Université de Strasbourg

UFR de mathématique et d'informatique
Université de Strasbourg

Cursus Master Informatique – Image et 3D
UFR de Mathématique et d'Informatique
Université de Strasbourg

Travail d'Étude et de Recherche

COMPLÉTION DE NUAGES DE POINTS 3D OCCULTÉS DE PLANTES

Auteur:
Hussein EL AMINE

Enseignants référents:
Franck HETROY-WHEELER
Joris RAVAGLIA

Année universitaire 2022-2023

Contents

1 INTRODUCTION **3**

2 ETAT DE L'ART **5**
2.1 MÉTHODES GÉOMÉTRIQUES 5
2.2 MÉTHODES D'APPRENTISSAGE PROFOND 5

3 APPORT DE CE TRAVAIL **9**
3.1 BASE DE DONNEES SYNTHETIQUES 9
3.2 ADAPTATION DE L'ALGORITHME 10

4 RESULTAT **10**
4.1 OBSERVATIONS 10

5 ETUDES QUALITATIVES **10**
5.1 DISCUSSION 14

6 CONCLUSION **15**

1 INTRODUCTION

Les représentations en nuage de points 3D sont couramment utilisées lors de la capture d'objets réels. Des techniques telles que LiDAR[1] ou la photogrammétrie sont largement employées pour générer des nuages de points 3D. Toutefois, ces méthodes de génération de nuages de points présentent plusieurs problèmes. D'une part, des imprécisions liées aux dispositifs de capture peuvent engendrer des divergences entre la réalité et la reconstruction. D'autre part, la qualité de la reconstruction dépend en grande partie du nombre de points de vue utilisés lors des acquisitions. L'une des conséquences du manque de points de vue est l'apparition de "trous" dans le nuage de points reconstruit. Dans ce contexte, nous définirons les "trous" comme des zones où l'information est manquante dans le nuage de points 3D. Ces trous, de tailles variables, peuvent altérer la topologie de l'objet observé.

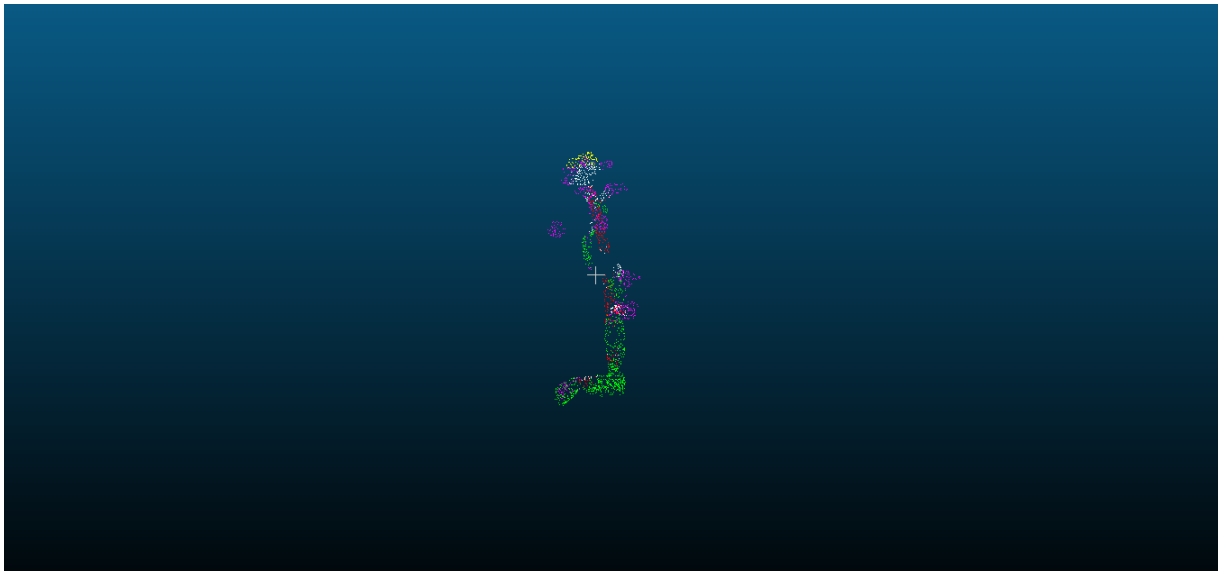

Fig. 1 Nuage de points occulté d'un plant de base de données Arabidopsis

En dehors du nombre de points de vue, des trous peuvent également être causés par des parties de l'objet qui occultent une partie de sa propre géométrie. Ce problème se pose particulièrement lors de l'acquisition de plantes. Par exemple, les feuilles d'une canopée d'arbre peuvent cacher une grande partie de la structure interne des branches.

Dans ce travail, notre objectif est de compléter automatiquement les zones occultées d'un nuage de points 3D de plantes, en prenant en compte la géométrie et la topologie de l'objet d'origine. En effet, bien que les problèmes évoqués précédemment puissent être en grande partie résolus en augmentant le nombre de points de vue, cette approche n'est pas efficace pour l'acquisition de plantes en milieu naturel. En effet, les plantes sont soumises aux conditions météorologiques et peuvent donc beaucoup se déplacer entre deux acquisitions. De plus, il n'est pas toujours possible pour l'observateur de se déplacer librement autour de l'objet observé. De ce fait, le nombre d'acquisitions nécessaires pour obtenir un nuage de points de qualité satisfaisante serait excessivement élevé. La complexité de résoudre les problèmes liés à l'acquisition de plantes justifie l'étude de méthodes spécifiques de complétion de nuages de points adaptées aux plantes.

Dans ce travail, nous commencerons par présenter l'état de l'art des méthodes de complétion de nuages de points 3D. Ensuite, nous aborderons spécifiquement la méthode que nous avons sélectionnée et évaluée pour son efficacité dans le contexte des plantes. Il convient de mentionner que ce travail s'inscrit dans la continuité des

travaux précédents dans [7] , qui ont permis de récupérer des données nécessaires pour la continuité de cette étude.

[1] LiDAR: Light Detection and Ranging, une technique de mesure à distance utilisant des lasers.

2 ETAT DE L'ART

Au fil des années, plusieurs approches ont été développées pour compléter des ensembles de points 3D, et elles peuvent être regroupées en deux catégories distinctes. La première catégorie, plus ancienne, comprend les méthodes géométriques, qui exploitent la géométrie de l'objet pour remplir un ou plusieurs trous dans celui-ci. La seconde catégorie regroupe les méthodes utilisant l'apprentissage profond.

2.1 MÉTHODES GÉOMÉTRIQUES

Les méthodes géométriques utilisent la géométrie de l'objet pour remplir les trous présents dans le nuage de points. Elles exploitent des informations telles que les contours, les surfaces ou les relations spatiales pour estimer les valeurs manquantes.

Les méthodes géométriques exploitent la structure géométrique de l'objet pour remplir les trous dans le nuage de points en utilisant des informations telles que les contours, les surfaces ou les relations spatiales. Cependant, dans le contexte de la complétion de nuages de points occultés, ces méthodes ont été moins explorées.

Par contre, les méthodes basées sur l'apprentissage profond utilisent des réseaux de neurones pour apprendre à compléter les zones manquantes dans le nuage de points. Elles sont capables de capturer des informations complexes et des relations non linéaires, ce qui les rend plus performantes dans des scénarios difficiles.

En raison de leur supériorité dans la complétion de nuages de points occultés, les méthodes basées sur l'apprentissage profond ont gagné en popularité et sont actuellement largement utilisées dans la recherche. Dans la suite de cette discussion, nous allons nous concentrer davantage sur ces méthodes et explorer comment elles peuvent être appliquées pour résoudre le problème de complétion de nuages de points occultés.

2.2 MÉTHODES D'APPRENTISSAGE PROFOND

Les méthodes basées sur l'apprentissage profond pour la complétion de nuages de points occultés utilise principalement d'architectures d'encodeur-décodeur et de transformateurs.

Les architectures d'encodeur-décodeur sont largement utilisées dans de nombreux domaines de l'apprentissage automatique. Dans le contexte de la complétion de nuages de points, un encodeur prend en entrée le nuage de points partiel avec des zones manquantes et le transforme en une représentation latente ou vectorielle de dimension réduite. Cette représentation latente capture les caractéristiques et les informations essentielles du nuage de points.

Ensuite, le décodeur prend la représentation latente et génère les valeurs manquantes, complétant ainsi le nuage de points. Le décodeur est chargé de générer des points cohérents et réalistes, en se basant sur la représentation latente et sur l'information contextuelle disponible.

Une autre approche prometteuse est l'utilisation de modèles basés sur les transformateurs. Les transformateurs sont des architectures de réseaux neuronaux qui ont été introduites pour des tâches de traitement du langage naturel, mais qui se sont avérées efficaces pour la génération de séquences en général. Dans le contexte de la complétion de nuages de points, les transformateurs peuvent être adaptés pour capturer les relations spatiales et les motifs dans le nuage de points, en se basant sur le contexte des points environnants pour prédire les valeurs manquantes.

Les modèles basés sur les transformateurs peuvent également exploiter des mécanismes d'attention pour pondérer l'importance des différents points dans la génération des valeurs manquantes. Ces mécanismes d'attention permettent aux modèles d'accorder plus d'importance aux parties pertinentes du nuage de points, améliorant ainsi la qualité et la cohérence des prédictions.

GRNet (Gridding Residual Network) [2] étudiée récemment dans[8], est une architecture de reseaux de neurones utilisée pour compléter les nuages de points 3D. Elle comprend plusieurs étapes. Dans un premier temps, le nuage de points LiDAR d'origine est converti en grilles 3D, qui servent d'entrée à GRNet. Cette conversion permet de régulariser les points désorganisés et offre une représentation structurée pour le traitement ultérieur.

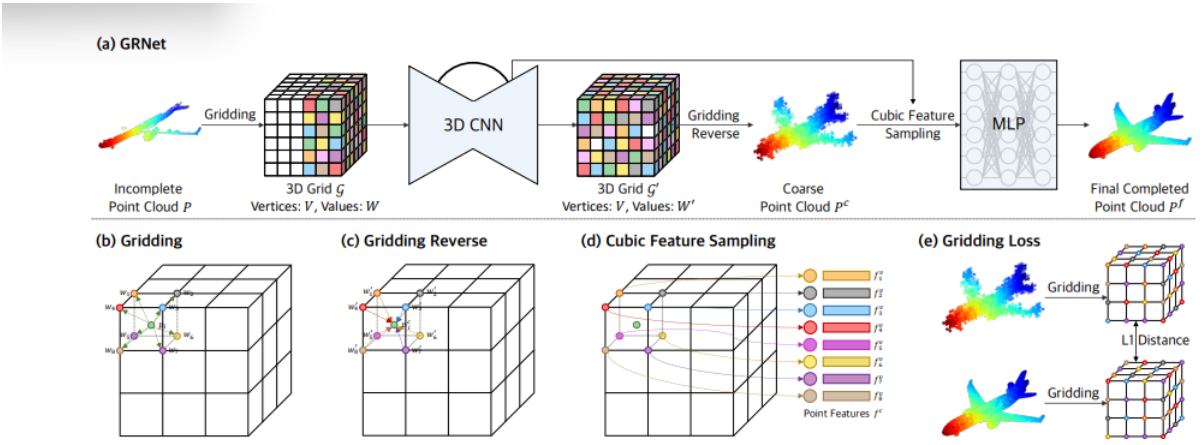

Fig. 2 Architecture de GRNet, extrait de [2]

Ensuite, un réseau neuronal convolutif (CNN) 3D est construit pour compléter les parties manquantes de la grille 3D. Le CNN suit une architecture encodeur-décodeur inspirée d'U-Net, intégrant des connexions sautées pour préserver et utiliser efficacement l'information spatiale. Le réseau comprend quatre couches de convolution 3D dans l'encodeur, suivies de la normalisation par lots (BN), de l'activation Leaky ReLU et du pooling maximal. Les canaux de sortie augmentent progressivement, et deux couches entièrement connectées suivent les couches de convolution. Le décodeur est composé de quatre couches de convolution transposée avec un pas de 2. La normalisation par lots et la fonction d'activation ReLU sont appliquées dans l'ensemble du réseau.

Après avoir complété la grille 3D, un processus de gridding reverse est effectué pour générer un nuage de points grossier. Ce processus implique le calcul des valeurs et des coordonnées des huit sommets de chaque cellule de la grille 3D, à partir desquels le nuage de points grossier est généré.

Pour extraire les caractéristiques et reconstruire les détails des objets, un échantillonnage de caractéristiques cubiques est utilisé. Les caractéristiques sont extraites du nuage de points grossier en sélectionnant les caractéristiques des sommets des cellules de la grille 3D. Les caractéristiques sélectionnées sont concaténées pour former une carte de caractéristiques.

Enfin, un perceptron multi-couches (MLP) est utilisé pour calculer l'écart entre le nuage de points grossier et le nuage de points complété, dans le but de récupérer les détails manquants des objets. Le MLP prend en entrée le nuage de points grossier et ses caractéristiques associées, et génère le nuage de points complété. Il est composé de quatre couches entièrement connectées de tailles décroissantes, ainsi qu'une couche de sortie qui génère le nuage de points complété.

Dans [8] GRNet est adapté pour compléter les points LiDAR des plants de maïs. Cependant, des tests manquent pour évaluer l'efficacité de cette méthode. De plus, GRNet repose sur la conversion du nuage de points en grilles 3D structurées pour effectuer la complétion, ce qui peut être un avantage dans certains cas en régularisant les points désorganisés. Cependant, cela peut aussi être un inconvénient lorsque la structure en grilles ne correspond pas parfaitement à la géométrie réelle des objets, ce qui peut entraîner la perte ou la mauvaise représentation de détails subtils ou de formes complexes. De plus, l'efficacité de GRNet peut être sensible à la taille de la grille 3D.
Après avoir brièvement examiné la méthode GRNet, étudiée récemment, nous allons maintenant passer à une autre méthode pour poursuivre notre exploration.

PointTr, qui introduit les Transformers, est un exemple illustrant l'utilisation des Transformers et de leurs mécanismes d'attention. Ces mécanismes d'attention permettent au réseau de se "souvenir" d'une quantité

variable d'informations pendant la phase d'entraînement. Initialement utilisés dans le domaine du traitement du langage naturel, ils ont ensuite été appliqués à d'autres domaines tels que la vision par ordinateur. Les Transformers sont une catégorie de réseaux de neurones qui utilisent des mécanismes d'attention. Leur principe a été présenté pour la première fois dans [1].

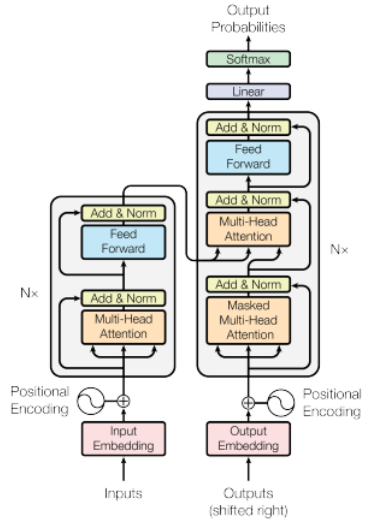

Fig. 3 Architecture de Transformer [1]

Dans le contexte des nuages de points, cela signifie que le Transformer prendra une série de points 3D en entrée, les mémorisera, puis recevra une autre série de points. Grâce à cela, le Transformer peut déduire des relations géométriques entre les points, ce qui est particulièrement utile dans notre cas. En se souvenant de plusieurs points, le réseau peut apprendre les relations géométriques entre eux. Cependant, un inconvénient majeur des Transformers est leur lenteur. Néanmoins, ils sont beaucoup plus rapides que les mécanismes existants, car ils peuvent tirer parti de l'architecture hautement parallèle des GPU, ce qui n'était pas possible avec leurs prédécesseurs.

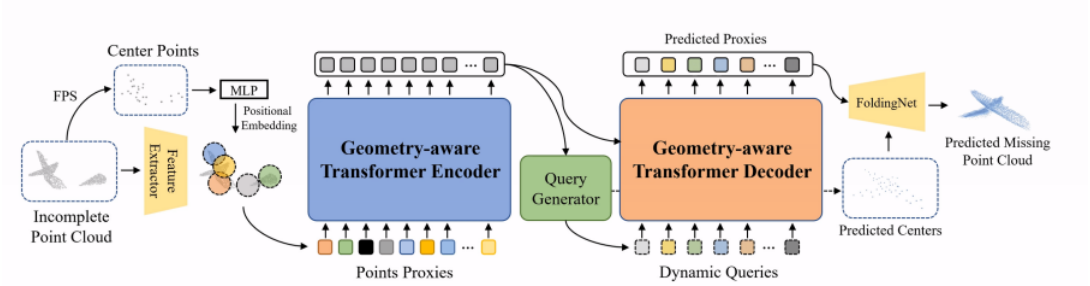

Fig. 4 Architecture de PointTr, extraite de [3]

Pour permettre l'utilisation des Transformers, PointTr simplifie d'abord le nuage de points en entrée. Cette simplification produit un nuage de "pseudo points" qui représente la géométrie et la topologie du nuage d'entrée. Ces "pseudo points" sont ensuite donnés en entrée au Transformer, qui génère d'autres "pseudo points" représentant les régions à compléter. Afin de réduire les temps de calcul, l'état initial des "pseudo points" en sortie - c'est-à-dire l'état des "pseudo points" avant de passer dans le décodeur - est utilisé comme centre pour la génération du nuage de points dense. Enfin, les "pseudo points" et les centres sont utilisés comme entrée pour FoldingNet [4], qui génère le nuage de points dense, résultat de PointTr.

Dans le cas spécifique de PointTr, l'entrée est un nuage de points dense de 2048 points, et la sortie est identique à celle de FoldingNet, c'est-à-dire un nuage de points dense de 2048 points.

SnowflakeNet [6] A la date de rédaction de ce travail, PointTr, SnowflakeNet et Local Displacement in [5] sont les méthodes les plus performantes dans leur tâche de complétion de nuage de points 3D, tenant en compte que entre la redaction de [7] et ce rapport, [5] a été intorduit. La figure 5 détaille l'architecture du modèle. L'encodeur de SnowflakeNet consiste en 3 niveaux d'abstraction d'ensembles de points en intercalant des Transformers pour extraire les informations locale (via le mécanisme de mémoire à court terme des Transformer).

L'encodeur fournit un vecteur de forme qui sert d'entrée au générateur de graine. Le générateur de graine produit un nuage épars représentant les graines autours desquelles l'étape suivante va construire le nuage dense. La dernière étape est appelé "Snowflake Point Deconvolution"(SPD). Trois niveaux de SPD permettent de successivement augmenter la taille du nuage en tenant compte des caractéristiques de formes, afin d'obtenir un nuage de points 3D dense en sortie, l'algorithme prend en entrée un nuage de points de taille N = 2048 et en sortie de taille N = 16348

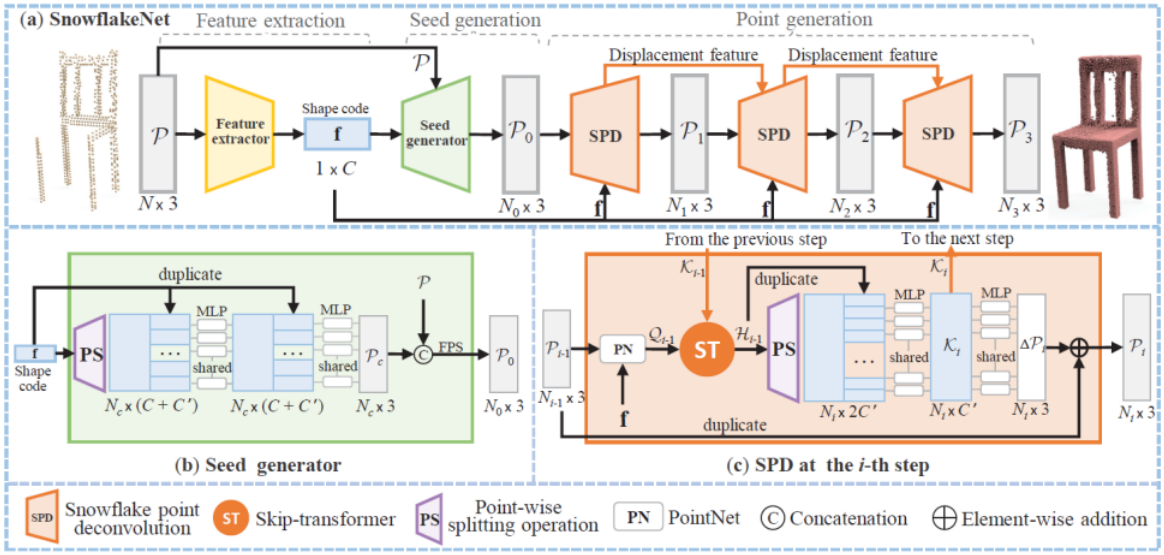

Fig. 5 Architecture de SnowFlakeNet, extrait de [6]

Learning Local Displacement [5] est une architecture basée sur les transformers qui vise à améliorer l'état de l'art par rapport à PoinTr [3]. Cette architecture dérive de PoinTr en incorporant nos propres opérateurs, comme résumé dans la figure 6.

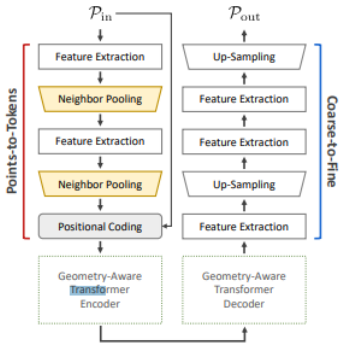

Fig. 6 Architecture de Learning Local Displacement, extrait de [5]

Avant de calculer les mécanismes d'attention dans le transformer, les scans partiels sont sous-échantillonnés en raison de la contrainte de mémoire du GPU. PoinTr [3] utilise l'échantillonnage des points les plus éloignés (FPS) pour réduire le nombre de points et le MLP pour convertir les points en caractéristiques.

En revanche, l'architecture applique des nouvels opérateurs, qui impliquent l'alternance de l'extraction des caractéristiques et du pooling des voisins. De plus, une codification positionnelle basée sur les coordonnées 3D est ajoutée aux caractéristiques. Cette étape est représentée dans la figure 6 .

Ensuite, il utilise les transformers sensibles à la géométrie de [3], ce qui permet de produire un nuage de points grossier.

À partir du nuage de points grossier, nous remplaçons ensuite leur stratégie de reconstruction grossière à

fine par nos propres opérateurs. Cela comprend une série d'opérations d'extraction de caractéristiques et de sur-échantillonnage alternées, comme indiqué dans la figure 6.

les résultats du pooling des voisins dessinent les contours du nuage de points d'entrée pour capturer les structures significatives de l'objet plus precis que [3]

Le choix entre les méthodes, qui sont présentées dans l'ordre chronologique, s'est finalement porté sur SnowFlakeNet [6] plutôt que Learning Local Displacement [5]. Cela s'explique par le fait que, visuellement, comme le montre la figure 7, Learning Local Displacement présente une sensibilité au bruit plus prononcée sur les structures no ordinaire, ce qui est le cas dans cette étude parce que les plantes ont une strucutre complexe et fine donc trop varié entre une plante et autre. Cependant, en termes de performances, les deux méthodes sont proches. Étant donné la contrainte de temps, il n'était pas possible de tester les deux méthodes, et il a été décidé de se concentrer sur SnowFlake. Il convient de noter que l'étude de Wang et al. a été introduite plus récemment, après [7]

Fig. 7 Voiture completé par LLD n'ayant pas des roues en entrée [5]

3 APPORT DE CE TRAVAIL

Dans la suite, nous présenterons les apports de ce Travail d'Etude et de Recherche en terme de production logiciel et de données. Ces apports se résument ainsi:

1. Un jeu de données prêt à l'emploi composé de nuages de points 3D synthethiques occutés d'Arabidopsis generée par [7] avec leur verité de terrain
2. Une version du logiciel SnowFlakeNet en Python(3.7), modifiée pour utiliser le jeu de données mentionné plus haut, avec d'autres script utils (plus en details au-dessous).

3.1 BASE DE DONNEES SYNTHETIQUES

Les données à notre disposition sont cinq acquisitions d'Arabidopsis. Ces cinq plantes ont été scannées à plusieurs stades de maturité. Au total, on dispose de 44 nuages de points. Sur ces 44 nuages, où il a été procédé par [7] à une sélection manuelle de points dont le retrait entraînerait tout ou partie des changements topologiques ci-après :

1. Augmentation du nombre de composantes connexes en retirant un noeud où se rencontrent plusieurs tiges
2. Apparition d'un bord en retirant un morceau de tige mais conservant la connexité
3. Augmentation du nombre de composantes connexes en coupant une feuille en deux
4. Augmentation du nombre de composantes connexes en coupant une tige en son milieu
5. Apparition d'un bord avec un trou au milieu d'une feuille

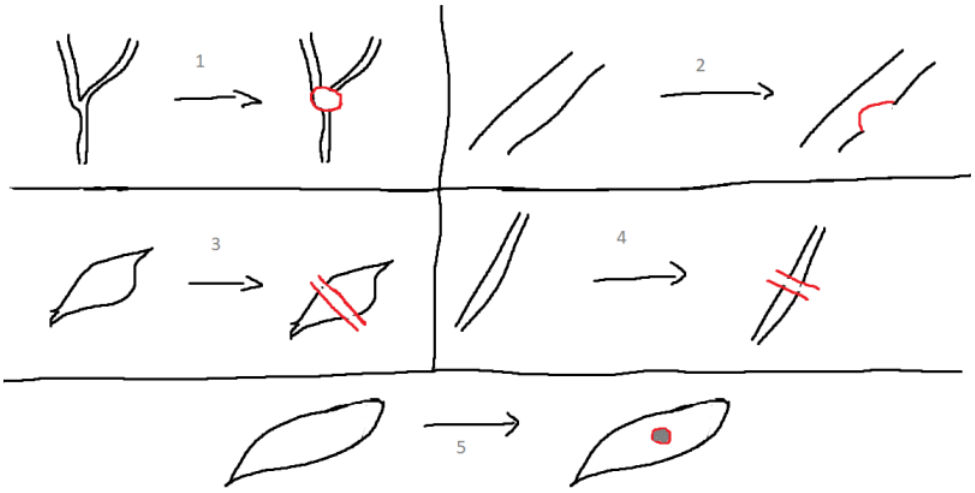

Fig. 8 Illustration des différents types de trous. La partie gris foncé du type 5 illustre une absence de matière, extrait de [7]

3.2 ADAPTATION DE L'ALGORITHME

Modifications de l'algorithme: Il a été nécessaire d'adapter plusieurs parties de SnowFlakeNET afin d'entrainer le modèle sur notre jeu de données, à savoir :

1. Les fichiers de configurations de modèle et les fichiers de catégories de données nécessaires pour le fonctionnement de l'algorithme (donc un script python a été implementé pour ce but).
2. Une classe destinée à présenter le jeu de données au modèle pendant l'entraînement et donc spécifique au jeu de données Arabidopsis
3. Un script de upsampling pour augmenter la densité de données prédites (n'a pas été utilisé dû à la taille du nuage de points et des contraintes du temps d'exécution)
4. Un script pour la sélection partielle de données

4 RESULTAT

4.1 OBSERVATIONS

Temps de calcul et d'entraînement de SnowFlakeNet: SnowFlakeNET a été entraîné sur une machine ayant à sa disposition 16Gb RAM, GTX 1060 (6Gb), donc SnowFlakeNET utilise une taille de batch de 32, et la base de données comprend 35564 nuages de points, donc après expérimentations sur la capacité de la machine en question, il a été jugé d'utiliser un batch size de 4 qui est relativement petit, et une sélection de 12000 échantillions approximativement qui sont reseléctionnés aléatoirement chaque 5-10 epochs, et avec une parallélisation de 4 workers, après 100 epochs d'entraînement, nous avons augmenté la taille de batch à 12 et une seléction totale (1100/1251 nuage de points par acquisistion) qui rend l'entraînement plus lent mais plus précis et proche du but, au final le modèle a été entraîné (avec expérimentation) de plus de 240 epochs (200h) approximativement.

surapprentissage: à l'epoch 100 le modèle commence à bien prédire quelques trous, mais le nuage de points est devenu de plus en plus "sparse" ayant un regroupement de points, plusieurs expérimentations ineffectives de weight decay et learning rate ont été effectués surtout après l'augmentation de taille de batch et une introduction plus uniforme de données d'entraînement, possiblement introduction de plus de données.

5 ETUDES QUALITATIVES

Nuage d'origine: La Figure 9 illustre plusieurs trous générés lors de la création du jeu de données, visible dans la partie intermédiaire de la tige.

Fig. 9 Illustration des différents types de trous. Nuage de points occulté de taille N = 2048 points

25 epochs commence a remplire le trou mais manque de structure.

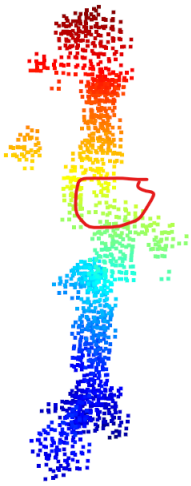

Fig. 10 Nuage de points prédit après 25 epochs points ajoutés dans le cercle rouge de taille N = 16384 points

Fig. 11 Nuage de points prédit après 31 epochs de taille N = 16384 points

Epoch 67 amélioration de densité de nuage de points avec une complétion sensiblement identique en observant et comparant avec l'**Epoch 31**

Fig. 12 Nuage de points prédit après 67 epochs de taille N = 16384 points

Epoch 100 ajout de plus de points, plus précis

Fig. 13 Nuage de points prédit après 100 epochs de taille N = 16384 points

Epoch 196 structure globale plus proche du résultat souhaité, mais perte sgnificative de densité et des informations locales

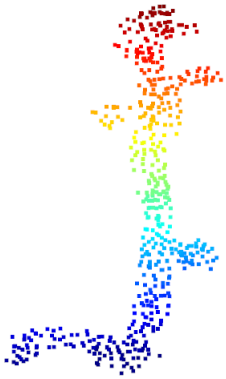

Fig. 14 Nuage de points prédit après 196 epochs de taille N = 16384 points

Epoch 250+ Plus de "perte" de points dans le nuage de points

5.1 DISCUSSION

En observant le progrès de l'entraînement, les premiers 60 epochs ne mènent pas à un résultat interressant mais commence à converger et à mettre des points dans les trous.

Dans les epochs 100+, les loss (Chamfer distance L2) ont diminué significativement par rapport à 67, et ont continué à diminuer encore plus sur les 200, ce qui n'est pas aligné avec la qualité de prédiction, donc ce qui peut être dû à la seléction partielle écrite au-dessus dans le rapport et l'efficacité de fonction de loss chamfer distance.

Par exemple il existe une autre fonction Earth Mover's Distance (EMD) qui est plus complexe en terme de calcul et ne prend pas que la distance entre les points, mais aussi des coûts d'affectation nécessaires pour transformer un ensemble de points en un autre.

6 CONCLUSION

Pour conclure ce rapport, nous avons généré (code fourni par [7]) des données synthétiques de nuages de points occultés pour entraîner le modèle SnowFlakeNET. Malgré quelques expérimentations de tuning de paramètres sur plusieurs epochs, nous avons rencontré des contraintes liées à la machine utilisée, qui m'appartient personnellement et est également utilisée pour d'autres travaux. Ces contraintes ont entravé notre démarche d'entraînement et nous n'avons pas pu tester ainsi la méthode récente de Wang [5]

Cependant, en prenant en compte ces limitations, je suis d'avis qu'il serait bénéfique de poursuivre les tests sur ce modèle en utilisant des machines appropriées. Nous devrions diversifier les données synthétiques pour réduire les biais et peut-être augmenter leur taille. De plus, étant donné que les nuages de points liés aux plantes présentent des particularités, il est nécessaire de mener des recherches plus approfondies dans ce domaine.

En fin de compte, j'ai pu explorer ces méthodes et entraîner ce modèle, en enregistrant des checkpoints qui pourront être utilisés pour de futures études. Il est donc possible de continuer à développer et améliorer les performances de la complétion de nuages de points pour les applications liées aux plantes.

References

[1] A. Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems* (2017).

[2] Haozhe Xie et al. "GRNet: Gridding Residual Network for Dense Point Cloud Completion". In: (2020).

[3] X. Yu et al. "Pointr : Diverse point cloud completion with geometry-aware transformers". In: *ICCV* (2021).

[4] Y. Yang et al. "Foldingnet : Point cloud auto-encoder via deep grid deformation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018).

[5] Yida Wang al. "Learning Local Displacements for Point Cloud Completion". In: *Cornell University* (2022).

[6] Peng Xiang. "SnowflakeNet: Point Cloud Completion by Snowflake Point Deconvolution With Skip-Transformer". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2021).

[7] Evan ZARELLA. "Complétion de nuages de points 3D de plantes. Rapport TER". In: *Université de Strasbourg* (2022).

[8] Ying Zhang et al. "Completing 3d Point Cloud of Thin Corn Leaves Using 3d Gridding Convolutional Neural Networks". In: *SSRN* (2022).