1 Mise en contexte

Ce projet vise à développer un système de reconnaissance d'actions humaines à partir de capteurs LiDAR. L'hypothèse centrale repose sur l'idée qu'une action peut être modélisée comme une succession de poses distinctes. Par exemple, considérons le cas d'une personne qui se lève à l'aide d'une marchette. Cette action peut être décomposée en trois poses successives :

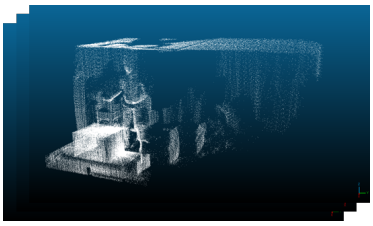
1. La personne est assise;
2. La personne est debout;
3. La personne est debout tout en tenant une marchette;

L'objectif est donc d'analyser une séquence de nuages de points générés par un capteur LiDAR, afin de reconnaître cette action à partir de l'enchaînement de poses détectées.

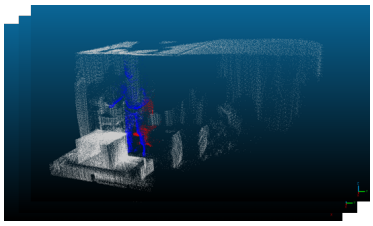
Dans un premier temps, chaque nuage de points est analysé individuellement. L'analyse commence par l'identification des objets pertinents pour l'action à détecter, c'est-à-dire les humains et les chaises. Un modèle de segmentation est utilisé pour distinguer les points associés à ces objets d'intérêt.

Une fois cette segmentation effectuée, un nouveau nuage de points est généré, contenant uniquement les objets pertinents, tandis que les points associés à l'arrière-plan ou à d'autres éléments sont supprimés. Ce nuage filtré est ensuite transmis à un modèle de classification, entraîné à reconnaître la pose correspondant à la configuration spatiale entre l'humain et la chaise.

Enfin, une fois les poses identifiées pour chaque instant de la séquence, celles-ci sont analysées dans leur ordre d'apparition. Si la succession observée correspond à une personne d'abord assise sur une chaise, ensuite debout, puis enfin debout en tenant une marchette, alors l'action consistant à se lever à l'aide de la marchette est considérée comme reconnue.

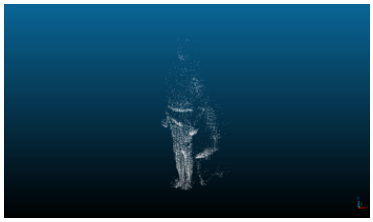


(1) Acquisition de nuages de points



(2) Segmentation des objets d'intérêts

(3) Regroupement des objets d'intérêts



(4) Nuages de points sur lesquels la classification des poses est effectuée afin de vérifier si la séquence correspond à l'action de se lever correctement

2 Entraînement des modèles

Avant d'entamer l'entraînement des modèles, il est nécessaire de préparer adéquatement les nuages de points. Pour cela, il est recommandé d'utiliser un logiciel permettant la visualisation et la modification de ces données, notamment en vue d'y apposer des annotations manuelles. L'outil recommandé pour cette tâche est CloudCompare, qui offre des fonctionnalités adaptées à ce de type de traitement.

Par ailleurs, une considération importante tient au fait que le modèle de segmentation utilisé ne prend pas en charge l'augmentation des données. Il est donc essentiel que tous les nuages de points soient exprimés dans un même repère spatial. Si ce n'est pas le cas, il convient de réaligner les scènes en effectuant les transformations nécessaires afin de garantir une cohérence entre les ensembles de données utilisés à l'entraînement et ceux utilisés en phase d'inférence.

Le lecteur est invité à consulter les capsules vidéo associées pour une démonstration complète du processus d'entraînement des modèles de segmentation et de classification.

2.1 Ensembles de données

Les capsules vidéo mentionnées précédemment présentement brièvement comment on peut construire un ensemble de données personnalisé pour l'entraînement des modèles. Dans le cadre de ce projet, un petit ensemble de données a été constitué, principalement à des fins de test et d'expérimentation.

Cet ensemble de données est disponible dans un dossier partagé. Il contient les fichiers nécessaires à l'entraînement des modèles de segmentation et de classification.

2.1.1 Structure des données pour la classification des poses

Pour la classification des poses, deux répertoires issus de l'ensemble des données partagé doivent être placés dans le dossier suivant, relativement à la racine du projet `pipeline`.

```
pipeline/docker/pose_classification/pose_classification/data/
```

Les deux répertoires à copier sont :

- `modelnet40_normal_resampled`
- `modelnet40_normal_resampled_default`

Ces répertoires contiennt les données nécessaires à l'entraînement et au test du modèle de classification. Ils doivent être positionnées exactement à cet emplacement pour assurer le bon fonctionnement des scripts associés.

2.1.2 Structure des données pour la segmentation des scènes

L'ensemble de données utilisé pour la segmentation sémantique est structuré en plusieurs dossiers correspondant à différentes postures.

Chaque dossier contient plusieurs fichiers et répertoires qui représentent différentes étapes du traitement des scènes :

- `raw/` : Nuages de points bruts enregistrés par le capteur LiDAR.
- `segmented.bin` : Scènes dans lesquelles les objets ont été segmentés, sans alignement.
- `segmented_aligned.bin` : Scènes segmentées et alignées dans un repère commun.
- `segmented_aligned/` : Exportation des scènes contenues dans `segmented_aligned.bin`.
- `human_chair_models.bin` : Objets d'intérêt pour la classification uniquement.
- `human_chair_models/` : Exportation des scènes contenues dans `human_chair_models.bin`.
- `processed/` : Scènes converties au format attendu par l'ensemble de données `ScanNet`.

3 Pipeline d'inférence

Le pipeline d'inférence est responsable de l'exploitation des modèles préentraînés afin de déterminer si une personne effectue correctement une activité donnée. Elle prend en entrée des nuages de points et applique les modèles de segmentation et de classification pour analyser la séquence capturée.

Le lecteur est invité à consulter une capsule vidéo dédiée, qui illustre le fonctionnement de cette pipeline lorsqu'elle est appliquée à des scènes de nuages de points.

4 Acquisition et traitement des nuages de points

Un second projet, distinct du pipeline principal présenté précédemment, permet d'enregistrer et de traiter des nuages de points capturés à l'aide du capteur LiDAR. Ce projet se concentre sur deux opérations principales :

- **L'enregistrement de nuages de points** à l'aide du script `point_cloud_record.sh`. Ce script exécute un programme qui utilise un capteur LiDAR pour capturer un nuage de points, puis enregistre ces points dans un fichier au format `.ply`. Pour que cette opération fonctionne correctement, il est nécessaire d'installer le SDK fourni par le fabricant.
- **Le traitement des fichiers** `.ply` à l'aide du script `point_cloud_process.sh`. Ce script prend un fichier `.ply` en entrée et le convertit vers un format conforme à l'ensemble de données `ScanNet`. Le processus de standardisation est déjà détaillé dans une capsule vidéo, à laquelle il est recommandé de se référer.

Bien que ce projet soit séparé du pipeline d'inférence principal, une intégration pourrait être envisagée si l'on souhaite faire de l'inférence en temps réel. En effet, le pipeline tel qu'il a été jusqu'ici repose sur des scènes préalablement enregistrées, et n'a pas encore été intégré à un flux d'acquisition en direct.