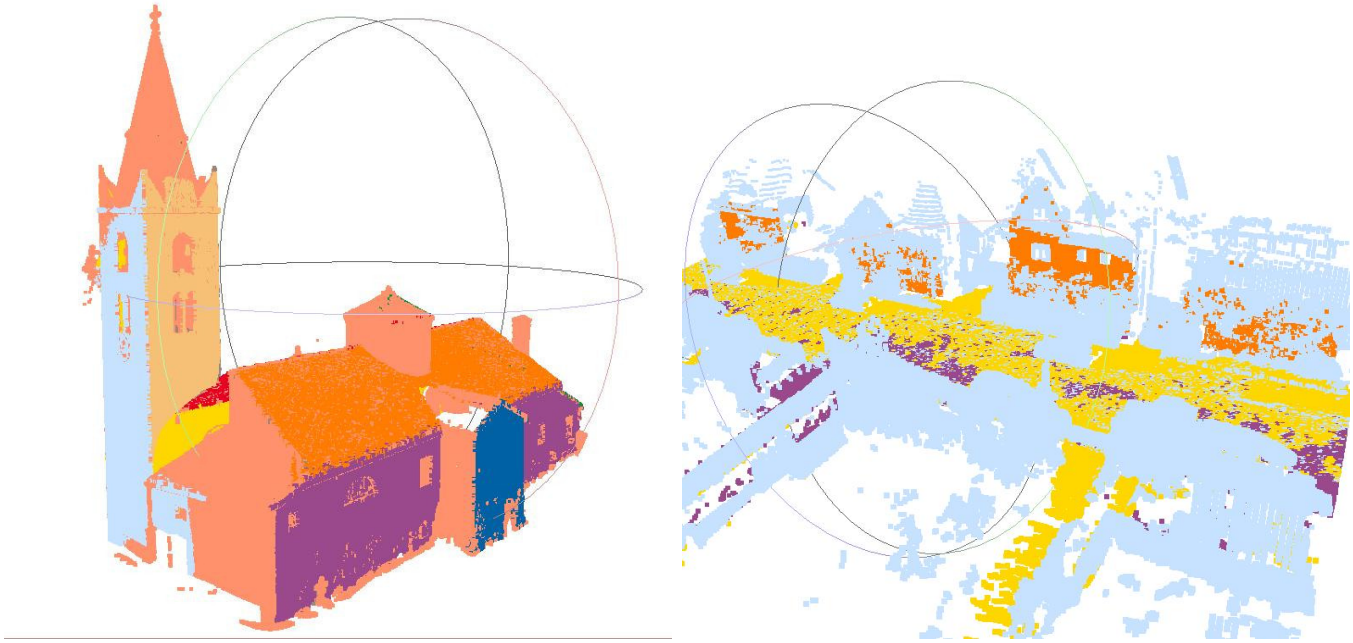


Figure 2: Détection de plusieurs plans grâce à RANSAC avec filtrage en fonction de normales

4 Suppression des outliers statistiques

Afin de retirer complètement les outliers des plans proposés par RANSAC, nous avons décidé d'appliquer une suppression des outliers statistiques sur le nuage de point proposé en sortie de RANSAC unique. L'algorithme suivi est le suivant :

- Répéter pour chaque point du plan détecté :
  - Calculer sa distance $\bar{d}$ moyenne à ses k plus proches voisins.
  - Calcul de la distribution des distances moyennes sur l'ensemble du nuage de points : valeur moyenne μ des $\bar{d}$ et déviation standard σ.
  - Conserver seulement les points pour lesquels $\mu - \alpha * \sigma <= \bar{d} <= \mu + \alpha * \sigma$, avec α un facteur de similarité.

Nous avons obtenu des résultats satisfaisants sur un nuage de test mais étant donné le temps de calcul, nous n'avons pas réussi à trouver les paramètres optimaux pour l'église (45 minutes).