CR de réunion d'avancement de projet

- **Participants** : Théo Larcher, Pascal Vasseur
- **Date** : 24/06/2021
- **Voie** : mail

Travail

- Récap travail (cf. JdB).

UE4

- Grâce aux réponses d'un internaute sur mon post sur le forum d'Unreal Engine, j'ai réussi à établir un 'Blueprint' permettant de trouver les transformations appliquées aux instances d'arbres (positions, rotation, mise à échelle ['scale']).
    - Un blueprint est une solution de coding visuelle proposée par UE4, similaire aux Simulink de matlab.
    - Les données ont d'abord été écrites dans le console de l'UE4 Editor puis exportées dans un fichier texte. On y trouve les données des transformation, le 'Mesh Component' associé (càd le modèle d'arbre parmi les 12 définis) et l'ID de son instance.
    - J'ai vérifié que les données étaient cohérentes en vérifiant les coordonnées d'un des arbres en y superposant un objet facilement déplaçable : le PlayerStart, et en retrouvant ces coordonnées dans le fichier texte. Tout correspond bien.
    - Dernier point, les coordonnées des arbres sont relatives à la map UE4, hors celles du drone exportées par AirSim **relatives** au drone. C'est-à-dire que pour le drone, le (0,0,0) est sa position de départ, quelle qu'elle soit dans le monde. Pour palier cela, il suffit juste d'enregistrer un nouveau parcours à partir d'un PlayerStart initialisé en (0,0,0) du monde.
- J'ai partagé ces informations avec Charles, j'attends son retour.
- Il reste encore cependant à trouver l'information de la grosseur des arbres. Si je parviens à la trouver pour chaque modèle d'arbre, il suffira ensuite de la multiplier par la transformation 'scale' pour obtenir sa grosseur dans le monde.
- Enfin il est théoriquement possible de demander directement la distance entre l'objet-caméra (notre drone dans notre cas) et les instances d'arbres, mais je n'ai pas de garantie que cela soit possible via l'API Python, or c'est bien ce qu'utilisent AirSim et le script de Charles pour faire leur captures. Il y aurait plusieurs couches de complexité donc je pense qu'on est aussi bien à calculer des distances nous-même.

Mapping bitterlich

- L'acquisition des vérités terrain des positions ouvre la porte aux métriques de précision de nos estimations via la méthode ad-hoc inspirée de bitterlich. Une fois les nouvelles acquisitions faites, cela peut être le prochain point de développement.

ORB-SLAM3

- J'ai créé de nouveaux datasets d'images plus proches temporellement parlant grâce à mon script d'interpolation appliqué sur les fichiers d'enregistrement de pose d'AirSim, pour atteindre un équivalent ~24fps pour des mouvements de translation légers (pas de rotation). Le nuage de point s'établie vite dès qu'il y a du mouvement et le mapping ne se brise pas. Pour 3000 descripteurs demandés, il y en a entre 400 et 600 de détectés à chaque image et une quinzaine de Keyframes.
    - Ces dernières sont, à l'issu de l'algo, stockées dans un fichier type csv contenant le nom de l'image en référence, les 3 positions dans l'espace 3D et ce que je devine être des quaternion de rotation (il n'y a pas le nom des champs).

Objectifs restants

- Estimer le décalage entre 2 images dans notre estimation de carte pour corriger les positions et grosseurs
- Revenir sur l'apprentissage deep pour la segmentation sémantique des arbres et ré-entraîner un réseau à partir d'un tracé commun entre le monde UE4 normal et le monde UE4 label
- Définir une première *pipeline* adaptée à des données nouvelles 'sans triche'