Université Gustave Eiffel

Point cloud based large-scale place recognition

Thèse de doctorat de l'Université Gustave Eiffel

École doctorale n° 532, Mathématiques et STIC (MSTIC)
Spécialité de doctorat: Sciences et Technologies de l'Information Géographique
Unité de recherche : LaSTIG - Laboratoire en Sciences et Technologies de l'Information Géographique

Thèse présentée et soutenue à l'Université Gustave Eiffel, le 20/02/2026, par :

Chahine-Nicolas ZEDE

Composition du Jury

Michel KASSER Professeur des universités, HEIG-VD	Rapporteur
Hervé LE BORGNE Directeur de recherche, CEA LIST	Rapporteur
Marie-Odile BERGER Directrice de recherche, INRIA Nancy Grand Est	Présidente du jury

Encadrement de la thèse

Valérie GOUET-BRUNET Directrice de recherche, Univ. Gustave Eiffel, IGN, LaSTIG	Directrice de thèse
Laurent CARAFFA Chargé de recherche, Univ. Gustave Eiffel, IGN, LaSTIG	Co-Encadrant de thèse

Acknowledgment

I would like to thank you for opening this manuscript, whether you are genuinely interested in my work, simply curious, someone who knows me and wonders what I have been doing during these past years of my thesis, or perhaps just a lost reader who happened to find these pages.

Je souhaite remercier chaque personne que j'ai eu la chance de côtoyer avant, pendant et en dehors de ma thèse. Je souhaite m'excuser auprès de toutes les personnes qui ne sont pas mentionnées ici ; leur présence et leur soutien n'en ont pas été moins précieux.

Je tiens tout d'abord à remercier mon encadrement de thèse pour son soutien tout au long de ces trois dernières années, et même bien avant, notamment pour m'avoir aidée à trouver un logement accessible qui m'a permis de réaliser cette thèse.

Ma directrice, Valérie Gouet-Brunet, pour ton niveau d'exigence qui me pousse à toujours faire mieux, pour ta rigueur constante et pour le temps précieux que tu as consacré à ma thèse.

Mon encadrant, Laurent Caraffa, pour ton soutien au cours de ces trois dernières années, à la fois technique, pour m'avoir aidé à démêler les lignes de mon code et humain, dans nos discussions et jusque dans ma première expérience en avion.

Je souhaite remercier tout le laboratoire LaSTIG, qui m'a accueilli au cours de ces trois dernières années et m'a permis de travailler dans un environnement à la fois stimulant et convivial, même si mes pauses du midi s'éternisent parfois à cause de mon rythme de repas très épicurien.

Merci à tous les membres permanents pour les exemples inspirants de chercheurs qu'ils représentent : Ana-Maria, Arnaud, Bénédicte, Bruno, Catherine, Clément, Emmanuel, Ewelina, Gérald, Jacques, Julien, Lâmân, Laurence, Loïc, Manchun, Marc, Maria-Jesus, Mathieu, Nathalie, Paul et Wu Teng.

Merci aussi à Karine pour ta gentillesse et ton aide précieuse : sans toi, de nombreuses démarches et impressions de thèse tardives n'auraient pas été possibles.

Je souhaite également remercier tous les membres non permanents, stagiaires, doctorants, ingénieurs et post-doctorants, pour avoir contribué à l'ambiance générale du laboratoire : Colin, Christian, Lu-Lin, Raphaël,

Sami, Justin, Vanessa, Paul, Christelle, Esteban, Matilde, Mouhamadou, Salomé, Quentin et Will Le boss09.

Merci à tous ceux qui m'ont servi de modèles à mon arrivée : Azelle, Charles, Damien, Emile, Grégoire, Helen, Luc, Melvin et Romain.

Merci à Alexane pour ton attention constante au bien-être des autres et pour l'organisation exemplaire du LYRD 2024, que tu as menée comme une cheffe.

Thank you, John, for all our discussions, from our thesis work to games and everyday life, and for occasionally forcing me to speak English (and, more importantly, to actually speak in general)!

Mes nouveaux camarades de bureau, Pierre-Loïc, Camille et Florent Le Clerc, merci d'avoir apporté de l'ambiance dans un bureau qui a été bien studieux en 2025. Je vous souhaite le meilleur pour vos thèses, et je sais déjà que mes anciens écrans continueront à vous servir.

Merci à tous les doctorants de la promotion de l'Épingle (2022/2023 à 2025/2026) qui m'ont adoptée à posteriori: Solenn, Martin, Karim, Florent Geniet, Charly, Bérénice.

Merci à Solenn pour ton enthousiasme communicatif, que ce soit en parlant de cartes anciennes, de la Bretagne, des expositions Lego, les sorties touristiques Paris by night et l'organisation des soirées jeux de sociétés.

Merci à Martin pour ta bienveillance, ta générosité, tes mêmes, tes présentations de monuments historiques de Paris en anglais et tes conseils tout au long de ces trois dernières années.

Merci à Karim, d'avoir partagé mon bureau, pour ton extrême gentillesse, ta curiosité à l'égard de mes travaux et pour toutes nos discussions vidéoludiques.

Merci à Florent Geniet, pour toutes les discussions où tu as pu partager ta culture sur les demi-arêtes, la musique, les jeux vidéos, la taille de chaussures d'escalade, les parties de baby-foot et tes conseils en chemises.

Merci à Charly, dont la passion communicative et l'énergie pour la cartographie m'impressionnent toujours, ainsi que pour ton engagement impressionnant au service des données ouvertes.

Merci à Laura, d'être la sérénité incarnée, pour ta bienveillance et ta disponibilité, les tips de l'ENSG et pour toute la logistique de mon pot de thèse. Encore bravo pour ton niveau de décoration d'halloween.

Merci à Bérénice dont la bonne humeur, l'intérêt pour mes projets en pixel art, ainsi que les nombreuses discussions et sorties geek ont grandement contribué à enjoliver ces années de thèse. Je fais maintenant beaucoup plus attention à où je laisse mon pain de cantine.

Je souhaite également remercier ma famille : leurs sacrifices discrets et silencieux, que peu de personnes remarquent mais dont j'ai pleinement

conscience, m'ont permis de bénéficier des meilleures conditions pour entreprendre et mener à bien cette thèse.

"For those who come after."

Abstract

This thesis focuses on indexing and retrieval in 3D point clouds for large-scale place recognition, and more precisely on its acceleration. The objective is to search for point clouds similar to a query cloud in a database of georeferenced point clouds by establishing a representation of the point cloud and then searching for it in the database to obtain a list of nearest neighbors ranked by similarity according to distance metrics.

State-of-the-art methods in place recognition from point clouds primarily focus on building representations of the cloud rather than on the efficient search for similar candidates. This second step typically relies on exhaustive search or better k-d trees, making it computationally expensive when the volume of the point cloud reference database increases or is inefficient in high dimensions. This is, however, a typical situation in current geographic information applications, such as large-scale place recognition or reference updates. In contrast, the literature on the subject, mainly in robotics, primarily focuses on loop closure scenarios, often limited to the scale of a street or neighborhood. To address this problem, we have chosen to draw inspiration from the state of the art in information retrieval. This thesis makes three contributions to the problem of large-scale place recognition, relying exclusively on the geometric information contained in point clouds.

Our first contribution, called DSI-3D, is based on the Differentiable Search Index (DSI) approach, which can generate identifiers for relevant documents in response to a textual query. To associate such indices with 3D data, we draw inspiration from Vision Transformers, such as GIT, which were originally designed for image captioning. By replacing its image encoder with a 3D encoder, our method learns to generate indices from point clouds, enabling similarity searches of point clouds in constant inference time. Several encoding strategies, some specialized for geolocation, have been studied and evaluated using the standard datasets from the state of the art in robotics (KITTI).

Our second contribution is the exploitation of LiDAR-based territorial mapping data, LiDAR HD from the IGN, for place recognition tasks. This dataset, present throughout the entire French territory, has acquisition characteristics that differ significantly from those of mobile mapping. The use of DSI-3D on this dataset thus required fine-tuning of the 3D encoder and adaptation strategies. To assess the generalization capability of DSI-3D, two areas of Paris were selected to form the new evaluation dataset.

Our final contribution explores two improvements for retrieval in 3D point clouds, aimed at scaling to larger areas and at improving the quality of the retrieval. First, we investigated a retrieval strategy based on Mixtures of Experts (MoE), which divides the model into distinct sub-models (experts), each specialized in a subset of the database. Second, we studied the application of a re-ranking mechanism to improve point cloud place recognition performance by re-ordering the list of candidates based on their local feature similarity.

This work is intended to open perspectives on improving large-scale retrieval in 3D point clouds for place recognition, particularly with regard to the use of generative search, while stimulating future research on the integration of multimodal methods for place recognition.

Keywords: Indexing and Retrieval of 3D Scenes, Place Recognition, Information Retrieval, Deep Learning.

Résumé

Cette thèse porte sur l'indexation et l'accélération de la recherche dans les nuages de points 3D, pour la reconnaissance de lieux à grande échelle. L'objectif est de rechercher des nuages similaires à un nuage requête dans une base de données de nuages géoréférencés, en établissant une représentation du nuage, puis en la recherchant dans la base pour obtenir une liste de candidats triés par ordre de similarité.

Les méthodes de l'état de l'art en reconnaissance de lieux à partir de nuages de points se concentrent principalement sur la construction de la représentation des nuages, plutôt que sur une recherche efficace de candidats similaires. Cette seconde étape repose généralement sur une recherche exhaustive, au mieux avec un arbre k-d, ce qui la rend coûteuse en temps de calcul lorsque le volume du référentiel de nuages de points augmente, ou moins intéressante pour des vecteurs de grande dimension. C'est pourtant une situation typique dans les applications actuelles de l'information géographique, telles que la reconnaissance de lieux à grande échelle ou la mise à jour des référentiels. À l'inverse, la littérature sur le sujet en robotique se concentre majoritairement sur des scénarios de fermeture de boucle, souvent limités à l'échelle d'une rue ou d'un quartier. Pour répondre à ce problème, nous avons choisi de nous inspirer des méthodes de recherche d'information de l'état de l'art. Cette thèse apporte trois contributions à la problématique de la reconnaissance de lieux à grande échelle, en s'appuyant exclusivement sur les informations géométriques contenues dans les nuages de points.

Notre première contribution, nommée DSI-3D, repose sur l'approche du Differentiable Search Index (DSI), capable de générer l'identifiant des documents pertinents en réponse à une requête textuelle. Pour associer ce type d'indices aux données 3D, nous nous inspirons des Vision Transformers tels que GIT, conçus initialement pour la génération automatique de descriptions d'images. En remplaçant l'encodeur d'image par un encodeur 3D, notre méthode apprend à générer des indices à partir de nuages de points, ce qui permet une recherche de nuages de points similaires en temps d'inférence constant. Plusieurs stratégies d'encodage, spécialisées pour la géolocalisation, ont été étudiées et évaluées sur un jeu de données de l'état de l'art en robotique (KITTI).

Notre seconde contribution consiste en l'adaptation des données de cartographie territoriale LiDAR HD de l'IGN, présentes sur tout le territoire français, à des tâches de reconnaissance de lieux. Ce jeu de données présente des caractéristiques d'acquisition sensiblement différentes de celles de la cartographie mobile, mais il offre la possibilité d'être étendu à l'échelle nationale. Cette évaluation a nécessité un nouvel entraînement de l'encodeur 3D, impliquant un découpage du jeu de données en ensembles d'apprentissage et d'évaluation.

Notre dernière contribution explore deux améliorations de DSI-3D, conçues pour s'adapter aux grands jeux de données. Premièrement, nous étudions une stratégie de recherche basée sur les mélanges d'experts (Mixtures of Experts, MoE). Leur principe consiste à scinder un modèle d'apprentissage en plusieurs sous-modèles indépendants, appelés experts, chacun étant spécialisé dans un sous-ensemble spécifique de nuages de points. Deuxièmement, dans le cadre d'un stage de Master, nous étudions l'application d'un mécanisme de reclassement pour améliorer les performances de la reconnaissance de lieux dans les nuages de points, en réorganisant la liste des candidats selon la similarité de leurs caractéristiques locales.

Ce travail vise à ouvrir des perspectives sur l'amélioration de la reconnaissance de lieux à grande échelle à partir de nuages de points 3D, en particulier en ce qui concerne l'utilisation de la recherche générative, tout en stimulant les recherches futures sur l'intégration de méthodes multimodales pour la reconnaissance de lieux.

Mots-clés: Indexation et recherche de scènes 3D, Reconnaissance de lieux, Recherche d'information, Apprentissage profond.

Contents

List of Figures **15**

List of Tables **17**

List of Acronyms **18**

Publications **19**

1 Introduction **20**

Introduction **21**
1.1 Motivation . . . 21
1.2 Contributions and outline of the thesis . . . 23
1.3 Context . . . 24

2 Related work **25**
2.1 Introduction . . . 25
2.2 Place recognition methods . . . 26
2.3 Point cloud representation . . . 27
2.3.1 Local features . . . 28
2.3.2 Global features . . . 30
2.3.3 Focus on LoGG3D-Net . . . 34
2.3.4 Place recognition datasets . . . 36
2.4 Retrieval accelerating techniques . . . 37
2.4.1 Inverted index methods . . . 38
2.4.2 Tree-based methods . . . 39
2.4.3 Hashing-based methods . . . 39
2.4.4 Graph-based methods . . . 39
2.4.5 About search time complexity and approximations . . . 40
2.5 Information retrieval . . . 41
2.5.1 Text-based Bag-of-Words retrieval . . . 41
2.5.2 Dual encoders . . . 42
2.5.3 Differentiable Search Index . . . 43
2.6 Conclusion . . . 46

3 DSI-3D, an adaptation of the Differentiable Search Index to 3D data **47**
3.1 Introduction . . . 47
3.2 Vision transformers . . . 48
3.2.1 Motivations . . . 48
3.2.2 Generative image-to-text transformers . . . 48
3.3 DSI-3D . . . 50
3.3.1 Point cloud encoder . . . 50

3.3.2 Point cloud indexing . . . 51
3.4 Conclusion . . . 56

4 Evaluation of DSI-3D for 3D point cloud retrieval 58
4.1 Introduction . . . 58
4.2 KITTI Dataset . . . 59
4.2.1 Overview of the KITTI dataset . . . 59
4.2.2 Building a new KITTI sequence . . . 60
4.2.3 Dataset splitting . . . 61
4.3 Experimental setup . . . 62
4.3.1 Contrastive learning . . . 62
4.3.2 Vocabulary restrictions . . . 63
4.3.3 Evaluation protocol . . . 63
4.4 Evaluation of DSI-3D . . . 64
4.4.1 Indexing strategies . . . 65
4.4.2 Positioning DSI-3D among retrieval approaches . . . 67
4.4.3 Complexity analysis . . . 68
4.5 Experimenting on DSI-3D hyperparameters . . . 70
4.5.1 Beam search width and vocabulary restrictions effects . . . 70
4.5.2 Optimizing training configuration . . . 72
4.6 Conclusion . . . 73
4.6.1 Overall assessment . . . 73
4.6.2 Chapter conclusions . . . 74

5 LiDAR HD as a place recognition dataset 76
5.1 Introduction . . . 76
5.2 LiDAR HD dataset overview . . . 77
5.2.1 Acquisition and characteristics . . . 77
5.2.2 Dataset preprocessing and analysis . . . 79
5.3 LoGG3D-Net applied to LiDAR HD . . . 82
5.3.1 Evaluation of the pre-trained LoGG3D-Net model . . . 82
5.3.2 Fine-tuning LoGG3D-Net on LiDAR HD . . . 83
5.3.3 Evaluation of the fine-tuning . . . 84
5.3.4 Training data augmentation . . . 91
5.4 Conclusion . . . 91

6 Extending DSI-3D 93
6.1 Introduction . . . 93
6.2 Mixture of Experts . . . 93
6.2.1 Overview of current methods . . . 94
6.2.2 Adapting MoE for DSI-3D . . . 96
6.3 Re-ranking . . . 106
6.3.1 Related work . . . 106
6.3.2 Re-ranking on LiDAR HD . . . 110
6.4 Conclusion . . . 112

Conclusion and Perspectives 114

A LoGG3D-Net descriptors robustness 118

B Distinctiveness of LoGG3D-Net descriptors 121

Bibliography **123**

Résumé détaillé de la thèse en français **134**

List of Figures

1.1 View of the LiDAR HD from Vieux-Port, Marseille . . . 20
1.2 LiDAR-based place recognition pipeline . . . 21

2.1 Example of modalities differing distributions under varying conditions . . . 27
2.2 Spin-Images method . . . 29
2.3 ScanContext method . . . 31
2.4 Locus method . . . 32
2.5 LoGG3D-Net method . . . 34
2.6 LoGG3D-Net binary classification . . . 36
2.7 Dual encoder method . . . 43
2.8 DSI method . . . 43
2.9 Example of semantically structured string identifiers . . . 45

3.1 GIT method . . . 49
3.2 Proposed architecture of DSI-3D . . . 50
3.3 Modified GIT method for 3D data . . . 51
3.4 Positional Structured identifier with GNSS method . . . 53
3.5 Hilbert curve iterations . . . 54
3.6 Effects of indexing strategies on revisits . . . 56

4.1 KITTI trajectories . . . 60
4.2 Sequence 22 construction . . . 61
4.3 DSI-3D binary classification . . . 64
4.4 Top-1 retrieval examples . . . 66
4.5 Retrieval time cost of exhaustive search . . . 69
4.6 Hyperparameter optimization on KITTI . . . 72
4.7 Hyperparameter optimization (training A complete) . . . 73

5.1 LiDAR HD acquisition progress . . . 78
5.2 Aerial view of Paris areas . . . 79
5.3 LiDAR HD preprocessing . . . 80
5.4 Visualization of LiDAR HD different viewing-angle . . . 80
5.5 LiDAR HD revisits distribution . . . 81
5.6 LiDAR HD filtering of incomplete scenes . . . 82
5.7 LiDAR HD training and evaluation set . . . 84
5.8 Visualization of voxel size . . . 84
5.9 Retrieval error repartition on $Paris_{West}$ and east . . . 87
5.10 Retrieval error repartition per flight ID, on Paris East . . . 88
5.11 Retrieval error repartition per flight ID, on Paris West . . . 90

6.1 Mixture of Experts method . . . 97
6.2 Coordinate rescaling for indexing . . . 98
6.3 Experts distribution across both areas . . . 99

6.4 Hyperparameter optimization on LiDAR HD 100
6.5 Hyperparameter optimization on LiDAR HD with a new tokenizer 101
6.6 Retrieval times across all database sizes 106
6.7 SpectralGV method 109
6.8 Local descriptors selected by SpectralGV 109
6.10 Rotation augmentation effect 112

A.1 LoGG3D-Net robustness evaluation under data augmentation 119

B.1 Étapes d'indexation et de recherche, du DSI 137
B.2 Vue aérienne de la couverture spatiale des deux zones sélectionnées de Paris. 140
B.3 Découpage des tuiles LiDAR HD 141
B.4 Répartition des erreurs de recherche 142
B.5 Mixture d'experts pour DSI-3D 143
B.6 Répartition des experts dans les deux zones 144

List of Tables

2.1 Summary of point cloud representation methods of the state of the art. . . 33
2.2 Average query complexity of exhaustive and approximate nearest neighbor search methods. 40

3.1 Example of a naively structured string identifier 52
3.2 Example of a semantically structured string identifier 53
3.3 Example of a positional structured string identifier (GNSS) 54
3.4 Example of a positional structured string identifier (Hilbert curve) 55

4.1 KITTI sequences length and their number of points and revisits. 59
4.2 Distribution of the KITTI sequence datasets into a training, validation and evaluation set. 62
4.3 Contrastive loss function 63
4.4 Evaluation across KITTI sequences using various indexing strategies for docids, based on LoGG3D-Net descriptors, using the $F1_{max}$ metric. 65
4.5 Evaluation across KITTI sequences using various indexing strategies (exact and approximate), based on LoGG3D-Net descriptors, using the $Hits@1$ metric. 65
4.6 Worst Top-1 retrieval errors (in meter), returned by DSI-3D for each KITTI sequence. The best results are bold and the second ones are underlined. . . 67
4.7 Evaluation across KITTI sequences using various indexing strategies (exact and approximate), based on LoGG3D-Net descriptors, using the $F1_{max}$ metric. 67
4.8 Evaluation across KITTI sequences using various indexing strategies (exact and approximate), based on LoGG3D-Net descriptors, using the $Hits@1$ metric. 68
4.9 Top-10 retrieval time 69
4.10 Beam search width effect on $F1_{max}$ metric 70
4.11 Beam search width effect on $Hits@1$ metric 70
4.12 Beam search width effect on average retrieval time 70
4.13 Vocabulary restrictions effect on $F1_{max}$ metric 71
4.14 Vocabulary restrictions effect on $Hits@1$ metric 71
4.15 Vocabulary restrictions effect on average retrieval time 71
4.16 Optimization of DSI-3D training hyperparameters 72
4.17 Overview of retrieval performance on KITTI sequence 00 74

5.1 Overview of 3D place recognition dataset coverage. Num_{scenes} refers to the total number of point clouds in a dataset. 81
5.2 Evaluation on LiDAR HD of the LoGG3D-Net model pre-trained on KITTI 83
5.3 Fine-tuned training hyperparameter evaluations of LoGG3D-Net on LiDAR HD 85
5.4 Fine-tuned training hyperparameter C, evaluated on standard and strict distance thresholds, on LiDAR HD. 86

5.5 LoGG3D-Net performance, trained on LiDAR HD 89
5.6 LoGG3D-Net performance, trained on LiDAR HD with rotation 91

6.1 Point cloud distribution per expert. 99
6.2 Training hyperparameters of DSI-3D on LiDAR HD 100
6.3 Training hyperparameters of DSI-3D on HDL 101
6.4 Evaluation of local DSI-3D experts and state-of-the-art approaches across multiple LiDAR HD expert regions with fine-tuned LoGG3D-Net descriptors, using the $F1_{max}$ metric. 102
6.5 Evaluation of local DSI-3D experts and state-of-the-art approaches across multiple LiDAR HD expert regions with fine-tuned LoGG3D-Net descriptors, using the $Hits@1$ metric. 102
6.6 LiDAR dataset distribution for gate training. 103
6.7 Performance of the gating network trained as a multi-label classifier. 104
6.8 Evaluation of LiDAR HD retrieval, using the $Hits@1$ metric 104
6.9 Evaluation of LiDAR HD retrieval, using the $F1_{max}$ metric 105
6.10 Top-10 retrieval time 105
6.11 Comparison of keypoint detectors on KITTI dataset. Results are taken from their respective papers. The best results are in bold. 110

B.1 LoGG3D-Net's descriptors distance ratio, fine-tuned on LiDAR HD 121
B.2 LoGG3D-Net's descriptors distance ratio of incorrect matches, fine-tuned on LiDAR HD 122
B.3 Évaluation sur plusieurs séquences KITTI à l'aide de diverses stratégies d'indexation, basées sur les descripteurs LoGG3D-Net, à l'aide de la métrique $F1_{max}$. 138
B.4 Évaluation sur plusieurs séquences KITTI à l'aide de diverses stratégies d'indexation, basées sur les descripteurs LoGG3D-Net, à l'aide de la métrique $Hits@1$. 139
B.5 Comparatif des temps de recherche 139
B.6 Résultat de la recherche exhaustive des descripteurs de LoGG3D-Net réentraîné sur le LiDAR HD 142
B.7 Évaluation de la recherche de données LiDAR HD à l'aide de la métrique $Hits@1$ 144
B.8 Évaluation de la recherche de données LiDAR HD à l'aide de la métrique $F1_{max}$ 145
B.9 Comparatif des temps de recherche d'un Top-10 145

List of Acronyms

ANN	Approximate Nearest Neighbor
BERT	Bidirectional Encoder Representations from Transformers
BOS	Beginning-of-Sequence
CNN	Convolutional Neural Network
CPU	Central processing units
Docid	Document Indentifier
DSI	Differentiable Search Index
EOS	End-of-Sequence
ePN	Eigen-value Power Normalization
FFN	Feed-forward Neural Network
FPN	Feature Pyramid Network
GIT	Generative Image-to-text Transformer
GPU	Graphics processing units
GNSS	Global Navigation Satellite System
HNSW	Hierarchical Navigable Small World
k-NN	k nearest neighbors
LiDAR	Light Detection and Ranging
LLM	Large Language Model
LSH	Locality Sensitive Hashing
NSW	Navigable Small World
SfM	Structure-from-Motion
SLAM	Simultaneous Localization and Mapping
ViT	Vision Transformers

Publications

Journals

Under review Chahine-Nicolas Zede, Laurent Caraffa, Valérie Gouet-Brunet, "Improving retrieval in large-scale 3D point clouds", Submission to the ISPRS Journal of Photogrammetry and Remote Sensing.

International Conferences

1. Chahine-Nicolas Zede, Laurent Caraffa, Valérie Gouet-Brunet, "DSI-3D: Differentiable Search Index for Point Cloud Retrieval", 22nd International Conference on Content-Based Multimedia Indexing (CBMI'25), Oct 2025, Dublin, Ireland, pages 1-8.

National Conferences

1. Chahine-Nicolas Zede, Laurent Caraffa, Valérie Gouet-Brunet, "Fast 3D point clouds retrieval for Large-scale 3D Place Recognition", ORASIS, journées francophones des jeunes chercheurs en vision par ordinateur, Jun 2025, Le Croisic, France, pages 1-9.

Chapter 1

Introduction

The growing availability of large-scale 3D data for modeling cities and even entire countries as digital twins, such as LiDAR HD[1], a French LiDAR (Light Detection And Ranging) acquisition program that provides high-density point clouds on a country scale, as seen in Figure 1.1, has opened the door to a wide range of applications for understanding the environment and its evolution (GOOS et al., 2011; Stoter et al., 2016). This program is led by the French mapping agency IGN, the national institute of geographic and forest information. Among these applications, we can cite autonomous navigation (Chen et al., 2024; Cadena et al., 2017), where a key aspect of scene understanding is the ability to determine a system's location within a reference map.

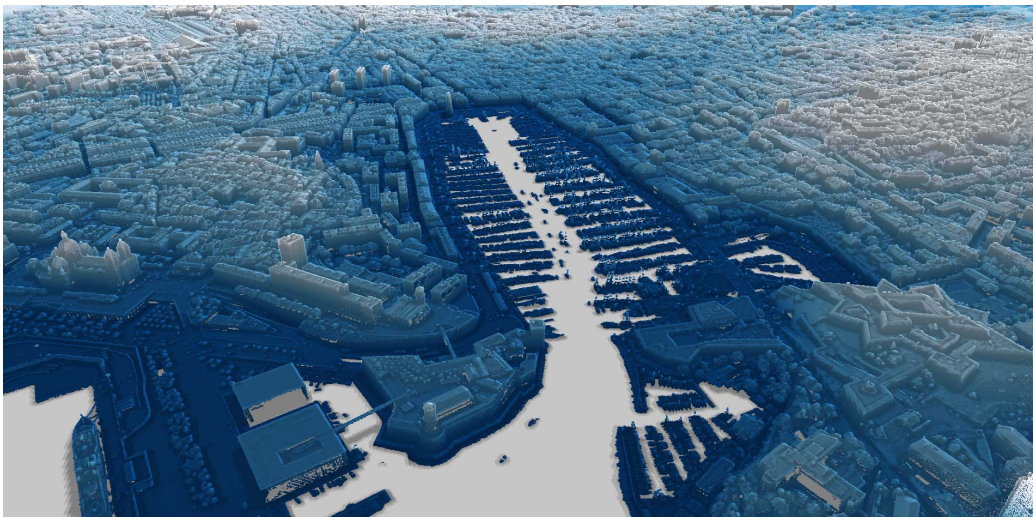


Figure 1.1: View of the LiDAR HD airborne acquisition campaign over Vieux-Port, Marseille, France. Figure taken from the IGN website [2].

Place recognition refers to the task of identifying the most relevant previously visited location depicted in a query (Yin et al., 2025). This identification can be achieved by pairing the query with a discrete identifier linked to a georeferenced database entry, which may estimate a 2D, 3D position, or even 6 degrees of freedom pose. The georeferenced database and the query are usually images of 3D representations. In this study, we focus on 3D representation in the form of point clouds. For the 3D place recognition community, location retrieval using an element from the database is commonly employed as a legacy of robotics and loop closure detection.

In a sense, 3D place recognition can be viewed as a large-scale, data-driven counterpart to the game GeoGuessr, in which players attempt to geolocate a randomly selected

[1] https://geoservices.ign.fr/lidarhd
[2] http://ign.fr

position from street view by observing the surrounding environment. Instead of a human player guessing a location from images, a model has to recognize a place purely from the geometric structure of its point cloud.

When the location to find is unknown, it is commonly framed as a two-stage process involving indexing and retrieval (Lowry et al., 2015) (see Figure 1.2).

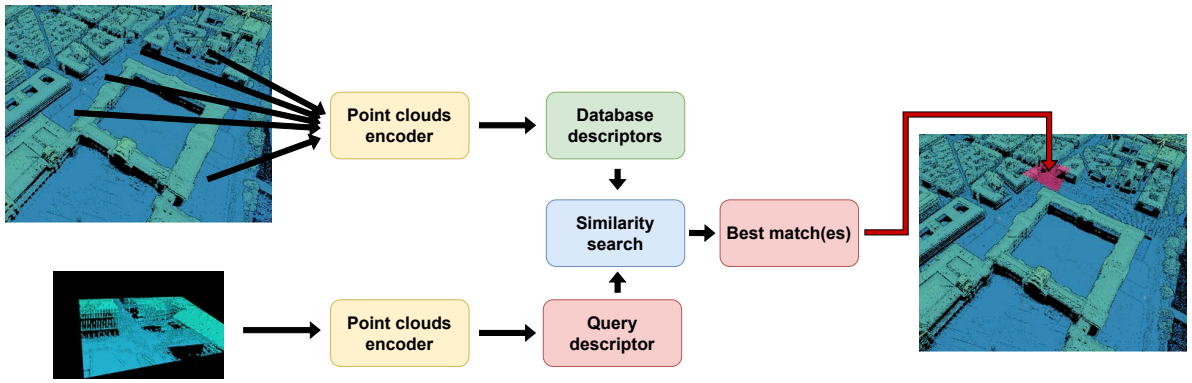


Figure 1.2: A typical LiDAR-based place recognition indexing and retrieval pipeline. For indexing, the database is split into smaller point clouds, each encoded into a descriptor. For retrieval, the query point cloud is similarly encoded and compared against the database descriptors. Similarity matching identifies the best candidates, from which the position of the query is estimated.

Indexing involves organizing the reference database to facilitate efficient searching. This may include encoding 3D scenes with descriptors, a compact vector representation of the geometry (Uy and Lee, 2018). This also includes building indexing structures from the database, such as tree structures (like k-d trees (Bentley, 1975)) or hash tables (Indyk and Motwani, 1998; Douze et al., 2024) for efficient retrieval. **Retrieval** corresponds to the querying of an indexing structure, where the query, described as the items of the database, is compared against these items in order to return one or more nearest neighbors, considered as the best candidate matches according to a distance metric (Muja and Lowe, 2009). In the context of scenes represented with 3D point clouds, this involves matching a query point cloud against a reference database of georeferenced point clouds, in order to retrieve the most similar scenes.

1.1 Motivation

This section presents the motivations behind this work, outlining both the scientific challenges in 3D retrieval-based place recognition and the practical applications of point cloud retrieval.

Scientific challenges

This thesis focuses on addressing a limitation of 3D place recognition, the scalability of the retrieval step in point clouds. While significant progress has been made in the design of robust point cloud descriptors as indexing techniques, the retrieval phase, i.e, the process of efficiently identifying relevant matches from large databases, remains relatively underexplored. In practice, most approaches rely on exhaustive search or k-d trees for nearest-neighbor search (Vidanapathirana et al., 2022; Kim and Kim, 2018; Uy and Lee, 2018; Komorowski, 2021; Guan et al., 2023). An exhaustive search compares the query descriptor against every element in the database sequentially, which severely limits real-time applicability, as the retrieval time grows linearly with the dataset size. To mitigate

this, k-d trees, the indexing strategy the most commonly used in the literature of place recognition with 3D point clouds (Uy and Lee, 2018; Xu et al., 2021), partition the k-dimensional feature space using recursive hyperplane splits, effectively reducing the search space and the number of distance computations. However, in high-dimensional settings, such as those used for place recognition, k-d trees become inefficient. The sparsity of high-dimensional spaces, often referred to as the curse of dimensionality (Pan et al., 2024), leads to the proliferation of nearly empty partitions, forcing the algorithm to explore multiple subspaces. As a result, the retrieval time approaches that of an exhaustive search.

To accelerate the retrieval phase, we explored approaches from text-based information retrieval. Information retrieval refers to the task of returning a list of documents (e.g., images, texts, or indexes) relevant to a given search query. In a broader context beyond text, these items can include *multimedia files*, such as text passages, images, or web pages. Information retrieval has been extensively studied and has proven highly effective in powering modern web search engines (Brin and Page, 1998). The objective of this thesis is to design scalable retrieval methods, inspired by DSI (Tay et al., 2022), for large-scale 3D place recognition, leveraging an information retrieval approach. The principle of DSI is to train a language model to generate meaningful document IDs (or, in our case, place IDs) in response to a query. The main advantage of this approach is its constant retrieval time, which remains independent of the database size, as all corpus information is parameterized within the model.

Applications

Retrieval in 3D point clouds for place recognition is a core process in several topical applications. A major application is autonomous navigation (Chen et al., 2024; Cadena et al., 2017), where 3D place recognition helps reduce SLAM accumulated drift errors and improve loop closure detection. While visual place recognition methods are more mature, approaches based on LiDAR offer advantages in terms of robustness to illumination changes and viewpoint variations (Uy and Lee, 2018; Maddern et al., 2017), making them more reliable in challenging, dynamic outdoor conditions.

In dense urban areas, GNSS[3] signals often suffer from multipath effects and occlusions caused by urban canyons (Groves, 2011; Wang et al., 2012). 3D-based place recognition can serve as a complementary source to correct systems relying on GNSS (Global Navigation Satellite System) by recognizing previously seen places and enabling better pose estimation.

3D point cloud retrieval can be used for digital twin maintenance by detecting changes (Stilla and Xu, 2023), where reference point clouds are updated according to new acquisitions. This is particularly useful in applications such as land surveying, infrastructure inspection, or monitoring structural changes over time, allowing for the accurate detection of modifications in the physical (Lague et al., 2013).

Another motivation behind this work lies in the verification of multimedia metadata. In the context of misinformation and fake news detection (Nakov et al., 2021), confirming whether a video or image was actually captured at the claimed location is essential. Thanks to advances in Structure-from-Motion (SfM) (Hartley and Zisserman, 2003; Özyeşil et al., 2017) techniques, it is now possible to reconstruct 3D point clouds from videos or image sequences (Ullman, 1979). This opens the door to location-based content verification via 3D retrieval methods.

[3]Global Navigation Satellite System

1.2 Contributions and outline of the thesis

This section provides a summary of the contributions of this thesis and introduces its overall structure.

Contributions

The primary contributions of this dissertation is to address the scalability challenge of 3D place recognition, with a particular focus on accelerating the retrieval phase while preserving retrieval accuracy. The objectives of this work can be divided as follows:

- Propose a new approach based solely on the geometric information contained in a query point cloud without resorting to auxiliary modalities.
- Accelerate the retrieval phase of current 3D place recognition methods.
- Exploit High-Density (HD) LiDAR data for large-scale location recognition by adapting coverage, density, and revisit scenarios.
- Design a hierarchical retrieval architecture capable of processing larger datasets and suitable for updating datasets and models.
- Apply a re-ranking mechanism to promote the best candidates and as a solution to compensate for the performance drop caused by the approximative search.

This thesis focuses exclusively on point cloud place recognition based on their geometric information, as we believe this area already presents substantial scientific challenges and practical applications. By discarding complementary information such as color, intensity, or images, the proposed approach becomes less dependent on the acquisition modality and is thus more broadly applicable across different sensors and acquisition techniques (LiDAR, photogrammetric reconstruction, SfM).

Thesis outline

This dissertation consists of 7 chapters organized as follows.

Chapter 1: Introduction outlines the motivation and context of this research, introduces the central problem of large-scale 3D place recognition, and summarizes the key objectives and contributions of the dissertation.

Chapter 2: Related work reviews the state of the art in LiDAR-based place recognition, highlighting the evolution of methods from early local feature aggregation to modern deep learning approaches. It further examines retrieval acceleration techniques, with a particular focus on approximate nearest neighbor search, and discusses advances in information retrieval that inspire our proposed methodology. Together, these elements establish the foundation upon which the contributions of this dissertation are built.

Chapter 3: DSI-3D, an adaptation of the Differentiable Search Index to 3D data presents our first contribution. It is the adaptation of the DSI framework to 3D data. This includes the design of novel indexing strategies tailored to point clouds and the integration of Vision Transformers (ViT), a deep neural network architecture based on self-attention mechanisms used for image captioning, as the backbone for scene representation.

Chapter 4: Evaluation of DSI-3D for 3D point cloud retrieval presents and evaluates the application of our first contribution on a mobile mapping dataset. This

chapter also introduces the design choices used for training and evaluating DSI-3D, and opens with a study of the impact of varying hyperparameters.

Chapter 5: LiDAR HD as a place recognition dataset presents the adaptation of the high-density LiDAR dataset provided by the IGN (French National Geographic Institute) into an extensible reference dataset for point cloud retrieval. This change of database also implies retraining the encoding method for this new modality. Then, the new encoding model has been validated in two areas of Paris.

Chapter 6: Extending DSI-3D explores several strategies to enhance the applicability of DSI-3D to large-scale datasets. One solution involves adopting a MoE architecture, which partitions the retrieval space into specialized experts trained in different sectors, thereby improving both scalability and geographic coverage. Another line of improvement focuses on strengthening the robustness of DSI-3D by exploring existing re-ranking strategies that reorder the initial retrieval results to produce more accurate matches.

Chapter 7: Conclusion and perspectives summarizes the key contributions of the dissertation, reflects on the limitations of the proposed approaches, and outlines promising directions for future research.

1.3 Context

This work was financed by the French Ministry of the Armed Forces - Defense Innovation Agency (AID) and the National Institute of Geographic and Forest Information (IGN). It was carried out using HPC resources from GENCI-IDRIS (Grant 2024-AD011014030R1).

Chapter 2

Related work

Contents of this Chapter

2.1 Introduction . . . 25
2.2 Place recognition methods . . . 26
2.3 Point cloud representation . . . 27
2.3.1 Local features . . . 28
2.3.2 Global features . . . 30
2.3.3 Focus on LoGG3D-Net . . . 34
2.3.4 Place recognition datasets . . . 36
2.4 Retrieval accelerating techniques . . . 37
2.4.1 Inverted index methods . . . 38
2.4.2 Tree-based methods . . . 39
2.4.3 Hashing-based methods . . . 39
2.4.4 Graph-based methods . . . 39
2.4.5 About search time complexity and approximations . . . 40
2.5 Information retrieval . . . 41
2.5.1 Text-based Bag-of-Words retrieval . . . 41
2.5.2 Dual encoders . . . 42
2.5.3 Differentiable Search Index . . . 43
2.6 Conclusion . . . 46

2.1 Introduction

Methods for point cloud retrieval and, more broadly, for content-based retrieval have evolved quickly over the last two decades: from the introduction of local descriptors used with bag-of-words approaches in the early 2000s (Lowe, 2004) for images, to the emergence of deep convolutional representations designed for instance-level image retrieval in the mid-2010s (Babenko and Lempitsky, 2015; Tolias et al., 2015), and the end-to-end trainable model based on global descriptors (Uy and Lee, 2018) for 3D point clouds. This chapter proposes to survey their evolution and discuss their current limitations for large-scale retrieval. To complement this, a review of vector database management systems presents solutions for embedding-based retrieval acceleration (Pan et al., 2024). This chapter also proposes reviewing information retrieval approaches used by web search engines to retrieve relevant documents from a database in response to a text query.

First, in Section 2.2, we present a broader overview of place recognition methods, organized by sensing modality, in order to position our work within the existing literature. In Section 2.3, we examine the evolution of point cloud retrieval from early 3D

descriptors to recent deep learning–based representations. Next, Section 2.4 discusses retrieval acceleration techniques, focusing on approximate nearest neighbor search methods to address the scalability challenges inherent in large-scale 3D place recognition. Finally, Section 2.5 broadens the scope to the field of information retrieval, highlighting concepts and methodologies that inspire our proposed framework.

2.2 Place recognition methods

Place recognition refers to the task of identifying a revisited place observed under new conditions (Yin et al., 2025). It usually consists of recognizing a query observed from a camera in visual place recognition (Lowry et al., 2015) or from a range sensor, such as LiDAR, for LiDAR-based place recognition (Zhang et al., 2024). This recognition also implies the existence of a reference *map* against which the query can be matched. This section presents place recognition methods applied to multiple modalities, such as images, 3D data, and hybrid (2D+3D) representations, and extends to general multimodal place recognition.

Image-based place recognition

Early place recognition approaches focused on **2D–2D** matching, where a query image is compared against a database of reference images. These methods typically detect interest points in the query image and match their descriptors to those stored in the database to identify the most similar candidates. The first approaches relied on handcrafted local descriptors such as SIFT (Lowe, 2004) and SURF (Bay et al., 2008). To improve retrieval scalability, local features were subsequently aggregated into global representations using Bag-of-Words models (Gálvez-López and Tardos, 2012) and VLAD-based techniques, culminating in learning-based variants such as NetVLAD (Arandjelovic et al., 2016).

LiDAR-based place recognition

Monomodal **3D–3D** place recognition has gained significant traction over the past decade, driven by the increasing availability of LiDAR sensors and large-scale point cloud datasets. In this setting, places are represented as point clouds, unordered and sparse geometric structures, and the reference map is typically partitioned into smaller submaps to enable efficient comparison with a query scan. Retrieval often relies on computing a compact global descriptor for each point cloud and performing a similarity search in the resulting embedding space. Early handcrafted methods exploited the geometric organization of the point cloud, representing it in structured forms such as polar grids or matrices, as in 3D Gestalt (Bosse and Zlot, 2013) and ScanContext (Kim and Kim, 2018). With the advent of deep learning, models such as PointNetVLAD (Uy and Lee, 2018) enabled end-to-end learning directly from raw 3D points. More recent methods leverage advances in sparse 3D convolutions and attention mechanisms to improve robustness and discriminative power, including LoGG3D-Net (Vidanapathirana et al., 2022) and TransLoc3D (Xu et al., 2021).

Hybrid place recognition

Hybrid approaches such as **2D–3D** place recognition aim to localize a query image within a 3D map. An example of this idea has been deployed at scale in systems such as the crowdsourced Smapshot platform (Produit et al., 2018), in which users align historical images within large 3D point-cloud environments. A closely related principle is Structure-from-Motion pipelines (Hartley and Zisserman, 2003; Özyeşil et al., 2017), in which the

poses of input images are estimated by establishing 2D–3D correspondences with a reconstructed 3D scene. Less common, an example of 3D–2D matching is presented in (Tang et al., 2021), in which a LiDAR is compared to an image database. This is possible with the transformation of images into a set of 2D points.

General place recognition

Place recognition also extends to multimodal settings, enabled in part by recent progress in large language models (LLMs) (Achiam et al., 2023; Bi et al., 2024). For example, LLM-VPR (Lyu et al., 2024) uses an LLM to refine location estimates by reasoning over visual cues present in candidate images. Other approaches focus on fusing heterogeneous sensor modalities. The Unifying Local and Global Multimodal Features model (UMF) (García-Hernández et al., 2024) uses a cross-attention mechanism between visual and LiDAR encoders to enhance recognition in challenging environments, such as planetary-like volcanic terrains.

This thesis concentrates on monomodal, LiDAR-based place recognition, with a particular focus on accelerating the retrieval step. The principle of generative place recognition introduced, however, is general and could be adapted to alternative sensing modalities in future work.

2.3 Point cloud representation

Place recognition is the task of recognizing a previously observed scene from a new observation. Traditional place recognition techniques based on 2D imagery face significant challenges due to changes in illumination, weather, and seasonal appearances. In contrast, 3D LiDAR data are more robust than imagery to such variations (Uy and Lee, 2018) because they capture the underlying geometric structure of the environment, which is mostly invariant over time, as seen in Figure 2.1.

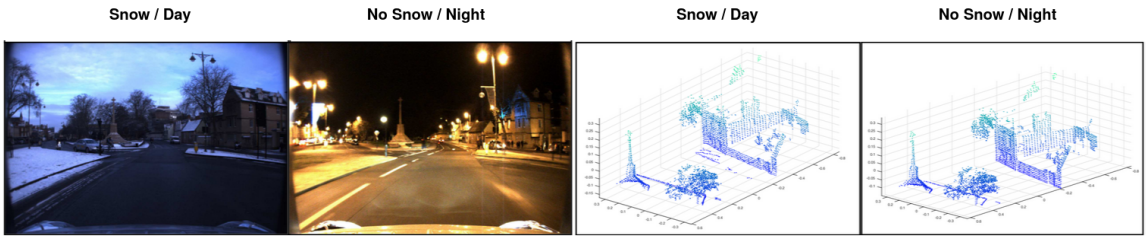


Figure 2.1: Example of differing distributions between two modalities under varying weather and illumination conditions. The left rows show images of a snowy daytime scene and the same location at night without snow, while the right rows show the same scenes captured by LiDAR. The acquisitions are from the Oxford RobotCar dataset (Maddern et al., 2017). Figure taken from (Uy and Lee, 2018).

Similar to Content-Based Image Retrieval (CBIR) (Pion et al., 2020), where images are first transformed into compact visual descriptors for efficient similarity search, 3D-based place recognition also requires converting raw scans into discriminative and compact representations. Efficient point cloud representations are, therefore, a key component of 3D scene understanding. These representations, often referred to as descriptors or embeddings, serve as implicit, compact encodings of a scene, typically in the form of vectors or matrices that capture its geometric structure in such a way that similar locations yield similar embeddings.

Point clouds tend to have irregular structures characterized by a high number of points with varying densities. Before computing their representations, they are generally preprocessed to achieve suitable properties for the chosen descriptor or model. Common operations include limiting the maximum number of points per cloud, either by random or geometric downsampling, or by cropping points outside a predefined bounding box centered on the point cloud's center to remove distant points. The density of the point clouds can also be uniformized using techniques like voxel[1] downsampling, which enforces a fixed resolution. Additionally, some description methods rely on enriched geometric information derived from the point cloud, such as meshes and normals.

These representations can be broadly categorized according to their spatial scope. Local descriptors capture fine-grained geometric details at the point or patch level, while global descriptors encode information about the entire point cloud. A complementary distinction lies in their computation methodology: descriptors can be handcrafted, relying on manually designed geometric measurements, or learning-based, where models are trained to extract features directly from data.

Section 2.3.1 and 2.3.2 trace the evolution of local and global features, respectively, from early handcrafted 2D adaptations to modern learning-based approaches. Section 2.3.3 details the principle of LoGG3D-Net, followed by Section 2.3.4, which presents commonly used place recognition datasets.

2.3.1 Local features

Local features are an encoding of the local point-cloud region, often seen as keypoints in content-based retrieval. They represent neighborhood information of a smaller region, such as a group of pixels in imagery, or, in 3D, a local group of points or voxels. We revisit the main categories below.

Handcrafted local features

Handcrafted local features typically detect keypoints first and then describe local information (e.g., image intensity gradients, 3D surface curvature, or point height distribution). Both stages rely on human-tuned parameters rather than learned models. These methods are commonly used in 3D shape retrieval applications, from point cloud registration and object recognition by surface matching, and finally, place recognition. We begin by reviewing 3D keypoint detection methods adapted from the imagery community.

3D Interest Point Detection

The detection of salient 3D points is often inspired by classical 2D feature detectors. For example, the **3D-Harris** method (Sipiran and Bustos, 2010) is an extension of the Harris corner detector in images (Harris et al., 1988) to 3D meshes. The original Harris operator identifies corners (or edges) by analyzing variations in image intensity gradients. To transfer this idea to 3D surfaces, 3D-Harris projects the local neighborhood of each point onto a tangent plane and then applies the Harris operator to obtain salient keypoints. Other methods, such as **3D-SIFT** (Godil and Wagan, 2011), can detect interest points while enriching them with representative local descriptors. The 3D-SIFT extends the principles of the 2D SIFT (Lowe, 2004) to 3D voxelized objects. In 2D, SIFT detects keypoints by constructing a scale space through repeated Gaussian smoothing and computing Difference-of-Gaussian images, considering extrema as salient points. Rotation invariance is achieved by assigning each keypoint a dominant orientation derived from

[1] A voxel is a cubic partition of space.

the local image gradient directions. 3D-SIFT adapts this process to volumetric data by applying 3D Gaussian filters to the voxel grid of the object to construct a 3D scale space and detect 3D salient points. With this process, an orientation histogram can be built to form a rotation-invariant descriptor.

Histogram-based local descriptors

Keypoint detection naturally leads to the representation of local features, which are used to establish correspondences between two scenes for tasks such as surface matching and point cloud registration. One classical approach to recognizing an observed shape within a reference database consists of matching these local features using histogram-based descriptors. One of the earliest approaches to 3D object recognition based on surface matching is **Spin-Images** (Johnson and Hebert, 2002). In this method, the recognition of meshed objects is performed by matching the observed surface of a query object against a database of referenced surfaces. The Spin-Images surface representation relies on 3D points and surface normals to build a 2D histogram for each point. Here histograms are images of the object that describes the distribution of neighboring points. The object is recognized by similarity comparison of those images with those of the database. This first approach effectively reduces a 3D matching problem to a matching problem of 2D representation, as seen in Figure 2.2

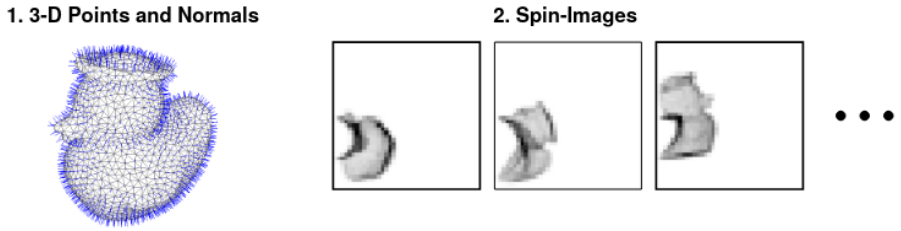


Figure 2.2: Surface representation by Spin-Images. The left image shows the set of 3D points with their associated surface normal, while the right image shows examples of Spin-image 2D histogram. Figure taken from (Johnson and Hebert, 2002).

The previous keypoint detection and local descriptor methods were designed for meshed or voxelized objects, tailored for object recognition tasks. The **Fast Point Feature Histogram (FPFH)** (Rusu et al., 2009) method focuses on point cloud registration between two point clouds. However, the surface normals are still required. FPFH is an accelerated variant of Point Feature Histogram (PFH) (Rusu et al., 2008) descriptor. Both follow the same descriptive principle for each point, the local feature is computed from neighboring surface normal information within a sphere with a predefined radius, and the normals of the neighboring points are used to describe local surface properties. These normal relationships are encoded into a histogram, where each bin represents the frequency of specific geometric surfaces (e.g, plane, cylinder, edges).

As mentioned earlier, **3D-SIFT** (Godil and Wagan, 2011) performs both keypoint detection and local feature construction for 3D shape retrieval of objects. The local features are clustered using K-means to construct a histogram-based representation for similarity search. Similarly, **Signature of Histograms of OrienTations (SHOT)** (Salti et al., 2014), inspired by the success of SIFT (Lowe, 2004) with its utilization of histograms at multiple steps, such as orientation definition and keypoint description. SHOT constructs a signature by combining multiple small orientation histograms based on the distribution of surface normals for 3D surface matching.

local descriptor in place recognition. Local handcrafted descriptors can also be used in the context of place recognition of point clouds. For example, 3D **Gestalt** (Bosse

and Zlot, 2013) encodes the local geometric structure around a keypoint by partitioning the surrounding space into bins in a polar grid and keeping point heights within each bin as values. Combined with a keypoint voting strategy (Bosse and Zlot, 2009), in which votes inform about the likelihood of a pair query-database element being relevant match. The database size can be reduced according to an empirical threshold score of negative matches, thus achieving sub-linear matching times in large-scale datasets.

Among handcrafted local descriptors, many approaches adapt 2D-inspired solutions to 3D problems (Johnson and Hebert, 2002; Godil and Wagan, 2011; Salti et al., 2014). While these methods have laid the groundwork for surface matching and place recognition, their reliance on handcrafted features often limits robustness and scalability (Zhang et al., 2024). These limitations motivate the adoption of learning-based approaches in which deep neural networks automatically learn more discriminative and adaptable features.

Learning-based local features

Large-scale visual recognition experienced a major breakthrough with the introduction of AlexNet (Krizhevsky et al., 2012), a deep convolutional neural network (CNN) that demonstrated the effectiveness of learned feature representations through feedforward architectures. This success marked a turning point, establishing convolutional models (LeCun et al., 1989) as powerful tools capable of implicitly capturing complex visual patterns. As a result, and with the democratization of GPUs[2], CNNs quickly gained popularity within the CBIR community (Dubey, 2021). Deep learning models, in general, are composed of multiple layers of interconnected neurons and are trained on large-scale datasets to automatically learn feature representations. An inherent challenge of applying deep learning to 3D point clouds is their unordered nature, as opposed to the sequential nature of text and images. **PointNet** (Qi et al., 2017a) is a pioneering deep learning architecture designed to process the unordered and irregular structure of raw point clouds. It learns to extract discriminative local features by applying a series of Multi-Layer Perceptrons (Rosenblatt et al., 1962; Rumelhart et al., 2013) to each point, followed by a symmetric aggregation function, here, a max-pooling function, which enables order-invariant representations by only selecting the most predominant features. To better capture local structure, **PointNet++** (Qi et al., 2017b) recursively applies PointNet, learning features at multiple scales.

While local descriptors remain valuable for tasks such as point cloud registration, reranking, and object detection, their reliance on keypoint detection makes them sensitive to variations in viewpoint and point cloud density. These limitations reduce their robustness in large-scale place recognition scenarios, where global descriptors provide a more stable and efficient alternative. An additional constraint of using local descriptors during retrieval is the need to compute and match a set of descriptors from the query to similar sets from the database.

2.3.2 Global features

Global descriptors aim to capture a representation of the entire scene without requiring keypoint detection. They encode the global geometry of the point cloud into a single, compact vector, usually by aggregating information from local features.

[2]Graphics processing units, which are much more efficient than CPUs, their predecessor, for parallel computation used in deep learning.

Handcrafted global features

ScanContext (Kim and Kim, 2018) reduces 3D point cloud retrieval to a 2D problem. It projects a point cloud into a compact 2D matrix in polar coordinates, where the rows represent radial bins and the columns represent angular sectors (see Figure 2.3). A column vector is computed by summing each column, and the best circular shift is found through a brute-force search to achieve rotational invariance. While effective, this method is sensitive to lateral displacements, such as lane changes between revisits.

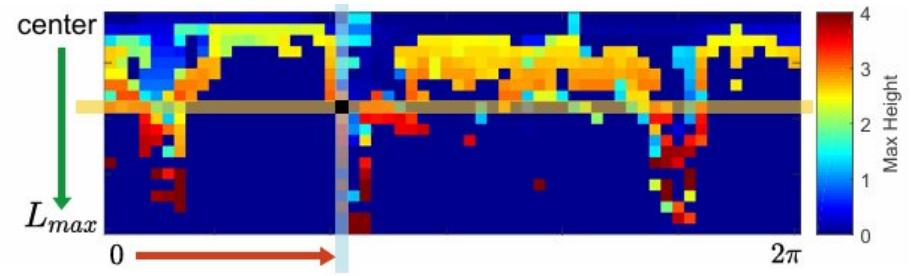


Figure 2.3: Example of a ScanContext 2D matrix obtained by projecting 3D points in 2D. The 3D space is partitioned by bins, ranging from 0 to 2π in azimuth, and from the *center* of the point cloud up to the maximum range L_{max}. The colors represent the maximum height value in each bin. Figure taken from (Kim and Kim, 2018).

ScanContext++ (Kim et al., 2021) improves this by using a sub-descriptor called a key to speed up candidate retrieval and by applying query augmentation to simulate shifted queries, thereby improving robustness to lateral shifts.

Learning-based global features

PointNetVLAD (Uy and Lee, 2018) was a pioneering end-to-end learning-based method for large-scale place recognition. It combines PointNet (Qi et al., 2017a), which directly processes unordered point clouds, with NetVLAD (Arandjelovic et al., 2016), a learnable aggregation layer that encodes local features into a single global descriptor. This approach outperformed earlier VLAD-based bag-of-words methods (Jégou et al., 2010), which were based on feature frequency. Building on this idea, the **Point Contextual Attention Network (PCAN)** (Zhang and Xiao, 2019) introduced a point contextual attention network in the NetVLAD layer, allowing the model to reweight local features based on their contextual importance. To further address the limitation of PointNetVLAD, namely its inability to capture the spatial distribution of local features, LPD-Net (Liu et al., 2019b) incorporated graph-based feature aggregation to explicitly model geometric relationships.

Segment-based descriptors. Another line of work focuses on segment-based methods that divide a scene into meaningful regions (e.g., roads, buildings, and vegetation). This provides an intermediate structured representation that captures topological relationships rather than treating the entire point cloud as a single entity or a set of local features. **Locus** (Vidanapathirana et al., 2021) aggregates these segment-based features into a global descriptor, enriched by the modeling of both topological relationships and temporal consistency (see Figure 2.4).

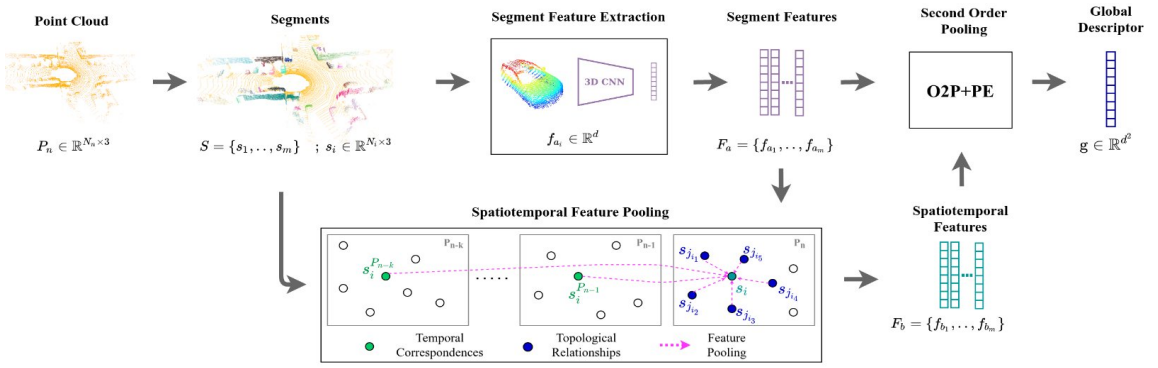


Figure 2.4: Locus pipeline method. The point cloud is segmented by topological classes, which are encoded into segment features, which are then enriched by spatio-temporal relationships. The global descriptor is the result of a second-order pooling on those segment features. Figure taken from (Vidanapathirana et al., 2021).

Convolutional-based descriptors. A limitation of processing 3D point clouds with a standard dense CNN is its computational cost, as most of the voxel grid is empty. Alternative convolutional architectures have therefore been explored, such as **Kernel Point Convolution (KPConv)** (Thomas et al., 2019), which applies kernel convolutions directly to each point. Since the points are not aligned with the kernel, convolution occurs on points with an added correlation coefficient based on their positions. Sparse convolutional networks, such as SparseConvNet (Graham et al., 2018) and the Minkowski Engine (Choy et al., 2019a), address this issue by operating only on non-empty voxels. This enables efficient 3D CNN computation on GPUs, reducing both memory usage and runtime while preserving high resolution in large scenes.

Building on this, **MinkLoc3D** (Komorowski, 2021) introduced the first 3D sparse convolution-based architecture for PR. It enhanced local feature extraction with a 3D feature pyramid network (Lin et al., 2017) for multi-scale convolution on sparsified point clouds and produced global descriptors using Generalized-Mean pooling (Radenović et al., 2018) instead of using NetVLAD. **LoGG3D-Net** (Vidanapathirana et al., 2022) builds on MinkLoc3D (Komorowski, 2021) by reusing sparse 3D convolutions and a U-Net backbone for point cloud embedding. It introduced a local consistency loss to improve the robustness of local descriptors under revisit scenarios. Global descriptors were obtained by aggregating local features through second-order pooling, followed by Eigen-value Power Normalization. The training objective combined a scene-level loss with the local consistency loss to jointly optimize global and local representations. Since we use it as the main descriptor in the remainder of the thesis, this approach is discussed in more detail in Section 2.3.3.

Transformer-based descriptors. While convolutional and graph-based networks have shown strong performance in encoding spatial relationships within point clouds, they remain limited by the difficulty of modeling long-range dependencies across a scene. The introduction of the Transformer architecture (Vaswani et al., 2017), a deep neural network that dynamically weighs the relevance of input sequence elements through self-attention mechanisms, resolves this limitation. While transformers were first introduced for translation tasks, they became more popular due to their versatility and reduced training time. Building on these ideas, **TransLoc3D** (Xu et al., 2021) addressed challenges specific to outdoor environments, such as discrepancies in object size, and the presence of moving objects (e.g., vehicles or pedestrians). It employed a transformer-based architecture (Vaswani et al., 2017) that reweighted features across multiple adaptive receptive fields, thereby capturing both local and global contextual information.

Cross-source descriptors. Previous methods typically relied on a single source

of data for both the reference dataset and the query set. Since aerial and ground LiDAR scans exhibit a significant representation gap, the point clouds obtained from these modalities differ in scope, coverage, density, and noise patterns. Inspired by diffusion models (Ho et al., 2020), **CrossLoc3D** (Guan et al., 2023) introduces a cross-source 3D matching framework that progressively shifts features using multi-scale sparse convolutions to construct a shared embedding space between modalities. This work also introduced the CS-Campus3D dataset, specifically designed to evaluate cross-source PR. While CrossLoc3D achieves performance comparable to prior approaches (Uy and Lee, 2018; Zhang and Xiao, 2019; Liu et al., 2019b; Xu et al., 2021; Komorowski, 2022) on the Oxford RobotCar dataset (Maddern et al., 2017), it outperforms them in a cross-source setting. However, the retrieval success threshold is set at 100 m, which may be too coarse for our task.

The use of global descriptors in point cloud retrieval offers several advantages. By representing each scene with a single descriptor, it becomes more memory-efficient and enables faster retrieval compared to local descriptors, as they remove the need for explicit keypoint detection and matching. Global descriptors are also less sensitive to local noise and viewpoint variations, making them particularly well-suited for large-scale applications. Since data-driven methods have consistently outperformed handcrafted approaches, we adopt LoGG3D-Net (Vidanapathirana et al., 2022) as our scene representation method. This choice allows us to evaluate the feasibility of our proposed approach in a controlled setting, where both the database and query point clouds are acquired from the same viewpoint, here on the KITTI dataset (Geiger et al., 2013). Table 2.1 resumes the state of the art of point cloud representation methods discussed in this section.

Table 2.1: Summary of point cloud representation methods of the state of the art.

Scope	Type	Method	Input Format	Representation Principle	Application	Retrieval Method	Contributions
Local	Handcrafted	3D-SIFT (Godil and Wagan, 2011)	Mesh	2D point normal histogram	Surface matching	Histogram similarity	Extension of SIFT to 3D surfaces
Local	Handcrafted	Spin-Images (Johnson and Hebert, 2002)	Mesh	2D point normal histogram	Surface matching	Efficient closest point search structure (Nene and Nayar, 1996)	Application of 2D matching techniques to 3D surfaces
Local	Handcrafted	FPFH (Rusu et al., 2009)	Point cloud	2D point normal histogram	Point cloud registration	Exhaustive search	Work on point clouds
Local	Handcrafted	SHOT (Salti et al., 2014)	Mesh	Normal-based histogram enriched by point location	Surface matching	Histogram similarity	More robust to noise and density variation
Local	Handcrafted	3D Gestalt (Bosse and Zlot, 2013)	Mesh	Height-based	Place recognition	Voting mechanism and histogram similarity	Descriptor for 3D LiDAR scans
Local	Learned	PointNet (Qi et al., 2017a)	Point cloud	Multi-linear perceptron	classification Segmentation	Feature embedding	First deep model for raw point clouds
Global	Handcrafted	ScanContext (Kim and Kim, 2018)	Point cloud	2D polar matrix projection	Place recognition	k-d tree	Rotation-robust global descriptor
Global	Learned	PointNetVLAD (Uy and Lee, 2018)	Point cloud	PointNet + NetVLAD	Place recognition	k-d tree	First end-to-end trainable model for place retrieval
Global	Learned	PCAN (Zhang and Xiao, 2019)	Point cloud	PointNet + weighted local features inside NetVLAD	Place recognition	k-d tree	Improve aggregation of relevant features
Global	Learned	LPD-Net (Liu et al., 2019b)	Point cloud	Graph-based feature aggregation	Place recognition	k-d tree	Capture both local and global geometry
Global	Learned	Locus (Vidanapathirana et al., 2021)	Point cloud	Segment feature + O2P	Place recognition	k-d tree	Multi-level features
Global	Learned	MinkLoc3D (Komorowski, 2021)	Point cloud	Sparse convolution-based	Place recognition	k-d tree	Efficient sparse CNN for global retrieval
Global	Learned	TransLoc3D (Xu et al., 2021)	Point cloud	Transformer-based	Place recognition	k-d tree	Introduce Transformer as an 3D encoder
Global	Learned	LoGG3D-Net (Vidanapathirana et al., 2022)	Point cloud	SparseConv U-Net + O2P	Place recognition	Exhaustive search	Add local consistency loss
Global	Learned	CrossLoc3D (Guan et al., 2023)	Point cloud	Transformer-based	Place recognition	k-d tree	Cross-source place recognition

2.3.3 Focus on LoGG3D-Net

We now describe in more detail the training and evaluation protocol of LoGG3D-Net, as it serves both as a baseline and as a 3D-encoder for our approach. This choice was motivated by the state-of-the-art performance it achieved on KITTI. This may be the combination of its local consistency loss, its local features obtained from efficient sparse convolution, and its contrastive learning.

Training protocol

During training, LoGG3D-Net relies on a contrastive learning strategy in which each query is represented by a tuple of point clouds [query, positive, negative, other], where positive is a scene close to the query, within 3 meters, and negative is a point cloud representing a scene located at least 20 meters away from the query (see Figure 2.5). The *other* is an additional negative that is distant from both the query and the first negative, serving as another contrastive example. This tuple is processed by the SparseConv U-Net (Komorowski, 2021), which computes local features by combining multi-scale convolutional voxel-based and point-based layers.

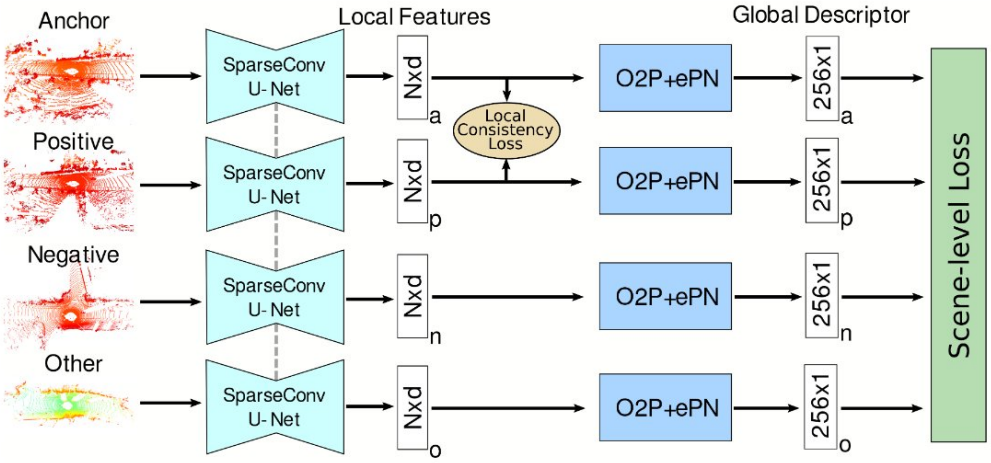


Figure 2.5: Overview of the LoGG3D-Net training method. The tuple [Anchor, Positive, Negative, Other] refers respectively to: the query point cloud (Anchor); a point cloud located within 3 meters of the Anchor (Positive); a point cloud located further than 20 meters from the Anchor (Negative); and a point cloud considered negative with respect to both the Anchor and the first Negative (Other). Local features are computed using a SparseConv U-Net, and a local consistency loss is applied to the features of the Anchor and Positive point cloud to increase their similarity. O2P and ePN refer to the second-order pooling and Eigen-value Power Normalization, respectively, which are used to aggregate local features into a global descriptor. The four global descriptors are then used to compute a scene-level loss that increases the dissimilarity with negative examples. The final loss is a combination of the local and global losses. Figure taken from LoGG3D-Net (Vidanapathirana et al., 2022).

Loss Functions

As introduced earlier, the local consistency loss enforces similarity between the local features of positive point cloud pairs while pushing apart negative pairs. This is implemented using the Hardest-Contrastive loss (Choy et al., 2019b), where the negative samples are selected to be the most similar and thus hard to differentiate from positive samples, defined as,

$$
\begin{aligned}
L_{lc} = & \sum_{(i,j)\in C^{1\leftrightarrow 2}} [\, \|f(P_i^1) - f(P_j^2)\|_2^2 - m_p \,]_+ / |C^{1\leftrightarrow 2}| \\
& + \lambda_n I_i [m_n - \min_{k\in M} \|f(P_i^1) - f(P_k^2)\|_2^2]_+ / |C_1^{1\leftrightarrow 2}| \\
& + \lambda_n I_j [m_n - \min_{k\in M} \|f(P_j^2) - f(P_k^1)\|_2^2]_+ / |C_2^{1\leftrightarrow 2}|
\end{aligned}
\tag{2.1}
$$

where $\{f(P_i^1), f(P_j^2)\}$ denotes the local features extracted from points i and j of the two point clouds. $C^{1\leftrightarrow 2}$ is the set of corresponding point indices between the two point clouds. M represents a randomly selected subset of features used for efficient training, and $[.]_+$ is the ReLU function. I_i are indicator functions that select valid correspondences. The hyperparameters m_p and m_n are positive and negative margins, respectively, and λ_n is a scalar weight used for negative losses, set at 0.5.

The set of local features extracted by the network is aggregated into a global descriptor using second-order pooling, and from its singular value decomposition, an Eigen-value Power Normalization step (Koniusz et al., 2016) is performed to obtain a normalized global descriptor. For local features of dimension d=16, this aggregation will produce a descriptor of size d^2=256. The scene-level loss is formulated as a quadruplet loss (Chen et al., 2017), $L_{quad} \in [0, +\infty[$, applied to the tuple $(P_q, P_p, P_n, P_{nbis})$. The objective of this loss is to reinforce the contrast between global descriptor similarities, ensuring that descriptors of positive pairs (P_q, P_p) are pulled closer together, while those of negative pairs (P_{nbis}, P_n) are pushed further apart. It is defined as :

$$
\begin{aligned}
L_{quad} = & \sum_{i}^{N} ([\|g(P_q) - g(P_p)\|_2^2 - \|g(P_q) - g(P_n i)\|_2^2 + \alpha]_+ \\
& + [\|g(P_q) - g(P_p)\|_2^2 - \|g(P_{nbis}) - g(P_n i)\|_2^2 + \beta]_+
\end{aligned}
\tag{2.2}
$$

where P_q is the query point cloud, P_p is a positive one within p = 3 m of the query, P_n is a negative point cloud distant from n = 20 m of the query, P_{nbis} respects the negative condition with both P_p and P_n. g(P_*) is the global descriptor vector. N is the number of sampled negatives, while α and β are the loss margins used to enforce separation between positive and negative pairs in the quadruplet loss, set at 0.5 and 0.3, respectively.

The global loss used is for the training of LoGG3D-Net is a combination of the local and global loss, such as :

$$
L = L_{quad} + \omega L_{lc} \tag{2.3}
$$

With ω a hyperparameter scalar that controls the contribution of L_{lc} relative to L_{quad}, set to 1 in the original paper.

Evaluation metric

The training procedure follows a leave-one-out cross-validation strategy, where the model is trained on ten sequences out of eleven. Only the six models that leave out sequences containing revisits are used for evaluation. In their article, LoGG3D-Net is assessed in terms of its loop closure detection capability with $F1_{max}$ metrics. The $F1_{max}$ score is defined as the highest value of the $F1$ measure, the harmonic mean between precision and recall, obtained by varying the decision threshold τ. It is defined as

$$
F1_{score} = 2\frac{precision.recall}{precision + recall} = \frac{2TP}{2TP + FP + FN} \tag{2.4}
$$

with TP, FP, TN, and FN being the classification of true positive, false positive, true negative, and false negative respectively.

For a sequence, the corresponding checkpoint is loaded, and the model processes the point clouds sequentially. For each scene, a global descriptor is computed and stored in a dynamically updated reference database. At inference time, the similarity between the current global descriptor and those in the database is measured using the cosine distance. To prevent trivial matches with immediately preceding scenes, a temporal buffer is applied where all scenes acquired within t seconds of the current time are excluded from retrieval, here t=30 seconds. Figure 2.6 presents how a query-database pairs descriptors are classified, according to their cosine distance and the Euclidean distance of their respective point cloud center. The threshold τ ranges from 0 to 1 with a step of 1/1000 in order to determine the best cosine distance threshold.

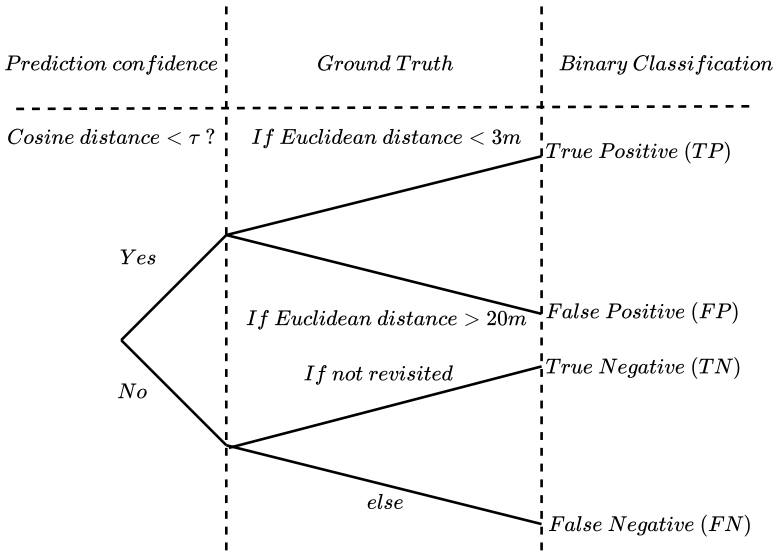


Figure 2.6: Binary classification of LoGG3D-Net descriptors. The prediction confidence corresponds to the distance in the representation space, computed using the cosine distance, between the query and the ground truth descriptor relative to a varying threshold τ. Depending on the outcome of the first classification, the binary classification then relies either on the geometric distance between the query and the ground truth or on the presence of a revisit.

Having reviewed descriptor learning approaches, we now describe the datasets used to evaluate them.

2.3.4 Place recognition datasets

After presenting the evolution of point cloud representations, we introduce the datasets most commonly used as benchmarks for place recognition in 3D point clouds, which mainly come from the robotics and mobile mapping domains. While many of these datasets include multiple sensing modalities such as image or video sequences, in this section, and throughout the remainder of the thesis, we refer to them exclusively by their LiDAR scan acquisitions.

Urban dataset from mobile mapping.

Among the various public LiDAR datasets available, urban driving environments are the most common for evaluating place recognition methods. The **KITTI** dataset (Geiger

et al., 2013), recorded in Karlsruhe (Germany) with a Velodyne 3D LiDAR scanner synchronized with GNSS/IMU, is one of the earliest and most widely used. The survey by (Shi et al., 2023) reports 34 methods evaluated on KITTI. It is divided into 21 sequences, with ground-truth poses available for the first 11, and six of them containing loop closures corresponding to revisits of previously seen locations. Its successor, KITTI-360 (Liao et al., 2022), extends the benchmark with improved GNSS localization and richer sensor modalities for full 360° scene perception. Other urban benchmarks include the **North Campus Long-Term** dataset (Carlevaris-Bianco et al., 2016) and the **Oxford RobotCar** dataset (Maddern et al., 2017), the latter of which captures the same 10 km route twice a week over a year, under diverse weather and lighting conditions. There is also **MulRAN** (Kim et al., 2020), which provides more varied and longer revisits scenarios on diverse environments (Dajeon city, KAIST university, riverside). More recent large-scale urban datasets include **ALITA** (Yin et al., 2022), which spans both urban and campus environments and is designed for long-term and large-scale place recognition tasks.

Non-urban and hybrid dataset

Hybrid datasets such as **CS-Campus3D** (Guan et al., 2023) combine terrestrial and aerial LiDAR scans, specifically aiming at cross-source place recognition. In contrast to these urban-focused datasets, **Wild-Places** (Knights et al., 2022) concentrates on natural forest environments, highlighting the challenges of place recognition in vegetation-rich, unstructured scenes. These datasets serve as benchmarks for evaluating both handcrafted and learning-based 3D descriptors introduced in the following sections. In the context of this thesis, we primarily focus on urban datasets, as they are more representative of our target application domain.

2.4 Retrieval accelerating techniques

In the previous section, we have presented the principles of 3D place recognition, with a focus on LoGG3D-Net and on the main datasets encountered in the literature for place recognition. However, despite the effectiveness of learned global descriptors, their deployment at scale remains constrained by the limitations of exhaustive search and tree-based retrieval structures, which become inefficient when handling high-dimensional descriptors over large databases. To overcome these challenges, this section introduces methods for retrieval acceleration, focusing on approaches that balance efficiency with retrieval quality.

Despite the richness of the database literature in terms of indexing strategies (Pan et al., 2024), the methods described in the previous section often rely either on an exhaustive search to find the k nearest neighbors (k-NN) of a query point cloud (Vidanapathirana et al., 2022), or on tree-based structures such as k-d trees (Kim and Kim, 2018; Uy and Lee, 2018; Zhang and Xiao, 2019; Komorowski, 2021; Xu et al., 2021; Bentley, 1975; Guan et al., 2023).

While brute-force search remains efficient for relatively small datasets, such as a single KITTI sequence (Geiger et al., 2013), its computational cost scales linearly with the size of the reference database (Pan et al., 2024). Conversely, k-d trees mitigate the need for full linear search by hierarchically partitioning the space, but their performance deteriorates rapidly in high-dimensional spaces due to the curse of dimensionality (Kibriya and Frank, 2007). For large-scale place recognition scenarios, such as city or country-wide maps containing millions of point clouds, exact search quickly becomes prohibitive, leading to long inference times and limiting the applicability of otherwise accurate descriptors.

To accelerate the retrieval of descriptor sets, global representation methods, such as the Bag-of-Features commonly used in information retrieval, are often used (O'hara and

Draper, 2011; Jégou et al., 2010). Another important class of acceleration techniques is Approximate Nearest Neighbor (ANN) (Indyk and Motwani, 1998) indexing structures, which propose a trade-off between exactness and efficiency. Instead of exhaustively computing all pairwise distances, ANN techniques accelerate retrieval by exploiting spatial or structural properties of the data.

The first acceleration technique, introduced in Section 2.4.1, is the very popular Bag-of-Features, a concept of vector aggregation by feature quantification into a histogram (or global representation), to build inverted index, which has inspired many subsequent aggregation methods (Arandjelovic and Zisserman, 2013; Arandjelovic et al., 2016). We also present tree-based retrieval, such as k-d tree, which is the most used structure in point cloud retrieval, in Section 2.4.2. The main ANN indexing structures, which tend to be the most efficient, can be broadly grouped into hashing-based and graph-based approaches (Pan et al., 2024), and are presented in Sections 2.4.3 and 2.4.4, respectively. Finally, Section 2.4.5 compares these structures in terms of time complexity and in terms of level of approximation.

2.4.1 Inverted index methods

When several multidimensional features are associated with the query (e.g. images with local image descriptors), a common strategy to accelerate retrieval of a set of features is the Bag-of-words representation (Qader et al., 2019). While Section 2.3.2 introduced the Bag-of-Words model as a vector aggregation method, this section focuses on its use as an inverted indexing structure for efficient retrieval.

This representation is based on the quantification of occurrence of words or local features (for example, based on visual contents), without regard for their relative position within the query. Its dimension is set by a predefined vocabulary of words that will be searched in the data. Local features are grouped into vocabulary centroids according to their similarity. The resulting histogram of features frequencies serves as a global descriptor. This histogram can then be used as an inverted index, which maps the local features of the query directly to its corresponding centroid and thus to relevant elements of the database. This strategy accelerates the retrieval by avoiding descriptor sets matching and by inverted indexes, for efficient information retrieval. The quantification of word occurrence can be done with the TF-IDF (Salton and Buckley, 1988), presented in Section 2.5.1. The TF-IDF is a score obtained by the product of a term (or local feature) frequency within a document and its inverse frequency of usage within the corpus, representing in how many documents the feature can be found (see Section 2.5.1). For example, the repetition of a feature increase its score, however, features that are very common across the database are downweighted since they are less salient than rarer features found in fewer documents.

The BoW was later adopted to computer vision as a Bag-of-Features model for 2D images (Gálvez-López and Tardos, 2012; Qader et al., 2019), typically after extracting SIFT or SURF descriptors and gathering them into centroids called *visual words*. The same idea has also been extended to 3D local features with (Steder et al., 2011; Cui et al., 2022). Similar to BoW, several aggregation techniques have been proposed to build global descriptors, including VLAD (Arandjelovic and Zisserman, 2013), its trainable variant NetVLAD (Arandjelovic et al., 2016), and second-order pooling approaches (Vidanapathirana et al., 2021; Komorowski, 2021; Vidanapathirana et al., 2022).

2.4.2 Tree-based methods

The k-d tree (Bentley, 1975) is a classical data structure for efficient points retrieval in a k-dimensional space. At each node, the dataset is recursively split by an axis-aligned hyperplane, producing a binary tree that partitions the space into smaller regions, up to a leaf size, determined by its number of elements. This reduces the number of regions and explicit distance computations required during search. Many point cloud retrieval methods, influenced by PointNetVLAD, adopted this search structure (Zhang and Xiao, 2019; Komorowski, 2021; Xu et al., 2021; Bentley, 1975; Guan et al., 2023). While k-d trees provide logarithmic search complexity in low dimensions, their efficiency degrades rapidly as dimensionality increases.

In high-dimensional spaces, the partitions fail to separate points effectively, most nodes end up being visited, and search complexity approaches linear (Indyk, 2004). This phenomenon, known as the curse of dimensionality, makes k-d trees unsuitable for the high-dimensional global descriptors, typically used in point clouds retrieval. Variants such as ball trees offer partial improvements but remain limited when scaling to large, high-dimensional datasets (Kibriya and Frank, 2007).

2.4.3 Hashing-based methods

The challenge of global descriptor-based 3D retrieval can be formulated as a large-scale vector database task. While most existing 3D retrieval rely on tree-based indexing structures, they suffer from the curse of dimensionality. In this state of the art, we also consider additional indexing structure families that have not been explored in point cloud retrieval.

In particular, among table-based indexing structures, a popular approach is Locality-Sensitive Hashing (LSH) (Indyk and Motwani, 1998; Douze et al., 2024), which maps points into hash *buckets* using similarity-preserving hash functions. Concretely, each point is projected onto k random hyperplanes, and the signs of these projections are encoded into an k-bit binary code. This binary representation allows nearest-neighbor queries to be approximated by hamming distance computations on binary vectors. However, achieving high recall typically requires selecting a sufficiently large number of discriminative hyperplanes, which in turn increases memory consumption and returns longer binary indexes.

Similarly, Product Quantization (PQ) (Gray, 1984; Jegou et al., 2010) compresses high-dimensional vectors into compact representations to accelerate similarity search. A vector of dimension d is divided into m sub-vectors of dimension d/m, and each sub-vector is assigned the index of its nearest centroid. The resulting compressed representation of size m is then used for similarity comparison, analogous to LSH binary vectors. The number of centroids c and the subdivision factor m control the trade-off between information retention and memory usage, with a time complexity of $O(m \cdot dk^{1/m})$. Several extensions of PQ exist, such as the inverted file system (Sivic and Zisserman, 2003) combined with asymmetric distance computation (Jegou et al., 2010), which reduces the training cost of the quantizer by introducing a coarse selection of centroids.

2.4.4 Graph-based methods

Although graph-based structures are not commonly used in 3D retrieval, we include them as complementary ANN baselines due to their strong performance in large-scale vector search (Pan et al., 2024).

NN-descent (Dong et al., 2011) demonstrated that graph-based ANN methods are both viable and scalable, achieving superior retrieval performance compared to the locality-sensitive hashing approaches previously presented. More recently, navigable small-world

(NSW) graphs (Malkov et al., 2014) and their hierarchical extension HNSW (Malkov and Yashunin, 2018) have emerged as state-of-the-art ANN techniques. These approaches construct a proximity graph in which nodes represent database items and edges connect neighboring nodes. At query time, retrieval is performed via greedy search, looking for short term best solutions. Starting from an entry point, the algorithm iteratively moves to the neighbor having the best score until no closer node can be found. In (Malkov and Yashunin, 2018), the graph is organized into multiple layers of decreasing density, which enables logarithmic search complexity $O(log(n))$ while outperforming NSW (Malkov et al., 2014). These methods offer excellent trade-offs between efficiency, scalability, and accuracy, and are now widely used in large-scale retrieval systems such as FAISS (Douze et al., 2024).

2.4.5 About search time complexity and approximations

This section compares the previous acceleration methods search time complexity and how they quantify the approximation introduced for the ANN ones.

Search time complexity. In the context of 3D place recognition, exact k-NN search not only scales poorly but also hinders real-time applications such as autonomous driving or large-scale localization. ANN methods provide a practical alternative, though their effectiveness depends on the search strategy as tree-based approaches degrade in high dimensions, hashing favors speed at the cost of recall, while graph-based methods achieve the best balance for dense, high-dimensional embeddings. The search complexities of these methods are summarized in Table 2.2.

Table 2.2: Average query complexity of exhaustive and approximate nearest neighbor search methods. The size of the database and the dimension of the elements are noted n and D, respectively, and ρ is LSH query-time exponent $0 \leq \rho \leq 1$.

Retrieval method	**Query complexity**
Exhaustive k-NN	$O(n)$
k-d tree (Lee and Wong, 1977)	$O(n^{1-\frac{1}{D}})$
LSH (Pan et al., 2024)	$O(n^{\rho})$ with $0 \leq \rho \leq 1$
HNSW (Malkov and Yashunin, 2018)	$O(log(n))$

This table illustrates the improvements in complexity efficiency achieved by indexes methods. Our work compares these ANN methods both in terms of retrieval quality and search-time complexity. At the same time, it is designed with the progress made in point cloud retrieval in mind, ensuring that it not only achieves competitive efficiency but also maintains the high retrieval accuracy required for practical deployment. In other words, the proposed framework seeks to bridge the gap between state-of-the-art acceleration techniques and the domain-specific constraints of 3D place recognition.

Approximations quantification. The approximate search methods, namely LSH and HNSW, introduce simplifications that may reduce performance compared to exact search.

The error introduced by LSH indexing can be quantified through collision probabilities (Indyk and Motwani, 1998; Gionis et al., 1999). For two points p and q, and a hashing function H, the probability of error (or undesired *collision*) corresponds to the case where the two points share the same hash index $H(p) = H(q)$. Let P_1 and P_2 denote the probabilities that two points collide when they are within a distance R_{near} or further

than R_{far}, respectively, with $P_1 > P_2$:

$$P_1 = P[\ H(p) = H(q) \mid \|p - q\| \leq R_{near}] \tag{2.5}$$

$$P_2 = P[\ H(p) = H(q) \mid \|p - q\| \geq R_{far}] \tag{2.6}$$

As mentioned earlier, one hash table uses k random projections to build an index, but in the general case, L independent hash tables are used. The probability of retrieving the correct neighbor $P_{success}$ then becomes (Andoni and Indyk, 2008):

$$P_{success} = 1 - (1 - p_1^k)^L \tag{2.7}$$

So the degrees of approximation can be controlled through the parameters L and k.

Unlike LSH, the approximation introduced by HNSW can only be estimated empirically (Malkov and Yashunin, 2018). The two parameters that influence search performance are m_L, the number of layers in the graph, and M_{max0}, the maximum number of neighbors per node in layer 0 (the deepest and the densest layer). Setting $m_L = 0$ reduces the HNSW into a K-NN or NSW graph, while increasing m_L also increases the overlaps of neighbors across different layers, which diminishes the benefit of having deeper layers. Consequently, an optimal choice for m_L exists, and is empirically set to $m_L = 1/ln(M)$. Experiments show that setting $M_{max0} = 2.M$, with M being the number of neighbors per node in upper layers, provides the best trade-off between performance and memory usage.

2.5 Information retrieval

As discussed in Section 2.3, many approaches of point cloud retrieval have drawn inspiration from techniques originally developed for 2D understanding. Following the same principle of transferring well-established solutions across modalities, we now turn to the field of information retrieval.

In its classical definition (Manning, 2008), information retrieval refers to the task of identifying documents (e.g., web pages, paragraphs, or metadata) within a large database that are relevant to a search query, which may itself take the form of text, images, or other content. Perhaps the most common example of information retrieval is web search, as ranking algorithms such as PageRank (Brin and Page, 1998), which orders web pages based on their hyperlink structure, laid the foundation for modern search engines such as Google.

In the remainder of this section, we review three major families of information retrieval approaches, term frequency–based methods in Section 2.5.1, dual encoder architectures in Section 2.5.2, and differentiable search index in Section 2.5.3 and discuss their relevance for retrieval in 3D point clouds. These approaches provide methodologies and conceptual frameworks that can inspire the design of scalable and effective retrieval strategies in the 3D domain.

2.5.1 Text-based Bag-of-Words retrieval

One of the earliest approaches to information retrieval in text documents was the Bag-of-Words model based on term weighting. As introduced in Section 2.4.1, documents are represented as sparse vectors according to a predefined vocabulary, without taking into account word order. This inverted-file representation accelerates the retrieval of similar documents. However, not all words in the natural language have the same level of descriptiveness, for example the word "the" is very common, whereas "trombone" is more topic-specific. Several weighting strategies therefore aim to assign higher importance to words that are more relevant for distinguishing documents.

A classical formulation is the Term Frequency–Inverse Document Frequency (TF-IDF) (Salton and Buckley, 1988). It is a bag-of-words approach which evaluates the importance of a term within a document relative to the entire corpus. It is defined as the product of the term frequency (TF), which measures how often a word occurs within a document, and the inverse document frequency (IDF), which downweights terms that are frequent across many documents while emphasizing rare, discriminative ones.

The term frequency of a term t in a document d is given by:

$$TF(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}} \tag{2.8}$$

where $f_{t,d}$ is the raw count of term t in document d, and the denominator is the total number of terms in d.

The inverse document frequency is defined as:

$$IDF(t,N) = \log\left(\frac{N}{n_t + 1}\right) \tag{2.9}$$

where N is the corpus size and n_t is the number of documents containing the term t.

The final TF-IDF weight of term t in document d is:

$$TF\text{-}IDF(t,d,N) = TF(t,d) \cdot IDF(t,N) \tag{2.10}$$

However, a limitation of TF-IDF is that it assumes a linear relationship between term frequency and importance (see Equation 2.8), which tends to overweight very frequent terms within long documents.

To address the limitations of TF-IDF, the Okapi BM25 model (Robertson and Jones, 1976; Trotman et al., 2014) was introduced as a probabilistic extension of TF-IDF. BM25 introduces two free parameters, k and b, to mitigate the linearity of the original TF-IDF formulation. The parameter k controls term frequency saturation, making the TF component non-linear. This ensures that repeated occurrences of a term do not indefinitely increase its weight. The parameter b adjusts the importance of the length of the document, relative to the average length of the corpus $avgdl$, as:

$$BM25(t,d,N) = IDF(t,N) \cdot \frac{f_{t,d} \cdot (k+1)}{f_{t,d} + k \cdot (1 - b + b \cdot \frac{N}{avgdl})} \tag{2.11}$$

These modifications allow BM25 to more accurately model term importance, making it an effective baseline for modern information retrieval tasks.

2.5.2 Dual encoders

A limitation of word-based retrieval methods is that they find the best document according to the presence of the query words in the document. This approach is sensitive to issues of synonymy and polysemy, where different words may carry the same meaning, or a single word may have multiple meanings, often resulting in mismatches between query intent and retrieved content (Deerwester et al., 1990).

An answer was proposed with the introduction of latent semantic indexing, building a semantic space from a term-document matrix using singular-value decomposition. It was the first step toward a learning based approach such as dual encoders (Gillick et al., 2018). Dual encoders are based on siamese networks where two models share the same embedding space. During training, they learn to encode documents into effective representations through contrastive learning, which pulls matching query–document pairs closer together

while pushing apart non-matching pairs. At inference time, retrieval reduces to a nearest-neighbor search in the shared embedding space, where the documents most similar to the query are returned (see Figure 2.7).

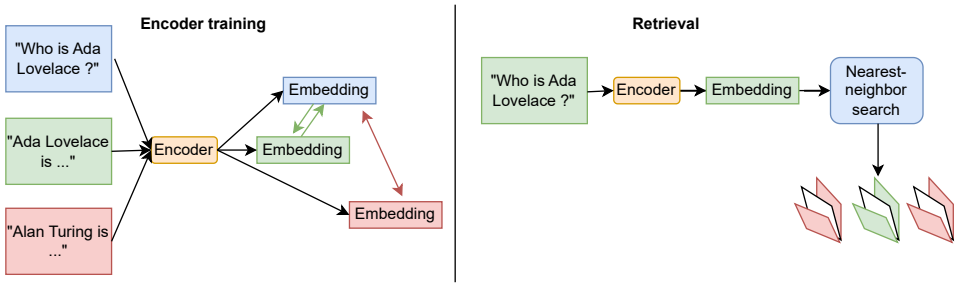


Figure 2.7: Overview of the Dual encoder training and retrieval method. (Left) The encoder is trained constrastively to maximize the representation similarity between a query (blue) and the relevant document (green), while also maximizing the dissimilarity with other documents (red). (right) The trained model first encodes the query into an embedding and then uses this representation to find the best documents, according to the query.

Building on the success of transformer architectures (Vaswani et al., 2017), dual encoder approaches now often rely on models such as Text-to-Text Transfer Transformer (T5) (Raffel et al., 2020) and its adaptation Sentence-T5 (Ni et al., 2021), having demonstrated the effectiveness of pre-trained transformers as dual encoders.

2.5.3 Differentiable Search Index

The core idea behind the Differentiable Search Index (DSI) (Tay et al., 2022) is to parameterize the traditional retrieve-then-rank pipeline, typically composed of two separate stages, indexing and retrieval, within a single unified T5 transformer model. This method has been recently categorized as generative retrieval (Zeng et al., 2024; Huynh et al., 2025). As illustrated in Figure 2.8, the DSI receives a query such as "Who is Ada Lovelace?" and directly outputs one or more document identifiers (docids), ranked by their relevance. Each docid corresponds to a specific document in the database, for example, the Wikipedia page of Ada Lovelace.

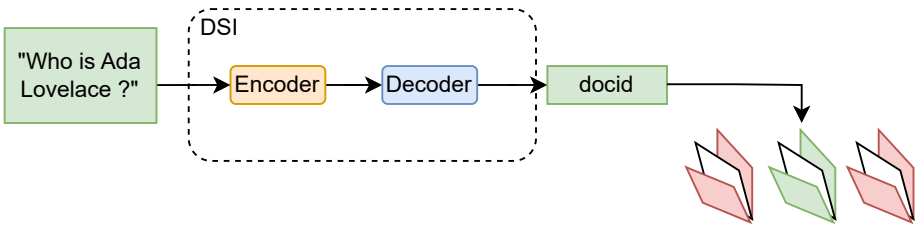


Figure 2.8: Overview of the Differentiable Search Index retrieval method. The DSI is trained to jointly encode and retrieve using a single transformer architecture that parameterizes corpus information and directly generates the docid relevant to the query.

Tokenization and generation algorithms

During indexing, DSI associates each document d_j with a unique string docid j. Textual information is represented as sequences of tokens, where each token corresponds to a

substring (e.g., a word or subword unit) that is mapped to a unique index in the vocabulary of the model. The three most common subword tokenization algorithms are Byte-Pair Encoding (BPE) (Gage, 1994; Sennrich et al., 2015), WordPiece (Schuster and Nakajima, 2012), and Unigram (Kudo, 2018). After corpus normalization (such as accent removal) and pre-tokenization (splitting sentences into lists of words), BPE (Gage, 1994; Sennrich et al., 2015) begins by associating each word with its frequency and decomposes it into characters that serve as the initial tokens. The algorithm then iteratively counts the frequency of adjacent token pairs across the corpus, merges the most frequent pair into a new token, and re-segments the corpus using this updated vocabulary. This process is repeated until the desired vocabulary size is reached, and the resulting tokenizer is defined by the learned sequence of merge rules. This strategy is widely adopted in models such as GPT (Radford et al., 2018), GPT-2 (Radford et al., 2019), and RoBERTa (Liu et al., 2019a). WordPiece tokenization, used in models like BERT (Devlin et al., 2019) and DistilBERT (Sanh et al., 2019), follows a similar strategy but selects candidate merges based on a score that balances pair frequency with overall token frequency. Finally, Unigram tokenization, employed in models such as ALBERT (Lan et al., 2019), and T5 (Raffel et al., 2020), takes the opposite approach: it starts from a large vocabulary and progressively prunes less informative tokens to reach the target vocabulary size.

During retrieval, the model generates docids in an autoregressive manner, i.e., tokens are produced sequentially, one after another, with each new token conditioned on the previously generated ones. The DSI architecture supports beam search, which enables the model to return multiple candidate identifiers ranked by their sequence scores. These scores, or logits $\in]-\infty, +\infty[$, correspond to the raw outputs of the language modeling head for each token in the vocabulary, prior to normalization. To obtain probabilities over the vocabulary, the logits are passed through a SoftMax function $\in [0, 1]$, defined as:

$$Softmax(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}} \tag{2.12}$$

The probability of generating a docid i given a query q is defined in (Zhuang et al., 2022) as:

$$p(i|q, \theta) = \prod_{i=1} p(i_t|T5_\theta(q, i_0, i_1, ...i_{t-1})) \tag{2.13}$$

where i_t is the i-th token in the docid string.

As previously mentioned, generative retrieval approaches based on sequence-to-sequence (seq2seq) models aim to decode a document identifier token sequence directly from a given query sequence. Consequently, the design of efficient and structured docids is crucial, as it directly impacts both the representation of documents and the accuracy of their retrieval. The original DSI framework (Tay et al., 2022) introduces several docid representation strategies to enable effective sequence-to-sequence retrieval:

Unstructured atomic Identifiers

The most basic way to represent documents is through *unstructured atomic identifiers*. In this approach, each document is assigned a unique docid, which is treated as an additional token in the output vocabulary O. Concretely, the output layer is extended to jointly predict both standard tokens and document identifiers by applying a softmax over the concatenated weight matrices:

$$O = \text{Softmax}\Big([W_{\text{tokens}}; W_{\text{docs}}]^\top h_{\text{last}}\Big) \tag{2.14}$$

where $W_{\text{tokens}} \in \mathbb{R}^{d_{\text{model}} \times N_{\text{tokens}}}$, $W_{\text{docs}} \in \mathbb{R}^{d_{\text{model}} \times N_{\text{documents}}}$, and $h_{\text{last}} \in \mathbb{R}^{d_{\text{model}}}$ are the hidden states of the final decoder layer. Retrieval is then performed by ranking the logits corresponding to document identifiers and selecting the top-k.

Naively structured string identifiers

In this approach, each document identifier is represented as a tokenizable string (e.g., a sequence of digits corresponding to a random integer). Instead of assigning one atomic identifier per document in the output layer, the model learns to generate the docid sequentially, token by token.

This formulation avoids the need for a large softmax layer over all documents, as the vocabulary remains limited to the standard token set. However, retrieval becomes a sequence generation problem: the model must decode an entire string correctly to retrieve a document. For this, beam search is employed to approximate top-k retrieval by exploring multiple high-probability candidate strings without exhaustively scoring every possible docids.

Semantically structured string identifiers

Contrary to the previous approaches, which relied on arbitrary docids, *semantically structured string identifiers* aim to encode document content information directly into the identifiers themselves. In this case, identifiers are not random, instead, they are assigned in a way that reflects document semantics.

Those identifiers should both capture semantic similarity between documents and reduce the search space during decoding by constraining candidates at each successive prefix. With this approach, similar documents will share identifier prefixes. DSI used hierarchical clustering with a k-means algorithm over document embeddings (see Figure 2.9).

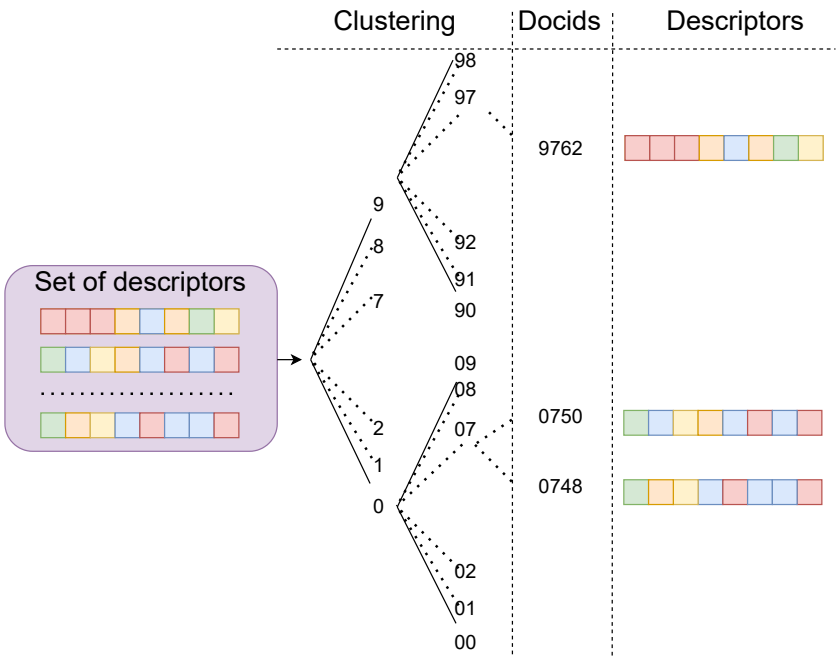


Figure 2.9: Example of semantically structured string identifiers. The set of descriptors is divided into multiple hierarchical clusters, until each cluster contains a single element. Each level consists of ten clusters, each assigned to a prefix digit. The prefixes along the hierarchical cluster are then concatenated to form the docids associated with each descriptor.

DSI evaluation on identifier structures

DSI was evaluated on the Natural Questions dataset (NQ) (Kwiatkowski et al., 2019), which contains 320k query–document training pairs. Each query–document instance is represented as a quadruple $[question, wikipedia\ page, long\ answer, short\ answer]$. For retrieval evaluation, the task is to identify the correct document (Wikipedia page) that provides the answer to each query question.

Results showed that semantically structured string identifiers consistently outperformed their naively structured counterparts. In contrast, atomic identifiers produced mixed results, while sometimes competitive, they exhibited high variance and failed to generalize reliably across datasets.

Query generation

Documents used to train DSI, such as the Wikipedia passages from the NQ dataset, are typically much longer than most user queries. As a result, DSI (Tay et al., 2022) indexes long documents during training but receives short queries at inference time, creating a query–document length imbalance. To address this mismatch, DSI-QG (Zhuang et al., 2022) introduces a query generation approach. Instead of directly training on full documents, each document is associated with a set of synthetic queries that are shorter and more representative of real user inputs. The model is then trained on these query–document pairs, effectively reducing the length disparity and improving generalization to unseen queries. Empirically, DSI-QG outperforms the original DSI on both monolingual datasets (e.g., NQ 320k (Kwiatkowski et al., 2019)) and multilingual datasets (e.g., XORQA 100k (Asai et al., 2020)), demonstrating the benefits of query generation for generative retrieval.

2.6 Conclusion

In this chapter, we have reviewed the evolution of point cloud representations designed for place recognition, from local handcrafted descriptor approaches to transformer architectures. We discussed their benefits in terms of scope, comparing local and global descriptions, and in terms of construction, differentiating between handcrafted and learning-based methods. A key limitation of current large-scale place recognition methods lies in their reliance on exhaustive descriptor matching, which is computationally expensive and often not optimized for high-dimensional embeddings (k-d tree). This chapter also revisits state-of-the-art acceleration techniques from vector database management systems, which reduce retrieval costs relative to exhaustive search while trading off a portion of retrieval quality. Lastly, this chapter also reviews the evolution of methods in information retrieval, currently used to retrieve documents within a very large corpora, from word-based frequency to generative retrieval.

As efficient retrieval at large scales is mostly underexplored, in the following chapter, we propose a generative retrieval framework inspired by information retrieval techniques such as DSI, which have shown promising results in the text domain.

Chapter 3

DSI-3D, an adaptation of the Differentiable Search Index to 3D data

Contents of this Chapter

3.1 Introduction 47
3.2 Vision transformers 48
3.2.1 Motivations 48
3.2.2 Generative image-to-text transformers 48
3.3 DSI-3D 50
3.3.1 Point cloud encoder 50
3.3.2 Point cloud indexing 51
3.4 Conclusion 56

3.1 Introduction

The Chapter 2 presents how point cloud representation methods evolved, along with common acceleration techniques and their limitations, and examines how information retrieval systems achieve retrieval in large textual databases.

From the conclusions of the previous chapter, we adopt learned global features for point cloud embeddings, as they are significantly less computationally expensive and scale better than local features. We chose Learned-based features, as they tend to have better performance on the place recognition task.

A second conclusion is that the DSI (Tay et al., 2022) provides an efficient solution that can be adapted to retrieve point clouds by directly generating document identifiers using a transformer-based pipeline for the retrieve-then-rank process. This chapter presents our first contribution: applying the Differentiable Search Index (DSI) framework to 3D data.

First, we bridge the design gap between 3D point cloud data and existing DSI-based models through vision transformers in Section 3.2, followed by Section 3.3, where we present how we implement DSI-3D and the various indexing structures adapted for place recognition tasks.

3.2 Vision transformers

The previous chapter also provides an overview of information retrieval approaches, focusing on generative retrieval. However, the original DSI framework (Tay et al., 2022) does not support the direct input of 3D point clouds, as it was designed for text-based retrieval tasks. To address this limitation, we adapt the DSI training and retrieval mechanisms to operate on modified Vision Transformers (ViTs), enabling the generation of docids in the form of scene captions. Section 3.2.1 explains our motivation for leveraging ViT in this context, while Section 3.2.2 details the design choices behind the adaptation, including the selection of an appropriate ViT backbone and the training strategy employed.

3.2.1 Motivations

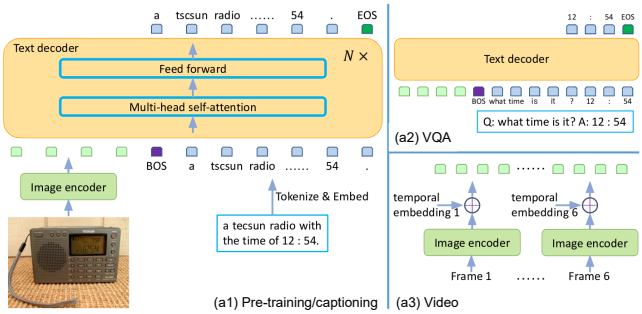
The DSI architecture is originally implemented on a standard framework for fine-tuning a seq2seq transformer (e.g., T5) to map queries directly to docids. However, this implementation does not support direct point cloud input.

To address the challenge of bridging the modality gap between 3D data and text, we draw inspiration from existing solutions for associating images with text, which leads us to Vision Transformers (ViTs). Transformer models (Vaswani et al., 2017) have demonstrated remarkable performance in maintaining long-range dependencies in sequences, making them well-suited for multimodal tasks where the full context is important. Consequently, the vision community has adapted transformers to a variety of problems, including image recognition (Dosovitskiy, 2020), object detection (Carion et al., 2020), and visual question answering (Tan and Bansal, 2019; Wang et al., 2022; Li et al., 2023). Adapting the multimodal aspect of ViTs to 3D-to-text by changing the image encoder to a 3D encoder seems a promising way to learn context-aware docids.

BLIP-2 (Li et al., 2023) was considered a potential ViT backbone. It achieves state-of-the-art performance on vision-language tasks such as visual question answering, image captioning, and image-text retrieval. This success can be attributed to its use of a triple-objective loss during the representation learning stage. While these losses have proven effective for vision-language tasks, they are not trivial to adapt to 3D data in a balanced and efficient manner. Additionally, the Q-Former module is designed to be followed by a frozen large language model (LLM), introducing further complexity for DSI applications due to its need for autoregressive output. For these reasons, we did not choose to use BLIP-2 as our backbone, and we have turned to GIT, presented below.

3.2.2 Generative image-to-text transformers

For the ViT backbone, we adopt the Generative Image-to-Text (GIT) transformer (Wang et al., 2022) due to its modular architecture and strong performance in image captioning and visual question answering tasks. GIT is a transformer-based framework that combines a frozen image encoder based on the Florence foundation model (Yuan et al., 2021) with an autoregressive text decoder to generate captions from visual inputs, as illustrated in Figure 3.1.



(a) GIT architecture



(b) Example captions generated by GIT

Figure 3.1: Illustration of the GIT training and inference process with example captions. (a1)(Pre-training) shows the pre-training, where a descriptor is computed by a frozen image encoder. This descriptor is associated with a target caption that the text decoder learns to generate based on the encoded image. (a2)(VQA) shows the captioning process and (a3)(Video) shows the capacity of GIT to be extended to video, by adding a temporal embedding. (b) shows examples of images captioning with GIT. Figure taken from (Wang et al., 2022).

The image encoder (Yuan et al., 2021) is pre-trained with Unified Contrastive Learning (UniCL) (Yang et al., 2022), a contrastive image–text matching approach in which multiple image–text pairs associated with the same semantic label (a higher-level concept) are treated as positive examples. This differs from the Contrastive Language-Image Pre-training (CLIP) framework (Radford et al., 2021), in which only exact image–caption pairs are considered positive, while all others are treated as negatives. During training, the frozen encoder produces 2D image embeddings that are flattened and projected into a D-dimensional feature space (D=768). The target caption, corresponding to the image, is tokenized and encoded into a D-dimensional embedding. The concatenated image and text features then serve as input to the transformer-based text decoder. The decoder generates the caption autoregressively, producing one token at a time, starting with a Beginning-of-Sequence (BOS) token and continuing until an End-of-Sequence (EOS) token is produced or a predefined maximum sequence length is reached. An attention mask is applied so that each text token attends only to the image features and the previously generated text tokens. The resulting token sequence is then decoded into the final natural language caption.

For both GIT pre-training and fine-tuning, the language modeling loss used is $L_m \in [0, +\infty[$, with I the image, y_i for $i \in \{1, ..., N\}$ the text tokens, y_0 the [BOS] token, and y_{N+1} the [EOS] one:

$$L_m = \frac{1}{N+1} \sum_{i=1}^{N+1} CE\Big(y_i, P(y_i \mid I, y_0, \ldots, y_{i-1})\Big), \tag{3.1}$$

where CE is the cross-entropy loss:

$$CE(y, x) = -\sum_{c=1}^{C} y_c \; log \frac{\exp(x_c)}{\sum_{j=1}^{C} \exp(x_j)} \tag{3.2}$$

and x is the input logit, y the target, and C the vocabulary size.

In summary, GIT employs a frozen vision encoder and an autoregressive text decoder, trained with a language modeling loss over image–text pairs. While this framework provides a baseline for multimodal understanding, it is not directly adapted for 3D data. In the following section, we describe how we adapt and train GIT according to the DSI trainer to match docids to point clouds.

3.3 DSI-3D

This work aims to extend the scalability of text information retrieval to the domain of point cloud retrieval. In this section, we show how the adaptations of GIT described above lead to the construction of our model, DSI-3D. Section 3.3.1 introduces the overall pipeline of DSI-3D, where a vision transformer is adapted to handle 3D inputs (see Figure 3.2). As discussed in Section 2.5.3 from Chapter 2, selecting an appropriate strategy is crucial for optimizing document indexing. Section 3.3.2 presents how existing indexing solutions can be applied to point cloud retrieval, with place recognition as the final objective. Finally, we introduce two new indexing strategies specifically designed for georeferenced documents.

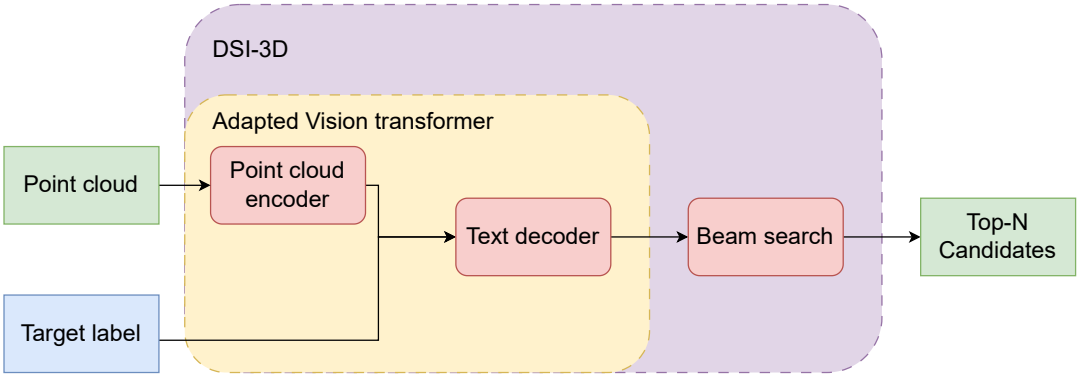


Figure 3.2: Proposed architecture of DSI-3D, using a point cloud encoder inside a vision transformer (in yellow) which serves as the core of a DSI model (in purple).

3.3.1 Point cloud encoder

To extend the GIT framework to 3D scene understanding, the image encoder is replaced with a point cloud encoder. In the implementation of this proposal and in the experiments, we adopt LoGG3D-Net as a point cloud descriptor (Section 2.3.3, Chapter 2), due to its robustness and strong retrieval performance on mono-source datasets (see Appendix A), but other descriptors could be considered. However, for the remainder of this chapter, DSI-3D models are trained on descriptors extracted from unperturbed point clouds, while inheriting the robustness properties identified in Appendix A. As discussed in Section 3.2.2, GIT training consists of two stages: image encoder pre-training and full model training. This ensures that only the text decoder is trained to generate the correct docid. In this configuration, the role of the image features is taken over by the point cloud descriptors, as illustrated in Figure 3.3. We retain the same tokenizer as in GIT, pre-trained on `git_base_coco`.

In our adaptation, the target sequences are docids rather than captions, allowing the training mechanism to remain unchanged while learning to map 3D features to identifiers.

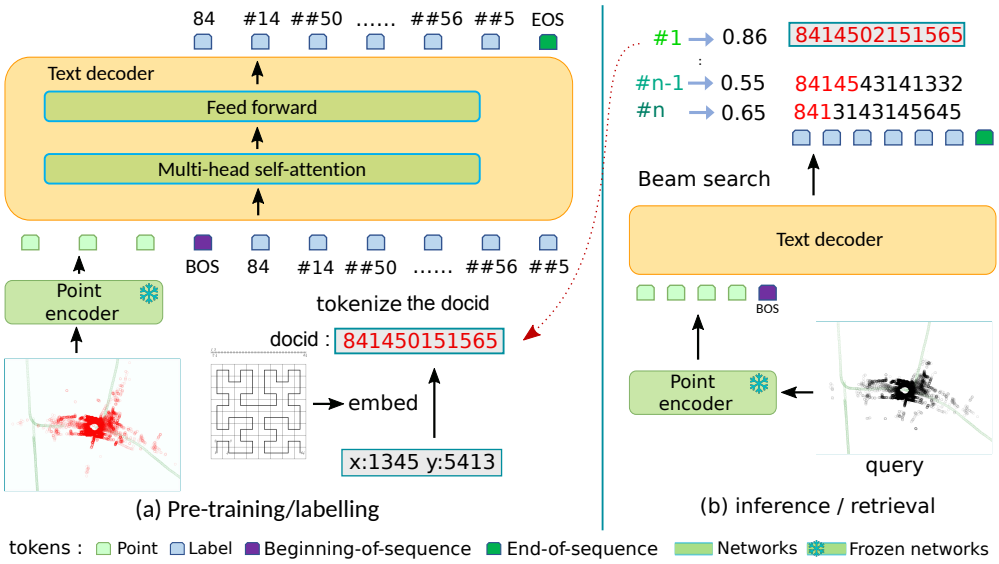


Figure 3.3: Indexing and retrieval step with the 3D Differentiable search index. (a) Indexing: For each point cloud (in red), a descriptor is calculated by a frozen point encoder (in green). This descriptor is associated with an index (docid) that the text decoder (in yellow) learns to generate based on the encoded point cloud. (b) Retrieval: For a query point cloud (in black), a beam search is performed by the text decoder to provide the n most probable solutions.

3.3.2 Point cloud indexing

This section describes how we adapt document indexing methods to georeferenced point clouds. The DSI (Tay et al., 2022) unifies indexing and retrieval by directly learning a mapping from queries to docids. In the context of place recognition, the analogous task is to map a scene query to its corresponding docid. To achieve this, the sequence-to-sequence (seq2seq) formulation must be extended into a scene-to-sequence paradigm. Unlike textual data, point clouds do not provide pre-existing tokens from which a docid can be constructed. Instead, a learned 3D representation must serve as the input to be associated with a docid. Following the DSI framework, we therefore enumerate the possible strategies for indexing point clouds and assess their feasibility. In this section, we present existing state-of-the-art strategies, such as unstructured atomic identifier, naively structured string identifier, and semantically structured string identifier, and then introduce our two proposed approaches relying on positional structured string identifier based on GNSS coordinates and Hilbert-curve distance.

Unstructured atomic identifier

The simplest indexing strategy from the state of the art is to assign docid represented by a unique token, to each document. The *Unstructured Atomic Identifiers* were not retained, as scaling up the number of documents directly increases the size of W_{docs} in Equation 2.14 from Chapter 2, resulting in a large output layer. This not only slows down the softmax computation but also leads to excessive memory usage.

Naively structured string identifier

Another standard docid construction is based on naively structured string identifiers. This approach represents each document using a tokenizable string that corresponds to a random integer. In our experiments, the identifier assigned to each point cloud was derived from the one defined in a sequential mapping dataset (as in the KITTI dataset

for instance), i.e. the file name with the file extension removed. For example, a point cloud file named 004541.bin is assigned the identifier *004541*. While being geometrically close, two distinct groups or representations, [*000001*, *000002*, *000003*] and [*004449*, *004450*, *004451*], do not share token elements between sequences. Although this indexing preserves a degree of temporal proximity between identifiers, since point clouds were acquired sequentially, it does not capture revisits, as seen in Table 3.1. Another example is provided in Figure 3.6a.

Table 3.1: Example of a naively structured string identifier for loop-closure revisits. This table lists scenes located close to a given point cloud *000000* in a mobile-mapping sequence. The docids are derived from the sequential indexes assigned in the KITTI dataset. The token sequence is the encoded representation of the docid by a list of pointed tokens. BOS and EOS are special tokens used to delimit the beginning and the end of the sequence. The # symbol indicates that the decoded sub-word is a suffix.

docid	token sequence	decoded sequence
'000000'	[BOS, 2199, 8889, 2692, EOS]	['000', '##00', '##0']
'000001'	[BOS, 2199, 8889, 2487, EOS]	['000', '##00', '##1']
'000002'	[BOS, 2199, 8889, 2475, EOS]	['000', '##00', '##2']
'000003'	[BOS, 2199, 8889, 2509, EOS]	['000', '##00', '##3']
'004449'	[BOS, 4002, 22932, 26224, EOS]	['00', '##44', '##49']
'004450'	[BOS, 4002, 22932, 12376, EOS]	['00', '##44', '##50']
'004451'	[BOS, 4002, 22932, 22203, EOS]	['00', '##44', '##51']

Semantically structured string identifier

A classical docids representation from the state of the art, used by DSI, are semantically structured string identifiers. They are an informed assignment of identifiers that satisfies two properties, capturing document semantics and improving the search space after each decoding step. As presented in the DSI paper, these semantically structured identifiers are constructed through hierarchical clustering of documents based on semantic similarity. For 3D point clouds, clustering is performed using k-means grouping of the point cloud embeddings. Since revisits of the same place tend to produce similar embeddings, they are more likely grouped into the same clusters (Figure 3.6b, Table 3.2). With this indexing, the same set of examples shares at least a prefix token with *000000*. In that case, it is also the maximum number of shareable tokens since *000000* is encoded by only two tokens.

Table 3.2: Example of a semantically structured string identifier for loop-closure revisits. This table lists scenes located close to a given point cloud *000000* in a mobile-mapping sequence. The docids are an aggregation of k-means cluster prefixes. The token sequence is the encoded representation of the docid by a list of pointed tokens. BOS and EOS are special tokens used to delimit the beginning and the end of the sequence. The # symbol indicates that the decoded sub-word is a suffix.

Naive index	docid	token sequence	decoded sequence
'000000'	8730	[BOS, 6584, 14142, EOS]	['87', '##30']
'000001'	8731	[BOS, 6584, 21486, EOS]	['87', '##31']
'000002'	8732	[BOS, 6584, 16703, EOS]	['87', '##32']
'000003'	9780	[BOS, 6584, 17914, EOS]	['87', '##80']
'004449'	8771	[BOS, 6584, 2581, 2487, EOS]	['87', '##7', '##1']
'004450'	8772	[BOS, 6584, 2581, 2475, EOS]	['87', '##7', '##2']
'004451'	8773	[BOS, 6584, 2581, 2509, EOS]	['87', '##7', '##3']

Positional structured string identifier

Following the two properties proposed by DSI for *Semantically structured string identifiers*, we propose two strategies tailored to georeferenced data in which docids are derived directly from spatial positions. Our first proposition is positional structured string identifiers based on GNSS coordinates. Each point cloud is defined by its center coordinates in a Cartesian system, denoted as $X = \{x_0, \ldots, x_{n-1}\}$ and $Y = \{y_0, \ldots, y_{n-1}\}$, with the Z-axis omitted as we focus on planar geolocation. To construct identifiers, the coordinates are first shifted to ensure non-negative values and are rescaled to centimeter-level precision, allowing us to distinguish even very close revisits. The identifiers are then obtained by interleaving the digit strings of each coordinate pair (x_n, y_n), starting from the most significant digit of x_n down to the centimeter precision of y_n. This process yields a unique and bijective mapping between point clouds and their identifiers. For example, a point cloud with center coordinates x_n = `"1111"` and y_n = `"2222"` is assigned the identifier `12121212`, as shown in Figure 3.4. This design ensures that revisits to the same location generate closely related identifiers, thereby preserving spatial proximity in the identifier space.

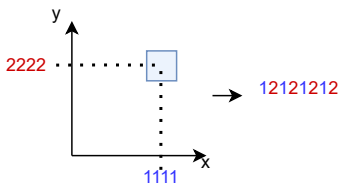


Figure 3.4: Visualization of the positional structured identifier using GNSS coordinates. The docid is constructed by interlacing the GNSS coordinates of the point-cloud center (blue square). In our case, the interlacing proceeds from the left to the right, beginning with the X-axis. Coordinates are shifted and rescaled to ensure they contained only positive integers.

This bijective mapping between point clouds and identifiers ensures that each revisit of a scene will have a closely related identifier (see Figure 3.6c and Table 3.3). The resulting

indexes are much longer than the two previous indexing methods, but revisits share a closer representation to *000000*.

Table 3.3: Example of a positional structured string identifier (GNSS) for loop-closure revisits. This table lists scenes located close to a given point cloud *000000* in a mobile-mapping sequence. The docids are constructed by interleaving the digit strings of the point cloud center coordinates. The token sequence is the encoded representation of the docid by a list of pointed tokens. BOS and EOS are special tokens used to delimit the beginning and the end of the sequence. The # symbol indicates that the decoded sub-word is a suffix.

Naive index	docid	token sequence	decoded sequence
'000000'	2071172680	[BOS, 19843, 14526, 2581, 23833, 17914, EOS]	['207', '##11', '##7', '##26', '##80']
'000001'	2071182436	[BOS, 19843, 14526, 2620, 18827, 21619, EOS]	['207', '##11', '##8', '##24', '##36']
'000002'	2071191392	[BOS, 19843, 14526, 2683, 17134, 2683, 2475, EOS]	['207', '##11', '##9', '##13', '##9', '##2']
'000003'	2072101148	[BOS, 19843, 17465, 24096, 16932, 2620, EOS]	['207', '##21', '##01', '##14', '##8']
'004449'	2071094072	[BOS, 19843, 10790, 2683, 12740, 2581, 2475, EOS]	['207', '##10', '##9', '##40', '##7', '##2']
'004450'	2071095673	[BOS, 19843, 10790, 2683, 26976, 2581, 2509, EOS]	['207', '##10', '##9', '##56', '##7', '##3']
'004451'	2072006245	[BOS, 19843, 28332, 2575, 18827, 2629, EOS]	['207', '##200', '##6', '##24', '##5']

Hilbert curve Positional Structured identifier

Our second proposition is positional structured string identifiers based on Hilbert-curve distances. In fact, a limitation of the previous interlacing approach is that one direction, X in our case, tends to dominate over translations along the Y axis. This bias arises because models typically learn lower-frequency components (the first tokens) earlier, while higher-frequency components require more training.

The problem of designing effective 2D indexes can thus be seen as a locality-preservation problem on how to map 2D spatial information into a 1D sequence while conserving neighborhood relationships. From this perspective, space-filling curves offer a natural solution. Among them, the Hilbert Curve is a continuous, fractal, space-filling curve that assigns each point in 2D space a position corresponding to the distance traveled along the curve to reach it. Figure 3.5 illustrates how the Hilbert curve maps 2D points into a 1D index at different iterations.

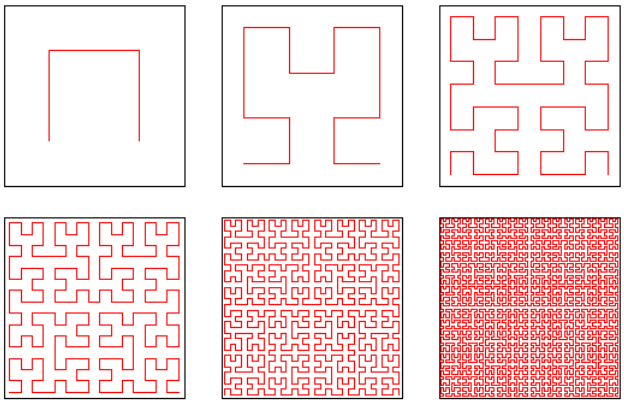


Figure 3.5: Visualization of the first six iterations of the Hilbert curve in a 2D space. Image taken from Wikipedia [1].

Compared to alternatives such as the Z-order or Peano curves, the Hilbert Curve provides superior locality preservation, meaning that distances between indexes more closely

[1] https://fr.wikipedia.org/wiki/Courbe_de_Hilbert

approximate the true geometric distances between points (Faloutsos and Roseman, 1989). We used the Hilbert curve implementation from the project "The Python Package Index"[2]. The Hilbert curve requires 2D points to be expressed as non-negative integers. To meet this requirement, we reuse the coordinates computed from the first *Positional Structured Identifier*, applying a positive shift and rescaling them to centimeter-level precision in order to distinguish very close revisits.

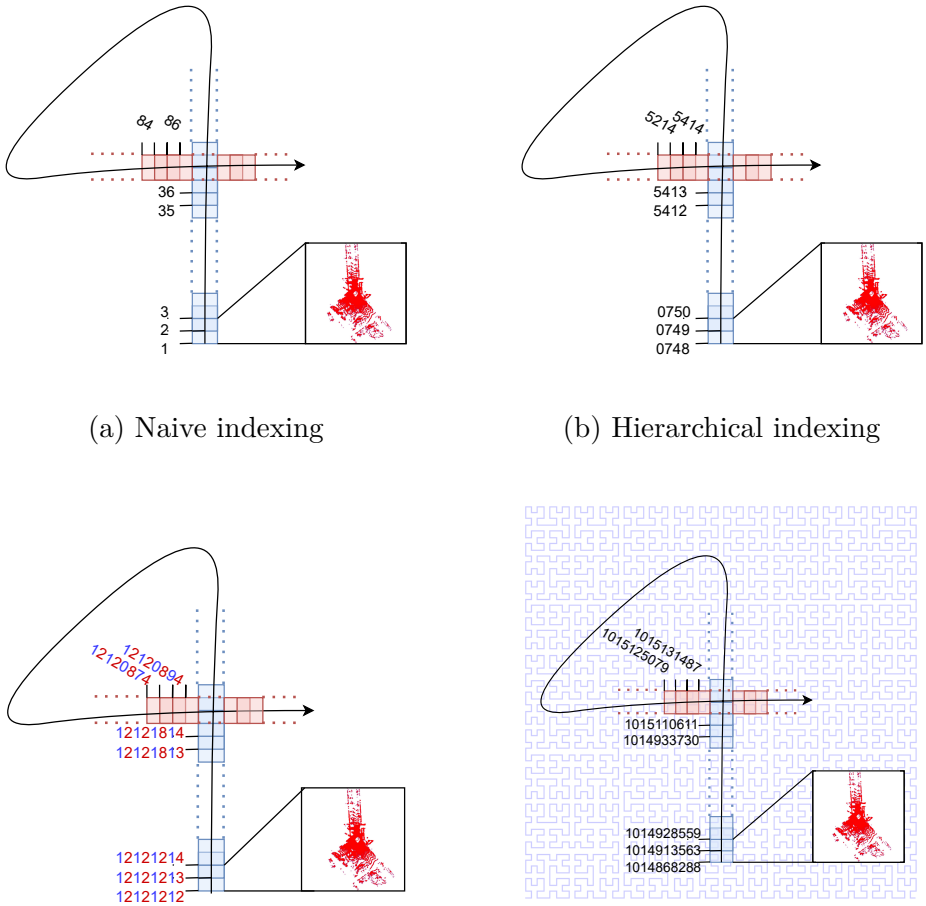
Table 3.4 presents identifiers produced using the Hilbert-curve–based positional scheme for frames near a given frame 000000. This example highlights how all revisits can share the same two-prefix structure, despite occasional index jumps (##48 and ##49) introduced by the Hilbert mapping.

Table 3.4: Example of a positional structured string identifier (Hilbert curve) for loop-closure revisits. This table lists scenes located close to a given point cloud *000000* in a mobile-mapping sequence. The docids correspond to the distance traveled along the Hilbert curve to reach each point cloud center. The token sequence is the encoded representation of the docid by a list of pointed tokens. BOS and EOS are special tokens used to delimit the beginning and the end of the sequence. The # symbol indicates that the decoded sub-word is a suffix.

Naive index	docid	token sequence	decoded sequence
'000000'	1014868288	[BOS, 7886, 18139, 2575, 2620, 22407, 2620, EOS]	['101', '##48', '##6', '##8', '##28', '##8']
'000001'	1014913563	[BOS, 7886, 26224, 17134, 26976, 2509, EOS]	['101', '##49', '##13', '##56', '##3']
'000002'	1014928559	[BOS, 7886, 26224, 22407, 24087, 2683, EOS]	['101', '##49', '##28', '##55', '##9']
'000003'	1014933730	[BOS, 7886, 26224, 22394, 2581, 14142, EOS]	['101', '##49', '##33', '##7', '##30']
'004449'	1014919747	[BOS, 7886, 26224, 16147, 2581, 22610, EOS]	['101', '##49', '##19', '##7', '##47']
'004450'	1014926180	[BOS, 7886, 26224, 23833, 15136, 2692, EOS]	['101', '##49', '##26', '##18', '##0']
'004451'	1014936449	[BOS, 7886, 26224, 21619, 22932, 2683, EOS]	['101', '##49', '##36', '##44', '##9']

As illustrated in Figure 3.6d, revisits of nearby locations are assigned similar indexes, preserving locality. However, the resulting identifiers tend to be longer compared to those generated by the other methods.

[2] https://pypi.org/project/hilbertcurve/



(a) Naive indexing

(b) Hierarchical indexing

(c) positional indexing

(d) positional indexing (Hilbert curve)

Figure 3.6: Visual example of the different indexing strategies for revisits. (a) Naive strategy; (b) Hierarchical strategy; (c) Positional (GNSS) strategy, (d) Positional (Hilbert curve) strategy. The black arrow indicates the trajectory, and each square represents a point cloud. The number next to the squares are examples of docids produced by each corresponding method.

3.4 Conclusion

In this chapter, we introduced DSI-3D, our first approach for indexing and retrieving 3D point clouds with a generative language framework. By leveraging a 3D point cloud descriptor (LoGG3D-Net) and the GIT Vision Transformer, we demonstrated how 3D spatial data can be transformed into discrete token sequences suitable for DSI-based retrieval.

We explored four point cloud indexing strategies. The first two follow the original DSI formulation, namely, naively structured and semantically structured string identifiers, while we introduce two additional positional structured identifiers, based respectively on interlaced GNSS coordinates and Hilbert curve distances.

The next chapter assesses the retrieval capability of DSI-3D on a commonly used mobile mapping dataset in 3D place recognition, KITTI. This means that the database and the queries will share the same acquisition methods (ground scans), with the possibility of expanding later to a multi-sourced dataset (aerial+ground scans). Our model is compared with approximate search techniques, which are widely used to accelerate retrieval in large data spaces. This determines whether the trade-off between acceleration and retrieval

capacities is comparable to that of other exhaustive and approximate search methods, and how they behave at a larger scale.

Chapter 4

Evaluation of DSI-3D for 3D point cloud retrieval

Contents of this Chapter

4.1 Introduction . . . 58
4.2 KITTI Dataset . . . 59
4.2.1 Overview of the KITTI dataset . . . 59
4.2.2 Building a new KITTI sequence . . . 60
4.2.3 Dataset splitting . . . 61
4.3 Experimental setup . . . 62
4.3.1 Contrastive learning . . . 62
4.3.2 Vocabulary restrictions . . . 63
4.3.3 Evaluation protocol . . . 63
4.4 Evaluation of DSI-3D . . . 64
4.4.1 Indexing strategies . . . 65
4.4.2 Positioning DSI-3D among retrieval approaches . . . 67
4.4.3 Complexity analysis . . . 68
4.5 Experimenting on DSI-3D hyperparameters . . . 70
4.5.1 Beam search width and vocabulary restrictions effects . . . 70
4.5.2 Optimizing training configuration . . . 72
4.6 Conclusion . . . 73
4.6.1 Overall assessment . . . 73
4.6.2 Chapter conclusions . . . 74

4.1 Introduction

In the previous chapter, we detailed our adaptations of the DSI architecture and indexing strategies for 3D place recognition. In this chapter, we turn to the evaluation procedure of DSI-3D on a mobile mapping dataset. In Section 4.2, we introduce the mobile mapping dataset used (KITTI) on which our models are trained and evaluated. An additional new KITTI sequence is built by aggregating the existing ones, and we present how the dataset is split for both place recognition and information retrieval objectives. In Section 4.3, we present the experimental setup used to train and evaluate DSI-3D, including a study of loss functions, vocabulary restrictions, and the metrics used to measure performance. Then, in Section 4.4, we evaluate our method, first in terms of indexation strategies for point cloud retrieval, and secondly in facing state-of-the-art retrieval approaches. We also analyze all of these in terms of retrieval complexity. Finally,we propose to study the

effects of the beam search width, vocabulary restrictions and training hyperparameters in Section 4.5.

4.2 KITTI Dataset

This section introduces the KITTI dataset, a widely used benchmark in autonomous driving and place recognition research (Shi et al., 2023). This choice also enables a direct comparison of our approach, using the pre-trained LoGG3D-Net point cloud representation, against state-of-the-art retrieval baselines. We start by describing the dataset's acquisition protocol, defining the notion of revisits, and outlining the sequences selected for our experiments in Section 4.2.1. Then, we present the construction of a new sequence in Section 4.2.2. Finally, Section 4.2.3 describes how this dataset is divided to prevent leakage between training and evaluation.

4.2.1 Overview of the KITTI dataset

The KITTI dataset (Geiger et al., 2013) is a mobile mapping benchmark recorded in Karlsruhe, Germany, using a Velodyne 3D LiDAR scanner synchronized with a GPS/IMU system. In our work, KITTI is employed to assess the effectiveness of our model since it is a commonly used dataset in place recognition. The dataset comprises 22 sequences (from 00 to 21), with ground-truth poses available for sequences 00 to 10. We focus on sequences 00, 02, 05, 06, 07, and 08, which contain revisited locations, as shown in Figure 4.1. A revisit is defined as a scene whose GNSS position lies within 3 meters of a previously visited location, provided that its acquisition time is at least 30 seconds apart. This temporal constraint ensures that consecutive frames or immediately adjacent captures are not mistakenly counted as revisits. Table 4.1 lists the sequence lengths with their respective numbers of revisits. The new sequence 22 is introduced in Section 4.2.2.

Table 4.1: KITTI sequences length and their number of points and revisits.

sequence	**00**	**02**	**05**	**06**	**07**	**08**	**22**
Points count	7.356e8	7.807e8	4.603e8	1.795e8	1.781e8	6.654e8	2.999e9
Scenes count	4541	4661	2761	1101	1101	4071	18236
Revisited scenes	777	296	425	268	28	321	2115

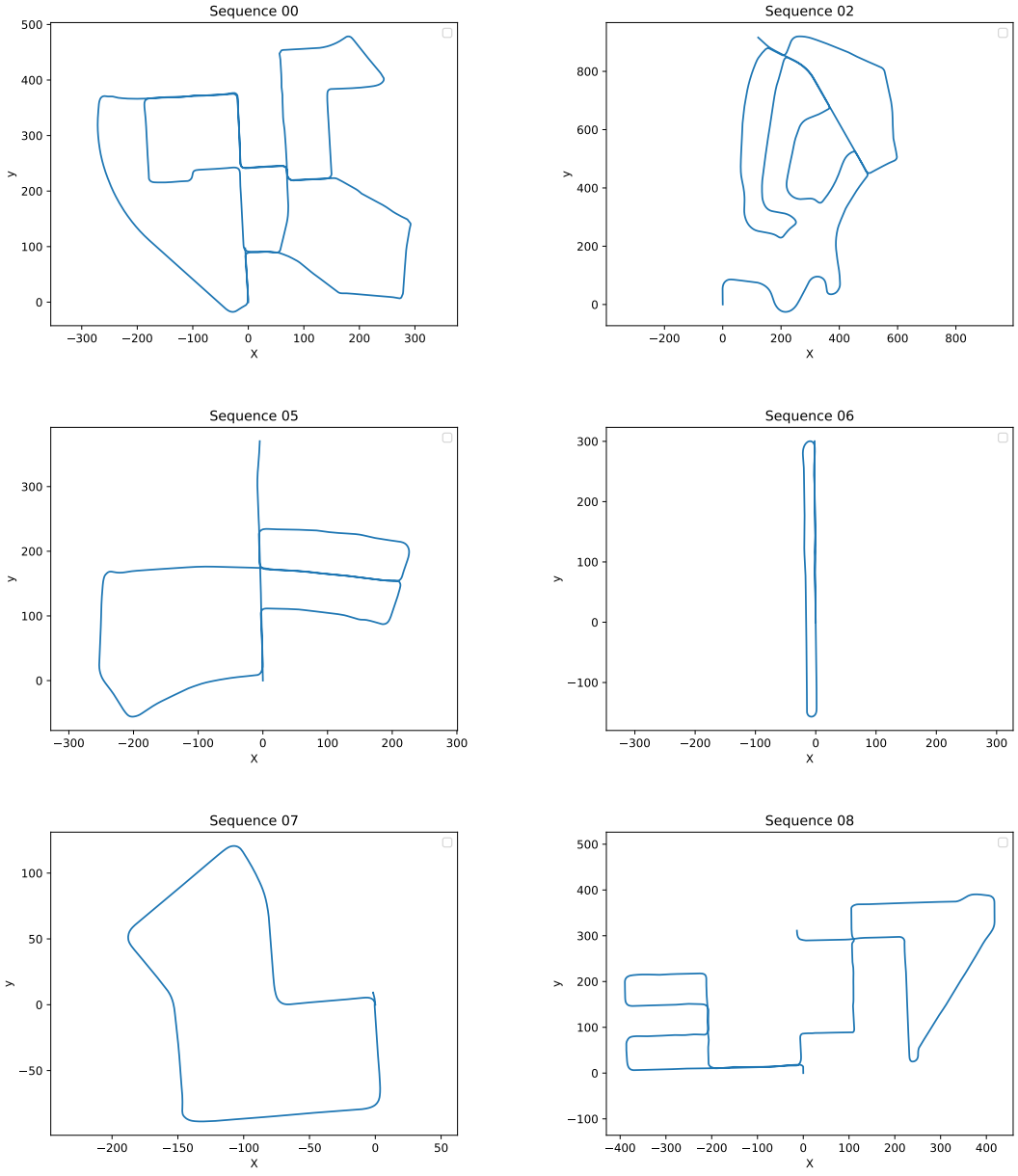


Figure 4.1: KITTI sequences trajectories (00, 02, 05, 06, 07, and 08), selected for containing revisits, shown in their local coordinate systems.

4.2.2 Building a new KITTI sequence

We propose a newly constructed “KITTI” sequence, named "22", which aggregates all scenes from the six selected sequences. This extended sequence serves as the first step toward evaluating the scalability of DSI-3D in large-scale retrieval scenarios. Since the original sequences are referenced in independent local coordinate frames, it is necessary to align them into a common reference frame. Sequence 00 is chosen as the base frame, and each subsequent sequence i is translated along the X and Y axes by $i \times 1000$ meters, where i is the order of addition, as shown in Figure 4.2.

Creating this new sequence also requires adapting the filenames. File naming is critical, as it serves as the key between point cloud IDs and their poses and is also a naive indexing strategy. To avoid duplicate filenames such as 000000.bin, which occur in all original sequences, the files in each subsequent sequence are renamed by adding the cumulative length of all preceding sequences to their original index. This results in a continuous numbering scheme, from 000000.bin to 018236.bin.

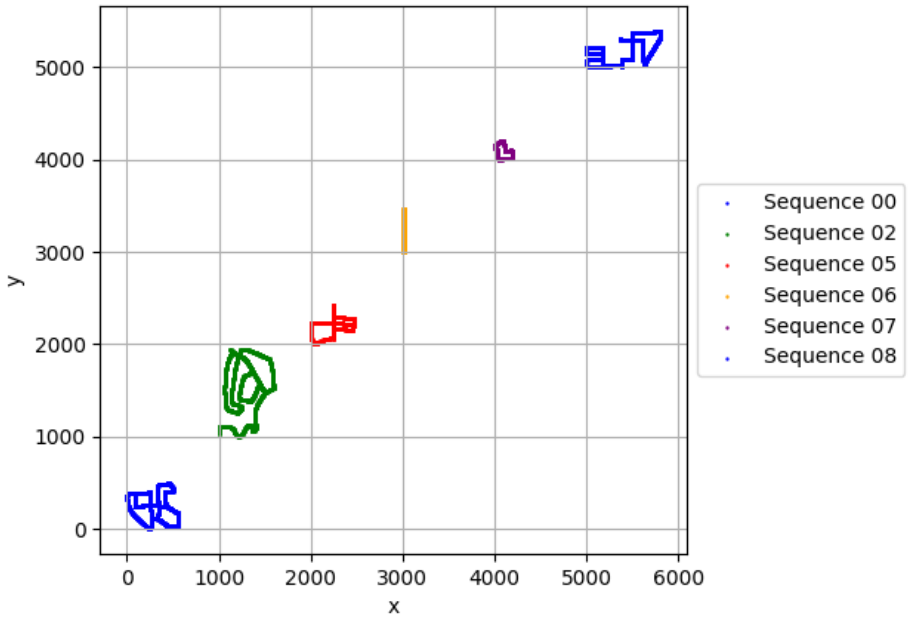


Figure 4.2: Construction of the additional KITTI sequence 22. The six original sequences, shown in color, are shifted along the X and Y axes, by a multiples of 1000 m and aligned into a common reference frame, with sequence 00 serving as the base.

4.2.3 Dataset splitting

A major difficulty is choosing the evaluation protocol of DSI-3D, which is inherently based on both information retrieval principles and the place recognition (PR) method. Place recognition descriptors are trained on dedicated training datasets and later evaluated on separate test datasets, whereas DSI requires training on the entire database, in order to retrieve the correct docids. To address this imbalance, data augmentation is considered a potential strategy: the model can be trained on every scene while transformations such as rotations, translations, and occlusions are applied to queries during inference. However, this approach introduces a critical trade-off. If the perturbations are too minor, the queries remain overly similar to the training data, offering little challenge. Conversely, if the perturbations are too strong, the resulting scenes may become unrecognizable, leading to an unfair degradation of retrieval performance.

To ensure independence between training and evaluation, the dataset is split rather than relying solely on data augmentation. Following a common practice, 80% of the data is allocated to the training set and 20% to the evaluation set. The separation is performed in a simple manner, point clouds whose naive index (last digit of their identifier) is a multiple of 5 are assigned to the evaluation set, while the remaining samples are assigned to the training set. This procedure also ensures that both datasets cover the entire sequence. The evaluation set is further split into two, with non-multiples of ten going into the validation set.

For example, the training set contains {"000001", "000002", "000003", "000004", "000006", ...}, the validation set {"000005", "000015", "000025", ...}, while the evaluation set contains {"000000", "000010", "000020"}. During evaluation, the model receives a query such as point cloud "000005" and is required to retrieve its closest neighbors, e.g., "000004" and "000006". For the retrieval with exact or approximate k-NN, the train set is used as a reference database upon which the hash table and the graph network are built. Table 4.2 presents the resulting scene distribution.

Table 4.2: Distribution of the KITTI sequence datasets into a training, validation and evaluation set.

sequence	**00**	**02**	**05**	**06**	**07**	**08**	**22**
scenes count	4541	4661	2761	1101	1101	4071	18234
Database size	3632	3728	2208	880	880	3256	14584
Validation set	454	466	276	110	110	407	1823
Evaluation set	455	467	277	111	110	408	1827

4.3 Experimental setup

In this section, we present the experimental setup used to train and evaluate DSI-3D. First, we present a study comparing triplet and quadruplet losses, inspired by GIT and LoGG3D-Net respective loss functions, in Section 4.3.1. Then, we motivate our choice of using vocabulary restrictions during tokens generation steps in Section 4.3.2. To finish, we present the metrics used for evaluating both DSI-3D and standard retrieval approaches in Section 4.3.3.

4.3.1 Contrastive learning

To reinforce the training by increasing the similarity between positive pairs, we use both the regular GIT loss and a variation of the language modeling L_m loss (Equation (3.1)) as triplet and quadruplet losses L_{tripl} and L_{quad} (both in $[0, +\infty[$), similar to Equation (2.2) in Chapter 2. These losses are computed using tuples of point clouds: (P_q, P_p, P_n) for the triplet one, and $(P_q, P_p, P_n, P_{nbis})$ for the quadruplet one, as follows:

$$L_{tripl}(P_q, P_p, P_n) = L_m(P_q) + \Big(L_m(P_p) - L_m(P_n) + \alpha\Big) \tag{4.1}$$

$$L_{quad}(P_q, P_p, P_n, P_{nbis}) = L_{tripl}(P_q, P_p, P_n) + \Big(L_m(P_p) - L_m(P_{nbis}) + \beta\Big) \tag{4.2}$$

We train DSI-3D on the six KITTI sequences for a maximum of 100 hours using an NVIDIA V100 GPU. Since the image encoder in the GIT pipeline is frozen, we precomputed and stored point-cloud descriptors for each scene. During training, these descriptors are loaded from disk, ensuring a deterministic descriptor association with each scene while avoiding redundant computations. This choice also means that no data augmentation (e.g., rotation, translation, noise, or occlusion) is applied to the scenes. As in (Vidanapathirana et al., 2022), the ground plane is preserved since its removal is shown not to affect evaluation performance. Hyperparameters α and β are set to 0.5 and 0.3, respectively.

We evaluated the impact of contrastive learning with a language modeling loss, as it required computing three or four GIT inferences per query triplet or quadruplet. Table 4.3 reports the $F1_{max}$ and $Hits@1$ scores obtained on the six original sequences using our model associated with the GNSS indexing strategy, comparing the triplet L_{tripl} and quadruplet L_{quad} losses at equivalent training epochs. To ensure informative results, we selected a negative distance threshold of 3 m, which constitutes a more challenging setting.

Table 4.3: Scores obtained at equivalent training steps of DSI-3D with contrastive loss function L_{tripl} and L_{quad}.

F1max	**00**	**02**	**05**	**06**	**07**	**08**	**mean**
L_{tripl}	0.9775	**0.9725**	**0.9946**	0.9479	0.9817	**0.9875**	0.9770
L_{quad}	**0.9809**	0.9713	0.9853	**0.9674**	**0.9955**	0.9825	**0.9805**
Hits@1	**00**	**02**	**05**	**06**	**07**	**08**	**mean**
L_{tripl}	0.9560	**0.9465**	**0.9892**	0.9009	0.9640	**0.9755**	0.9554
L_{quad}	**0.9626**	0.9422	0.9711	**0.9369**	**0.9909**	0.9704	**0.9624**

The results show that the quadruplet loss yields a marginally higher average $F1_{max}$ and $Hits@1$ score compared to the triplet loss. Based on this observation, we adopt the quadruplet loss for the remaining experiments.

4.3.2 Vocabulary restrictions

The generation of docids can be either constrained or unconstrained. In the unconstrained case, the model is allowed to generate any combination of tokens, which may result in docids that do not correspond to existing places. This approach avoids the need to check token validity during autoregressive decoding, but it requires additional post-processing to filter out hallucinated identifiers.

By contrast, constraining the generation ensures that only valid docids associated with existing places are produced. Although this introduces the overhead of verifying tokens during generation, we implement it efficiently using a dictionary-based lookup. In this structure, the keys correspond to all possible token sequence prefixes, and the values store the set of allowed next tokens. This design allows for token validation with an average complexity of O(1) (Python Software Foundation, 2024) (and a worst-case complexity of O(N)), making constrained generation computationally efficient while guaranteeing that the output docid exists in the database.

Here, we use the default tokenizer of GIT (Wang et al., 2022), git-base-coco, which has a vocabulary of 30522 tokens. Many of these tokens correspond to words, subwords, letters, and symbols; for example, ("look", "##ing", "a", "!"). In our use case, however, we only utilize digits to construct docids. Consequently, all non-digit subword tokens are filtered out, reducing the effective vocabulary size to 939 tokens.

4.3.3 Evaluation protocol

DSI-3D draws inspiration from both the field of place recognition and the principles of information retrieval. Consequently, two main criteria are used to evaluate it and baselines:

- The $Hits@N$ metric, used in information retrieval, reports the proportion of correct docids ranked within the top N predictions. While the beam search explores ten concurrent candidates, we focus on the more discriminative case with $N = 1$.
- To evaluate predictive performance from a place recognition perspective, we adopt retrieval metrics based on the maximum $F1$ score ($F1_{max}$) (see Equation 2.4 in Chapter 2. This approach is commonly used in place recognition because the optimal threshold on similarity scores is difficult to predict a priori.

For the $F1$ score, the binary classification follows a two-step process (Vidanapathirana et al., 2022), first, a similarity score between descriptors is compared to a threshold, second, the prediction is validated against geometric ground-truth criteria to determine whether it corresponds to a true or false match. The $F1_{max}$ score is then obtained by sweeping over different similarity thresholds and selecting the maximum value.

A limitation of this definition is that DSI-3D does not explicitly compute similarity scores between the query and candidate point clouds. Instead, the model directly produces logit values associated with candidate docids. To make the comparison fair, we choose to adapt the $F1$ evaluation by iterating over thresholds on the logit scores while maintaining the same geometric revisit criteria as in (Vidanapathirana et al., 2022). A top-1 retrieval is considered positive if its sequence score is below the maximum confidence threshold τ. A positive retrieval is classified as a true positive if it is within 3 m of the ground truth pose of the query, and as a false positive if it exceeds 20 m. In order to better distinguish each approach, we set the negative euclidean threshold at 3 m. With this criterion, a retrieved point cloud that is assigned high model confidence (low logit error) but has an inaccurate geolocation is considered a false positive. Figure 4.3 summarizes the updated binary classification used for DSI-3D predictions.

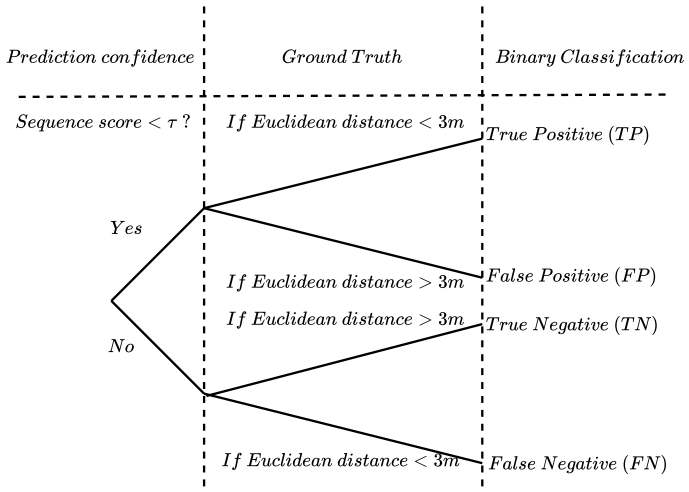


Figure 4.3: Binary classification of DSI-3D descriptors. The prediction confidence corresponds to the score of the top-ranked token sequence produced by the beam search, relative to a varying threshold τ. The binary classification relies solely on the geometric distance between the query and the ground truth, without the tolerance buffer (from 3 to 20 m), in order to better highlight the performance differences.

All experiments presented in this thesis are conducted on remote HPC[1] resources from GENCI–IDRIS, using a single V100 GPU.

4.4 Evaluation of DSI-3D

This section presents the evaluation of our adaptation of DSI. First, indexing strategies are compared to determine the most appropriate one for retrieving geometrically close point clouds in Section 4.4.1. Section 4.4.2 reports the retrieval performance of DSI-3D against

[1]High-performance computing

state-of-the-art benchmarks and Section 4.4.3 analyzes the computational complexity of our method relative to these baselines.

4.4.1 Indexing strategies

Having identified the most effective loss function for training, we proceeded to train a DSI-3D model on each sequence and for each point cloud indexing approach. Each training run lasted approximately 100 hours, and we selected the best checkpoint based on the validation scores. Training on the aggregated sequence "22" required around 400 hours, primarily due to slower convergence with the same parameters. Table 4.4 and Table 4.5 report the $F1_{max}$ and $Hits@1$ scores, respectively, for several DSI-3D variants.

The names **Label**, **Hierar**, **GNSS**, and **Hilbert** refer to the following DSI-3D indexing strategies introduced in Section 3.3.2:

- **Label**: *Naively Structured String Identifiers*,
- **Hierar**: *Semantically Structured Identifiers*,
- **GNSS**: *Positional Structured Identifiers (GNSS)*,
- **Hilbert**: *Hilbert-curve Positional Structured Identifiers*.

For the Hilbert curve Positional Structured identifiers, we index all point clouds using the Hilbert curve at iteration 17 (and iteration 16 for the sequence "00").

Table 4.4: Evaluation across KITTI sequences using various indexing strategies docids, based on LoGG3D-Net descriptors, using the $F1_{max}$ metric. The best results are bold and the second ones are underlined.

Indexing	00	02	05	06	07	08
DSI-3D with Label	0.8991	0.8774	0.8831	0.7954	0.8844	0.9044
DSI-3D with Hierar	0.9647	0.9316	0.9684	0.9626	<u>0.9817</u>	0.9684
DSI-3D with GNSS	<u>0.9809</u>	<u>0.9758</u>	<u>0.9891</u>	<u>0.9674</u>	**0.9955**	<u>0.9851</u>
DSI-3D with Hilbert	**0.9933**	**0.9781**	**0.9909**	**0.9817**	**0.9955**	**0.9913**

Table 4.5: Evaluation across KITTI sequences using various indexing strategies (exact and approximate), based on LoGG3D-Net descriptors, using the $Hits@1$ metric. The best results are bold and the second ones are underlined.

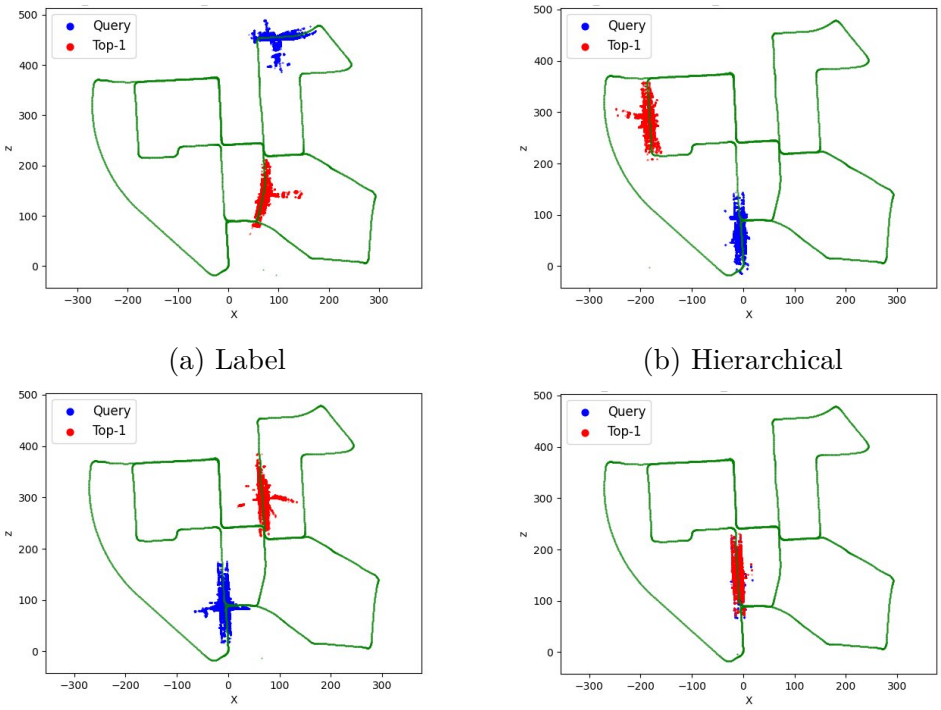
Indexing	00	02	05	06	07	08
DSI-3D with Label	0.8132	0.7645	0.7906	0.6306	0.7928	0.8235
DSI-3D with Hierar	0.9319	0.8630	0.9387	0.9279	0.9640	0.9387
DSI-3D with GNSS	<u>0.9626</u>	<u>0.9507</u>	<u>0.9783</u>	<u>0.9369</u>	<u>0.9909</u>	<u>0.9705</u>
DSI-3D with Hilbert	**0.9846**	**0.9550**	**0.9819**	**0.9640**	**0.9910**	**0.9828**

The first observation is that constructing docids using the Hilbert curve yields the best results in a place recognition context, where the spatial proximity between the prediction and the ground truth is important. This observation is further supported by the performance of the positional structured identifiers based on GNSS coordinates, which achieve the second-best results. In contrast, the hierarchical indexing strategy tends to prioritize descriptor similarity over spatial proximity. While this property could be advantageous

for retrieving geometrically similar scenes, it is less suited for precise place recognition. As illustrated in Figure 3.6a, the naive integer-based indexing achieves the lowest retrieval scores, as it fails to encode information about spatial relationships or revisits.

Errors visualization

To better assess the influence of the indexing strategies, we examine the worst-case retrieval scenarios, defined as cases where the distance between the ground truth and the retrieved point cloud is maximal. Figure 4.4 presents examples of the worst Top-1 retrieval errors (shown in red) returned by DSI-3D for query scans (shown in blue) from sequence 00. In Figure 4.4a, the query with index 002750 is incorrectly matched to 000278 due to their similar token patterns, illustrating that the model relies on token-level similarities rather than on spatial information. When using the Hierarchical strategy (Figure 4.4b), the model retrieves the most geometrically similar point cloud, even when it is not the closest in space. With the Positional Structured Identifiers (Figure 4.4c), only one error exceeds a 20 m distance, and even in that case, the retrieved scan remains geometrically consistent with the query. Finally, under the Hilbert Curve Positional Structured Identifiers (Figure 4.4c), the largest observed error is 3.95 m, indicating that the retrieved scans are always spatially relevant within the sequence. The prediction is only marked as incorrect because our evaluation threshold is set to 3 m.



(a) Label (b) Hierarchical

(c) GNSS (d) Hilbert

Figure 4.4: Visualization of the most distant Top-1 retrievals, in red, from the query in blue, using the DSI-3D model with different indexing methods: (a) Label, (b) Hierarchical, (c) GNSS, and (d) Hilbert. Distances and axes are in meters.

To complement these observations, Table 4.6 reports the maximum Top-1 retrieval errors across KITTI sequences for each indexing strategy. As expected, the Label and Hierarchical identifiers yield the largest mismatches, occasionally exceeding several hundred meters due to the lack of spatial structure in their indexing. Incorporating positional information with GNSS-based identifiers reduces these errors, but it still suffers from

inconsistencies across sequences. In contrast, the Hilbert-based indexing consistently achieves the lowest maximum errors across all sequences, except Sequence 02, confirming its ability to preserve spatial proximity and improve retrieval robustness.

Table 4.6: Worst Top-1 retrieval errors (in meter), returned by DSI-3D for each KITTI sequence. The best results are bold and the second ones are underlined.

Indexing strategy	**00**	**02**	**05**	**06**	**07**	**08**
DSI-3D with Label	316.8	526.11	296.18	109.15	81.16	106.82
DSI-3D with Hierar	284.72	771.56	435.44	<u>5.46</u>	5.77	484.69
DSI-3D with GNSS	<u>220.12</u>	**11.83**	<u>6.11</u>	100.09	<u>3.18</u>	<u>63.55</u>
DSI-3D with Hilbert	**3.95**	<u>165.19</u>	**4.03**	**4.52**	**3.09**	**7.60**

In conclusion, the retrieval behavior of DSI-3D can be regarded as at least as effective as conventional state-of-the-art approximate methods. In the following section, we further highlight its advantages in terms of computational complexity and retrieval time.

4.4.2 Positioning DSI-3D among retrieval approaches

Since we have determined the best indexing strategy, now we compare it to state-of-the-art baselines, which are:

- **Exact k-NN**: Exhaustive search,
- **LSH$_H$**: Locality-sensitive hashing with $H \in \{32, 256\}$ hash functions (see Section 2.4.3 from Chapter 2),
- **HNSW$_N$**: Hierarchical navigable small-world graphs with a neighborhood size $N \in$ 32 (see Section 2.4.4 from Chapter 2).

Table 4.7 and Table 4.8 report the $F1_{max}$ and $Hits@1$ scores, respectively, for these baselines and DSI-3D based on Hilbert curves.

Table 4.7: Evaluation across KITTI sequences using various indexing strategies (exact and approximate), based on LoGG3D-Net descriptors, using the $F1_{max}$ metric. The best results are bold and the second ones are underlined.

Indexing	**00**	**02**	**05**	**06**	**07**	**08**	**22**
Exhaustive k-NN	**0.9956**	**0.9859**	**1.0**	**0.9955**	**1.0**	**0.9975**	**0.9925**
k-d tree	**0.9956**	**0.9859**	**1.0**	**0.9955**	**1.0**	**0.9975**	**0.9925**
LSH$_{32}$	0.9416	0.8979	0.9681	0.9051	0.9670	0.9448	0.9290
LSH$_{256}$	<u>0.9945</u>	<u>0.9792</u>	<u>0.9946</u>	0.9815	0.9626	0.9888	<u>0.9850</u>
HNSW$_{32}$	**0.9956**	**0.9859**	**1.0**	**0.9955**	**1.0**	**0.9975**	**0.9925**
DSI-3D with Hilbert	0.9933	0.9781	0.9909	<u>0.9817</u>	<u>0.9955</u>	<u>0.9913</u>	0.9819

Table 4.8: Evaluation across KITTI sequences using various indexing strategies (exact and approximate), based on LoGG3D-Net descriptors, using the $Hits@1$ metric. The best results are bold and the second ones are underlined.

Indexing	**00**	**02**	**05**	**06**	**07**	**08**	**22**
Exhaustive k-NN	**0.9956**	**0.9722**	**1.0**	**0.9910**	**1.0**	**0.9950**	**0.9852**
k-d tree	**0.9956**	**0.9722**	**1.0**	**0.9910**	**1.0**	**0.9950**	**0.9852**
LSH_{32}	0.6110	0.5225	0.6173	0.5676	0.8108	0.6176	0.5049
LSH_{256}	0.9890	<u>0.9572</u>	<u>0.9892</u>	0.9550	0.9820	0.9755	<u>0.9693</u>
$HNSW_{32}$	<u>0.9912</u>	**0.9722**	**1.0**	**0.9910**	**1.0**	**0.9950**	**0.9852**
DSI-3D with Hilbert	0.9846	0.9550	0.9819	<u>0.9640</u>	<u>0.9910</u>	<u>0.9828</u>	0.9644

In terms of retrieval quality, the exact approaches obviously provide the best results. Among the approximate search acceleration methods, the graph-based approach (HNSW) maintains performance closest to that of an exhaustive search. In comparison, accelerating point cloud retrieval with Locality-Sensitive Hashing (LSH) degrades retrieval quality, depending on the chosen projection parameter H. Overall, the current version of DSI-3D does not yet reach the performance level of the exhaustive baseline, which achieves a slightly higher $Hits@1$ score, with an average improvement of 1.6% (median 1.3%). For $F1_{max}$, the Hilbert-based indexing remains 0.8% lower on average (median 0.5%). Nevertheless, DSI-3D exhibits performance comparable to that of a hashing-based method such as LSH. On average, DSI-3D with Hilbert curve indexing even surpasses descriptors with LSH_{256} by 0.4% in $F1_{max}$ (median 0.6%) and by 0.1% in $Hits@1$ (median 0.6%). These results mark a promising step toward generative retrieval. However, the relatively small size of the current database may limit the differentiation between methods, as approximate approaches like HNSW still achieve nearly exhaustive-level performance. The next section examines the worst-case retrieval scenarios in more detail.

4.4.3 Complexity analysis

A notable advantage of DSI-3D lies in the fact that its search complexity does not depend on the size of the database. Indeed, this load is transferred to the training phase, unlike the LSH and HNSW methods, presented in Section 2.4 in Chapter 2, whose complexity, although sublinear, increases with the size of the dataset.

For 3D place recognition methods, the retrieval is performed over the entire reference dataset through sequential comparisons. Consequently, in approaches like LoGG3D-Net, the retrieval time scales linearly with the dataset size n, yielding a time complexity of $O(n)$. Other methods, such as PointNetVLAD or TransLoc3D, used a tree-based structure (see Section 2.4.2 in Chapter 2), have a theoretical sub-linear complexity $O(D \cdot n^{1-\frac{1}{D}})$, but they tend to be slower due to the curse of dimensionality caused by the high-dimensional descriptors. As presented in Sections 2.4.3 and 2.4.4 of Chapter 2, the similarity search in high-dimensional spaces can be accelerated using ANN techniques such as LSH and HNSW. They enable also sub-linear retrieval time, LSH being bounded by $O(n^{\rho})$, with $0 \leq \rho \leq 1$ and HNSW having an average retrieval complexity of $O(\log n)$. In contrast, DSI-3D performs retrieval via a single model inference, with a constant-time complexity of $O(1)$ due to the fixed number of tokens per sequence, beams width and vocabulary size. This property provides a significant advantage when scaling to large point cloud datasets. Table 4.9 presents the average retrieval time for the methods used in our benchmark.

With LSH_{256}, the retrieval time is 6.72 times faster, 8.10 times faster with LSH_{32} (although with poor retrieval quality), and 5.67 times faster with HNSW. For DSI-3D

Table 4.9: Retrieval time (in ms) required to obtain the Top-10 nearest neighbors under the new place recognition protocol.

Sequence	**00**	**02**	**05**	**06**	**07**	**08**	**22**
database size	3632	3728	2208	880	880	3256	14584
Exhaustive search (Top-1)	2.4	2.3	1.4	0.6	0.6	2.1	10.5
Exhaustive search	2.6	2.7	1.6	0.7	0.7	2.4	10.9
k-d tree	2.6	2.6	1.6	0.7	0.7	2.3	9.8
LSH_{256}	0.2	0.3	0.2	0.2	0.1	0.3	0.4
HNSW_{32}	0.3	0.4	0.3	0.2	0.2	0.3	0.4
DSI-3D label	62.1	63.0	63.2	63.4	65.5	61.5	/
DSI-3D GNSS	92.1	93.5	90.7	95.0	96.8	93.5	/
DSI-3D Hierar	59.7	55.4	54.2	52.1	58.6	59.9	/
DSI-3D Hilbert	93.3	94.5	90.4	86.6	91.7	89.8	95.4

indexed with *Positional Structured identifiers*, the current retrieval time is 0.095s, measured by the time it took for a descriptor to pass through all GIT layers and return the docid.

It is worth noting that the current code implementation used is not yet fully optimized. Nevertheless, to provide insight into the current scalability of DSI-3D, we can fit a linear equation (see Figure 4.5) between the size of the reference dataset (N_{desc}) and retrieval time (T) with exhaustive retrieval, such as:

$$T = 0.75N_{desc} - 0.03 \tag{4.3}$$

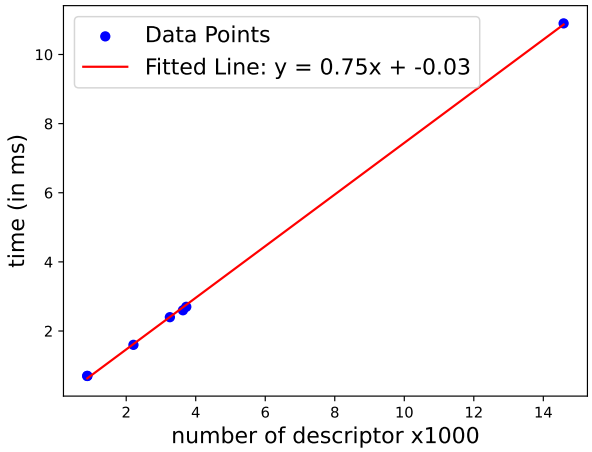


Figure 4.5: Linear fit of the retrieval time cost of exhaustive search, according to the database size of KITTI sequence (including sequence 22).

This allows us to estimate that the inversion point occurs at $N_{desc} = 126.7\text{e}3$ descriptors in favor of DSI-3D facing exhaustive retrieval. According to the settings used, this amount roughly corresponds to a spatial coverage of 8.2 km^2, at which point DSI-3D would become faster in its current implementation.

Among the factors affecting the retrieval time of DSI-3D is the number of beams explored during the beam search, which is fixed at 10, as well as the indexing strategies that affect decoding steps. Reducing the number of beams could improve retrieval time but may result in less precise responses. Such an evaluation will be part of future experiments.

4.5 Experimenting on DSI-3D hyperparameters

This last section studies the behavior of DSI-3D hyperparameters. We start by measuring the benefits and costs of constraining token generation and applying various sizes of beam search width in Section 4.5.1. We explore training configurations to decrease our initial DSI-3D training costs in Section 4.5.2.

4.5.1 Beam search width and vocabulary restrictions effects

To better understand the trade-offs governing the retrieval performance and efficiency of DSI-3D, we perform an ablation study focusing on two critical decoding factors: beam search width and vocabulary restrictions. The beam search width controls the number of concurrent decoding hypotheses considered at each generation step, directly affecting both retrieval accuracy and computation time. The vocabulary restrictions, on the other hand, determines which tokens are allowed during docid generation. Unlike vocabulary size, which was already reduced to digit-only subword tokens (939 total), this restricts the model to generate only valid token sequences corresponding to existing docids. Without this restriction, the model may generate syntactically valid but non-existent docids, degrading retrieval consistency.

We first investigate the impact of the beam search width parameter, which controls the number of concurrent decoding hypotheses explored during docid generation. The default configuration uses a beam width of ten. Tables 4.10 and 4.11 report the retrieval performance of DSI-3D, in terms of $F1_{max}$ and $Hits@1$, respectively, using the Hilbert curve indexing strategy.

Table 4.10: Effect of beam search width on retrieval performance using the Hilbert curve indexing strategy and LoGG3D-Net descriptors, evaluated with the $F1_{max}$ metric. The best scores are in bold.

Indexing	Beam width	00	02	05	06	07	08	22
DSI-3D with Hilbert	10	**0.9933**	**0.9781**	**0.9909**	**0.9817**	**0.9955**	**0.9913**	**0.9819**
DSI-3D with Hilbert	2	0.9911	0.9768	**0.9909**	0.9770	0.9909	**0.9913**	0.9815

Table 4.11: Effect of beam search width on retrieval performance using the Hilbert curve indexing strategy and LoGG3D-Net descriptors, evaluated with the $Hits@1$ metric. The best scores are in bold.

Indexing	Beam width	00	02	05	06	07	08	22
DSI-3D with Hilbert	10	**0.9846**	**0.9550**	**0.9819**	**0.9640**	**0.9910**	**0.9828**	**0.9644**
DSI-3D with Hilbert	2	0.9824	0.9507	**0.9819**	0.9549	**0.9910**	**0.9828**	0.9611

To assess whether this reduction provides a favorable trade-off between speed and accuracy, Table 4.12 reports the average retrieval time per query under both configurations.

Table 4.12: Average retrieval time (in ms) as a function of beam search width using the Hilbert curve indexing strategy. The best scores are in bold.

Indexing	Beam width	00	02	05	06	07	08	22
DSI-3D with Hilbert	10	**93.3**	**94.5**	**90.4**	**86.6**	**91.7**	**89.8**	**95.4**
DSI-3D with Hilbert	2	70.4	71.9	67.5	69.7	72.4	68.9	72.2

Reducing the beam width from 10 to 2 slightly decreases the $F1_{max}$ and $Hits@1$ scores by 1%, but reduces the average retrieval time by 25%. This suggests that DSI-3D's decoding process is robust to smaller search spaces, and a valid way of accelerating the search. For the remainder of this thesis, we keep the beam width fixed at ten.

Unrestricted generation

The vocabulary restrictions, introduced in Section 4.3.2, limits the number of tokens that are considered at each decoding step. This ensures that the model generates tokens of sequence to represent docids (Section 2.5.3 from Chapter 2) that can be retrieved in the database, at the cost of adding a hash table lookup operation. We analyze the impact of removing vocabulary restrictions during the generation process. When unrestricted, the model gains expressive flexibility but risks generating invalid or unreferenced docids, which can reduce overall retrieval accuracy. Tables 4.13 and 4.14 show that constrained generation leads to more stable and consistent retrievals, with a $F1_{max}$ and a $Hits@1$ scores higher by 4% and 8% respectively.

Table 4.13: Effect of vocabulary restrictions on retrieval performance using the Hilbert curve indexing strategy and LoGG3D-Net descriptors, evaluated with the $F1_{max}$ metric. The best results are in bold.

Indexing	Restriction	00	02	05	06	07	08	22
DSI-3D with Hilbert	Yes	**0.9933**	**0.9781**	**0.9909**	**0.9817**	**0.9955**	**0.9913**	**0.9819**
DSI-3D with Hilbert	No	0.9563	0.9425	0.9498	0.9479	0.9815	0.9618	0.9166

Table 4.14: Effect of vocabulary restrictions on retrieval performance using the Hilbert curve indexing strategy and LoGG3D-Net descriptors, evaluated with the $Hits@1$ metric. The best results are in bold.

Indexing	Restriction	00	02	05	06	07	08	22
DSI-3D with Hilbert	Yes	**0.9846**	**0.9550**	**0.9819**	**0.9640**	**0.9910**	**0.9828**	**0.9644**
DSI-3D with Hilbert	No	0.9143	0.8844	0.8917	0.9009	0.9550	0.9265	0.8350

Table 4.15 presents the average retrieval time of DSI-3D, with and without restrictions. The beam search width was set to ten for both cases. Removing the restrictions during token generation leads to an acceleration of 10% of the retrieval time.

Table 4.15: Average retrieval time (in ms) as a function of vocabulary restrictions using the Hilbert curve indexing strategy. The best results are in bold.

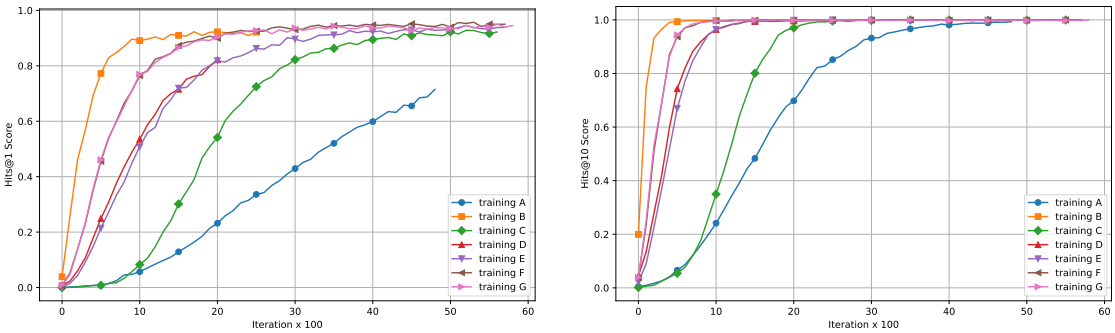
Indexing	Restriction	00	02	05	06	07	08	22
DSI-3D with Hilbert	Yes	**93.3**	**94.5**	**90.4**	**86.6**	**91.7**	**89.8**	**95.4**
DSI-3D with Hilbert	No	85.7	82.8	79.7	82.9	80.0	81.1	83.0

Overall, these results confirm that both a moderate beam width and the use of vocabulary restrictions improve the efficiency and reliability of the generative retrieval of DSI-3D. This study also showed that compromises can be made to accelerate retrieval while maintaining reasonable performance. The safest way to accelerate the retrieval with minimal impact on performance is by decreasing the beam width. For the remainder

of this work, we will keep the beam search width set at ten to allow for further re-ranking and to maintain the vocabulary restrictions active.

4.5.2 Optimizing training configuration

We showed that our initial implementations of DSI-3D achieve performance comparable to that of other retrieval approaches. To counterbalance the high training cost of the model, we investigated variations of the hyperparameters. Training A corresponds to the current model presented in Section 4.4.2. It was trained over four jobs of 100 hours each, although Figure 4.6 displays only the first quarter of this training. The other trainings, labeled B through G, represent hyperparameter variations, visible in Table 4.16.



(a) Hits@1 validation score

(b) Hits@10 validation score

Figure 4.6: Validation scores across training steps, measured by *Hits*@1 (a) and *Hits*@10 (b), obtained while optimizing the training hyperparameters of DSI-3D on the sequence 22, with the Hilbert curve indexing. *Training A* corresponds to the first quarter of the full model training used in Section 4.4.2, while the other training represents variations with different hyperparameters. A complete list of hyperparameters is provided in Table 4.16.

Table 4.16: Training hyperparameters for Training A to H used in the optimization experiments of DSI-3D on KITTI sequence 22.

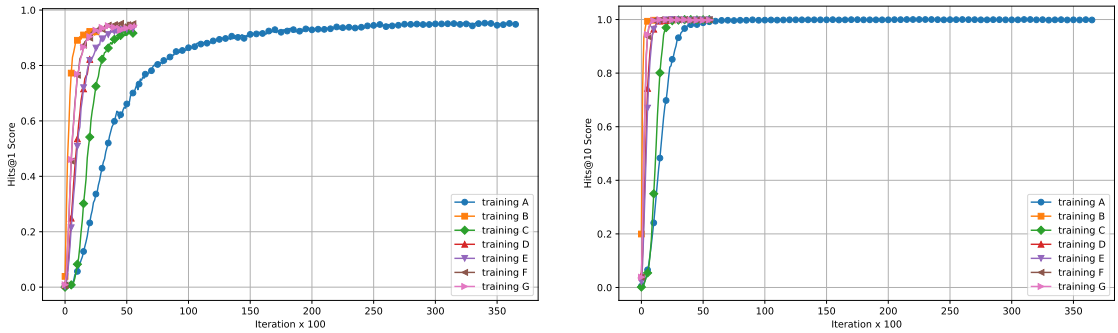
Training	**A**	**B**	**C**	**D**	**E**	**F**	**G**
Model	DSI-3D	**DSI-3D**	DSI-3D	DSI-3D	DSI-3D	DSI-3D	DSI-3D
Training samples	18236	**18236**	18236	18236	18236	18236	18236
Batch size	512	**256**	256	256	256	256	256
Learning rate (LR)	$1e-5$	**5e-4**	$5e-5$	$5e-5$	$5e-5$	$1e-4$	$1e-4$
Maximum training time	400h	**24h**	48h	24h	48h	48h	48h
LR scheduler	linear	**cosine**	linear	linear	cosine	cosine	cosine
Warmup steps	1000	**100**	2000	1000	100	100	100
Max epochs	10000	**10000**	10000	10000	10000	10000	10000
Max token length	18	**18**	18	18	18	18	18
Data augmentations	Disabled	**Disabled**	Disabled	Disabled	Disabled	Disabled	Disabled

Besides training A, which converges at a linear rate, the other runs show a much faster score increase during the early steps. One key factor influencing convergence in the first epochs is the number of warmup steps. Warmup is used to start training with a small learning rate and gradually increase it to the standard learning rate, after which the chosen scheduler strategy takes over.

Training C, with 2000 warmup steps, shows a slower start and ultimately achieves a lower maximum value. In contrast, the training using a cosine-based learning rate

scheduler (B, E, F, G) consistently outperforms those relying on a linear scheduler (A, C, D). Using a smaller batch size increases the variance of the gradient estimates, which can encourage the model to explore a broader solution space and potentially escape sharp local minima. The best training hyperparameters came from training B, which achieved the fastest convergence by using a smaller number of warmup steps, a cosine-based scheduler, and a higher learning rate of $5e-4$, achieving a $Hits@1$ score of 0.92 in 24 hours.

To better illustrate the effect of the learning rate on convergence, Figure 4.7 shows the complete validation scores for training A, which represents 400 hours of computation.



(a) Hits@1 validation score

(b) Hits@10 validation score

Figure 4.7: Validation scores across training steps, measured by $Hits@1$ (a) and $Hits@10$ (b), obtained while optimizing training hyperparameters for Training A to G, with complete list of validation scores of Training A, used in the optimization experiments of DSI-3D on KITTI sequence 22.

4.6 Conclusion

To conclude this chapter, Section 4.6.1 reviews the various DSI-3D models presented in terms of retrieval performance, inference time, and training costs. Section 4.6.2 resumes our takeaways on applying a DSI model to index and retrieve point clouds.

4.6.1 Overall assessment

This section proposes an overview of all the experiences performed on DSI-3D in this chapter in Table 4.17. In order to compare all methods, we only show model performance on the KITTI sequence 00, where a trained DSI-3D model exists for each configuration. Complementary experimentation on the beam width and vocabulary restrictions are added. The retrieval time of the classical approaches is measured based on a top-10 nearest neighbor search.

Table 4.17: Overview of retrieval performance on KITTI sequence 00 using various indexing strategies, metrics, beam widths, and vocabulary restrictions. The first and second best results of each group are in bold and underlined, respectively.

Indexing	**Hits@1**	**F1$_{max}$**	**Beam width**	**Vocab restriction**	**Retrieval time (ms)**
Exhaustive k-NN	**0.9956**	**0.9956**	/	/	2.6
k-d tree	**0.9956**	**0.9956**	/	/	2.6
LSH$_{32}$	0.6110	0.9416	/	/	0.2
LSH$_{256}$	0.9890	<u>0.9945</u>	/	/	0.2
HNSW$_{32}$	<u>0.9912</u>	**0.9956**	/	/	0.3
DSI-3D with Label	0.8132	0.8991	10	Yes	62.1
DSI-3D with Hierar	0.9319	0.9647	10	Yes	59.7
DSI-3D with GNSS	<u>0.9626</u>	<u>0.9809</u>	10	Yes	92.1
DSI-3D with Hilbert	**0.9846**	**0.9933**	10	Yes	93.3
DSI-3D with Label	0.8132	0.8981	2	Yes	51.7
DSI-3D with Hierar	0.9297	0.9647	2	Yes	44.7
DSI-3D with GNSS	<u>0.9582</u>	<u>0.9809</u>	2	Yes	70.4
DSI-3D with Hilbert	**0.9824**	**0.9911**	2	Yes	70.4
DSI-3D with Label	0.8088	0.8945	10	No	71.4
DSI-3D with Hierar	<u>0.9121</u>	<u>0.9540</u>	10	No	52.2
DSI-3D with GNSS	0.8396	0.9235	10	No	93.7
DSI-3D with Hilbert	**0.9143**	**0.9563**	10	No	85.7

As mentioned before, the exhaustive search and k-d tree remains the most efficient retrieval methods according to the two metrics. However, in incorporating the retrieval cost, HNSW becomes an excellent trade-off for being ten times faster than exhaustive search while obtaining equivalent $F1_{max}$ scores and with a slower complexity growth alongside the database.

For DSI-3D, the Hilbert curve positional structured identifier is the most effective docids representation of point clouds for place recognition in this setup, having a similar score to LSH$_{256}$ across all KITTI sequences. The second best indexing strategy, except on unrestricted vocabulary, is the positional structured string identifier, also benefiting from position information represented inside the docids. Semantically structured string identifiers may be more relevant for tasks where the similarity is more important than the proximity to a ground truth. However, naively structured string identifiers are not recommended for point cloud retrieval.

DSI-3D inference times are much longer than standard approaches but it does not depend on the size of the database since the growing costs intervene instead during the model training. A way to accelerate DSI-3D is to use indexing strategies resulting in shorter docids like Semantically structured string identifier, which is almost two times faster than Hilbert-curve indexes. Reducing the beam search width is also a good trade-off between acceleration and slight performance drops. Removing the vocabulary restriction is more risky, as it is not really a reliable way to accelerate the model for a high performance drop.

4.6.2 Chapter conclusions

In this chapter, we have evaluated the DSI-3D models, training on variants of point clouds indexes facing standard retrieval approaches, on the KITTI dataset. To successfully train

DSI-3D on KITTI, we adapted the quadruplet loss principle to the language modeling loss used in GIT. Furthermore, we filtered the pre-trained vocabulary to retain only digit-based subword tokens and constrained token generation at each decoding step to ensure that the model produced valid docids.

Our experiments on the KITTI dataset, including the construction of an extended Sequence 22, showed that DSI-3D, trained with a positional structured identifier based on Hilbert curve distance, achieves retrieval performance comparable to state-of-the-art baselines while offering a scalable alternative. For place recognition tasks, where spatial proximity between the retrieved and ground-truth locations is crucial, positional structured identifiers offer a clear advantage. By encoding geometric relationships directly into the identifiers, they produce smaller retrieval distance errors and consequently yield improved overall performance compared to standard indexing strategies.

This setup allowed us to study the impact of vocabulary restrictions and beam search width on retrieval performance. Depending on the application and whether the confidence score is incorporated into the decision process, reducing the beam search width can provide a simple means of acceleration with minimal degradation in accuracy. However, removing the vocabulary restrictions is riskier, as it leads to a more significant performance drop while offering only limited gains in retrieval speed compared to the beam width.

Finally, hyperparameter fine-tuning was explored to reduce training time while maintaining comparable retrieval performance. This will be useful for the remainder of this work, as many training executions need to be repeated.

Overall, these results establish DSI-3D as a promising first step of retrieval toward scalable 3D place recognition using language-inspired retrieval models. In the next chapter, we build upon these insights to explore the IGN High-Definition LiDAR dataset, addressing the challenges of transposing our model to an aerial-based point cloud dataset with different characteristics.

Chapter 5

LiDAR HD as a place recognition dataset

Contents of this Chapter

5.1 Introduction 76
5.2 LiDAR HD dataset overview 77
5.2.1 Acquisition and characteristics 77
5.2.2 Dataset preprocessing and analysis 79
5.3 LoGG3D-Net applied to LiDAR HD 82
5.3.1 Evaluation of the pre-trained LoGG3D-Net model 82
5.3.2 Fine-tuning LoGG3D-Net on LiDAR HD 83
5.3.3 Evaluation of the fine-tuning 84
5.3.4 Training data augmentation 91
5.4 Conclusion 91

5.1 Introduction

The previous chapter demonstrated that our proposal achieved performance comparable to other approximate nearest neighbor methods on the KITTI sequences (Geiger et al., 2013), comprising approximately 18000 scenes. Even when including all KITTI sequences with ground truth, the dataset totals only around 23000 scenes, or 43000 if the last ten sequences without poses are considered.

Although several other place recognition datasets exist, such as Oxford RobotCar (Maddern et al., 2017), MulRan (Kim et al., 2020), or Alita (Yin et al., 2022), most provide limited geographic diversity. As they rely on a mobile mapping acquisition process, they are constrained to areas accessible by roads. Additionally, their fixed viewpoints do not capture elevated geometries, such as rooftops, which limit their generalizability as a reference dataset.

To evaluate the applicability of DSI-3D on different types of data, we chose the LiDAR HD ("LiDAR Haute Densité" or high density LiDAR) dataset. This dataset covers the French territory and provides an average point density of 10 points/m^2. It originates from the French LiDAR HD program, launched in 2020[1], which aims to produce a high-resolution digital twin of the entire French territory[2]. Compared to KITTI's 20000 available scans, the LiDAR HD dataset results from a country-wide, tile-based acquisition

[1] https://geoservices.ign.fr/sites/default/files/2025-03/SE_LiDAR_HD.pdf
[2] https://geoservices.ign.fr/lidarhd

campaign that spans a wide variety of environments. This large-scale coverage is complemented by the flexibility to define scene boundaries and control spatial overlap between adjacent scenes, enabling tailored dataset configurations.

Beyond research applications, the LiDAR HD program is also motivated by operational use cases such as urban planning[3], forest surveying[4], and flood risk prediction[5]. This chapter presents the LiDAR HD dataset, describing its acquisition conditions, intrinsic properties, and potential as a large-scale benchmark for place recognition. Section 5.2 details the technical characteristics and acquisition setup, while Section 5.3 explains the preprocessing steps performed to adapt them to our DSI-3D approach.

5.2 LiDAR HD dataset overview

This section presents the LiDAR HD dataset in the context of place recognition. Section 5.2.1 introduces its coverage and the two selected study zones in the Greater Paris area. Because the dataset is distributed in 1 km^2 tiles, which are too large for our purposes, Section 5.2.2 explains its partitioning into smaller, overlapping point clouds. Finally, we compare the resulting coverage with existing datasets dedicated to place recognition.

5.2.1 Acquisition and characteristics

The LiDAR HD program, initiated in March 2020 by the French National Institute of Geographic and Forest Information (IGN), aims to produce a high-resolution 3D model of the French territory every five years. Data acquisition began in the spring of 2021 and is carried out through airborne LiDAR surveys. Each acquisition block covers an area of 50 km $\times$ 50 km, which is then divided into 1 km $\times$ 1 km tiles. The acquisition progress can be seen in figure 5.1.

[3] https://geoservices.ign.fr/amenagement-du-territoire
[4] https://geoservices.ign.fr/foret
[5] https://geoservices.ign.fr/prevention-des-risques

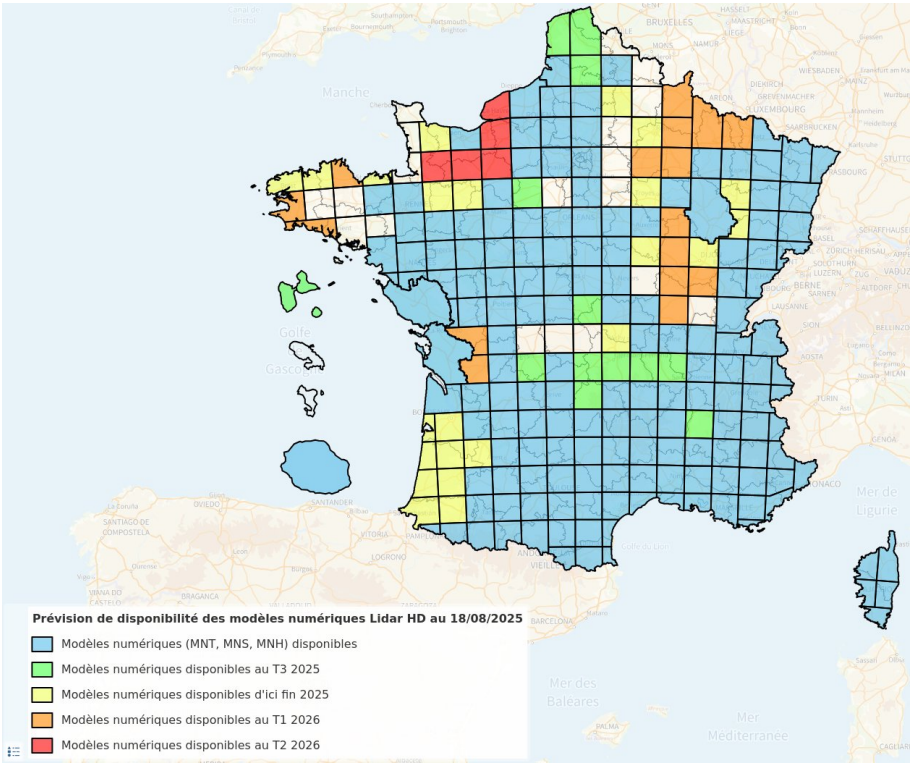


Figure 5.1: LiDAR HD acquisition progress. Blue area indicate data already available, while the remaining areas, shown up to red, will be progressively released by mid 2026. Image taken from IGN geoservices [6].

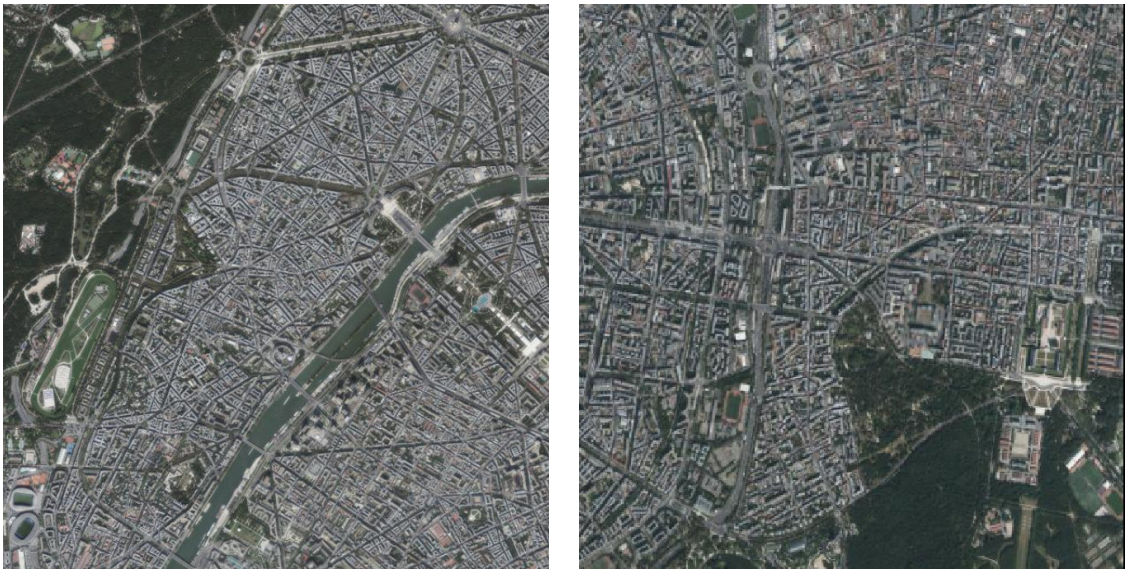
The LiDAR HD data are delivered in .las format, a binary standard for 3D point clouds. Each point includes geometric coordinates, intensity values, and classification metadata. A minimum of 10 points/m^2 is guaranteed, ensuring high-resolution coverage suitable for detailed mapping. However, this density is not guaranteed at altitudes above 2500 m. The altimetric accuracy is estimated at $\pm$10 cm, and the planimetric accuracy at $\pm$50 cm. Each point in the dataset is annotated with a semantic label drawn from a predefined taxonomy of 11 classes, ranging from ground and vegetation to buildings. Semantic segmentation is automated using the Myria3D deep learning model (Gaydon, 2022), which is designed for aerial LiDAR. Additionally, quality checks of the segmentation are conducted through automatic processes, semi-supervised verifications, and manual sample analysis. Although the dataset contains rich semantic annotations, this thesis focuses exclusively on the geometric structure of the point cloud without exploiting semantic labels or LiDAR intensity values.

We selected the Paris area as our primary area of study due to its diverse urban environments, which provide a challenging dataset for large-scale place recognition. We acknowledge that using only Paris area LiDAR acquisitions may limit the representativeness of our dataset, as it does not capture the diversity of French landscapes, such as rural, mountainous, or coastal environments. This choice was motivated by the simplified transition from the KITTI dataset, which covers the city of Karlsruhe, where urban environments are more relevant. This chapter explores how our methods could be extended to new types of environments. In particular, we consider two areas:

- The eastern part of Paris, named $Paris_{East}$, extends from Saint-Mandé to Vincennes (Figure 5.2b),
- The western part of Paris, named $Paris_{West}$, extends across parts of the Bois de Boulogne up to the Champ-de-Mars (Figure 5.2a).

[6] https://geoservices.ign.fr/lidarhd

The chosen regions encompass areas with varying urban densities, including different types of buildings, significant vegetation such as the Bois de Vincennes and Bois de Boulogne, and water features including the Seine and lac de Saint-Mandé.



(a) $Paris_{West}$ (16km^2) (b) $Paris_{East}$ (9km^2)

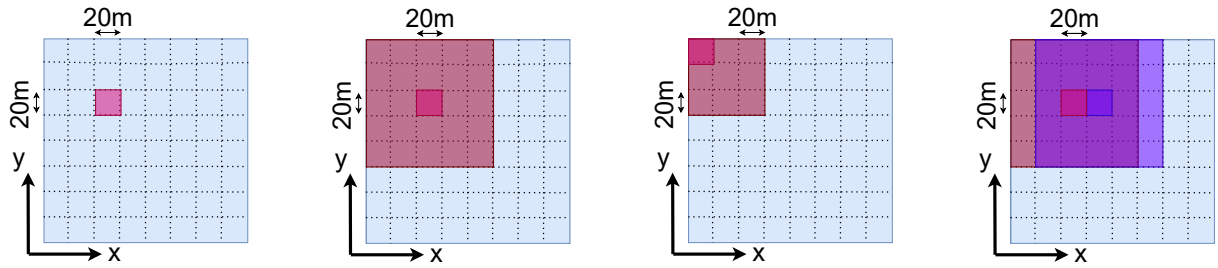
Figure 5.2: Aerial view[7] of the two selected Paris areas : (a) $Paris_{West}$, containing 93378 point clouds for a total of 9.904e9 points; (b) $Paris_{East}$, containing 54863 point clouds for a total of 5.903e9 points.

5.2.2 Dataset preprocessing and analysis

To prepare this dataset, with the objective of training a point cloud encoder, several conditions must be satisfied, including a sufficient number of scenes to split into training sets and the existence of revisits for validation. The original LiDAR HD data are provided as tiles of 1 km^2, which are too coarse to be used for reference pairing, as ground-based queries tend to have much smaller spatial coverage. The tile format also lacks spatial continuity due to the absence of overlaps between tiles.

To construct our dataset, we drew inspiration from the CS-Campus3D aerial dataset (Guan et al., 2023), aiming to create a comparable benchmark for future multi-source retrieval. To obtain finer spatial granularity, we subdivided each tile into a regular grid: 150×150 cells for the eastern Paris area and 200×200 m cells for the western Paris area. This resulted in core point clouds of 20×20 m in size (see Figure 5.3a). To construct input scenes for point cloud retrieval, each core cell located at position (x, y) was expanded by aggregating its neighboring cells observed within the range $[x \pm 2, y \pm 2]$, yielding a point cloud of 100×100 m composed of 25 sub-cells, in order to match the 100 meter patch size of CS-Campus3D (see Figure 5.3b).

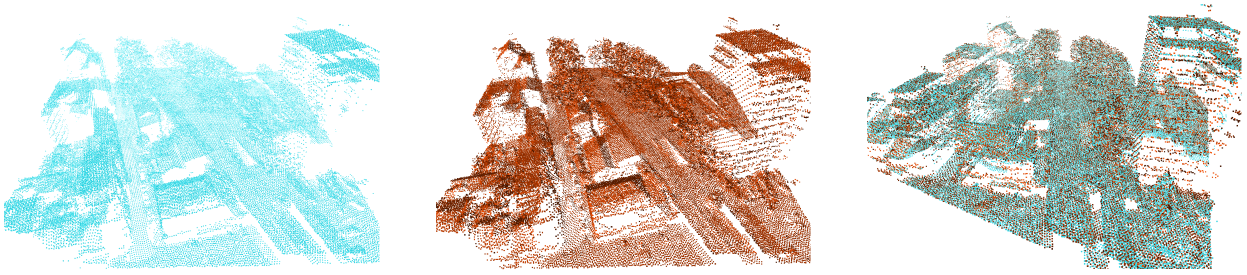
Figure 5.3c illustrates edge cases that occur when a cell does not have two neighbors in every direction, for example, near tile or flight ID boundaries, or when the cell is empty due to water bodies or missing data. The choice of the 20 m grid is also motivated to mimic the spatial distribution existing in dataset KITTI, in order to maintain similar spatial distributions with our first contribution from Chapter 3. Since each 100 m point cloud was constructed from 25 cells, it has two overlapping neighbors in each direction, similar to the overlap structure in KITTI sequences. With this construction, consecutive point clouds, such as (x, y) and $(x + 1, y)$, exhibit an overlap of approximately 80%, ensuring scene continuity between adjacent scenes (see Figure 5.3d).



(a) Grid cell (b) Point cloud (c) Edge cases (d) Overlaps

Figure 5.3: Construction of point clouds from LiDAR HD tiles. (a) Tiles (blue) are divided into grid cells of 20×20m (pink). (b) Neighboring cells sharing the same acquisition ID are aggregated to form a 100×100m point cloud. (c) Example of edge cases. (d) The process (b) is repeated for every cell.

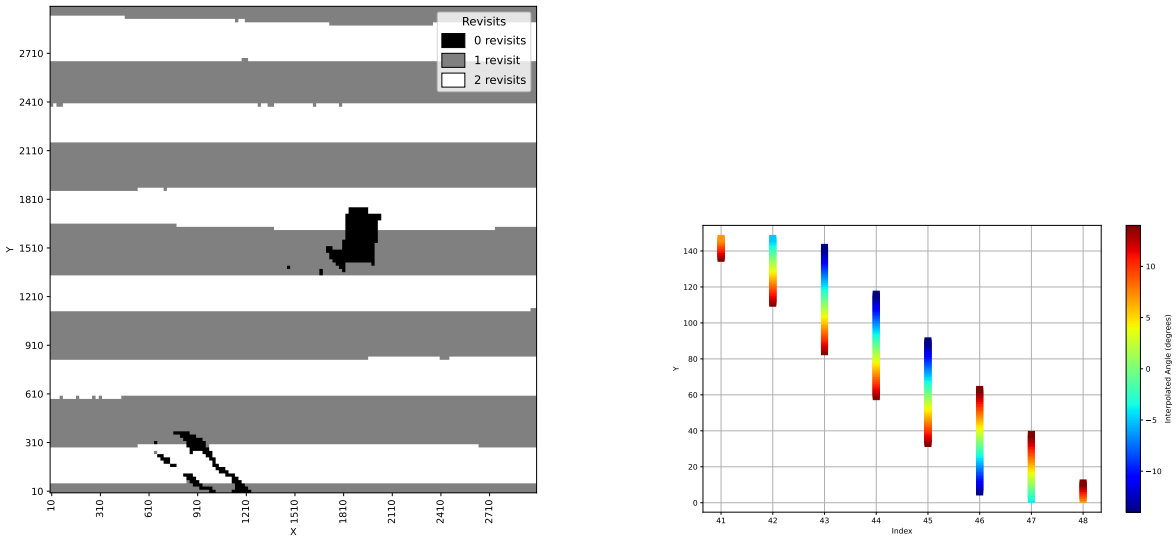
In this dataset, the definition of revisit comes from the acquisition process. Several aerial LiDAR HD acquisitions are required to capture the entirety of a tile. Since each 1 km^2 tile is covered by multiple overlapping flight lines, every scene is observed from at least two different viewpoints. Those flights are identified by distinct flight IDs, as illustrated in Figure 5.4. By decomposing the tiles into smaller sub-scenes, we can create multiple point clouds of the same area of interest (e.g, a specific crossroad, building), revisited from different viewing angles based on their flight IDs. These overlapping observations serve as an exact source of revisits to a scene under different conditions. In contrast to mobile-mapping datasets, where revisits are defined based on sequential acquisition involving temporal information, our approach defines revisits purely in terms of spatial overlap.



(a) View from the flight ID f (b) View from the flight ID g (c) Overlaps of different flight ID scenes

Figure 5.4: Visualization of point clouds of the same location from LiDAR HD tiles, captured from different viewing angles. (a) First observation of a scene from the flight ID f. (b) Second observation of a scene from the flight ID g, with $g \neq f$. (c) Overlap of the two observations, where each viewpoint contributes complementary information, such as recovering missing façade points.

Once every point cloud has been constructed, each cell contains between two and three acquisitions recorded from different flight passages, as illustrated in Figure 5.5a. To enforce viewpoint invariance in the new descriptor, we select the farthest angular viewpoint whenever more than two acquisitions are available as training positive pairs (Figure 5.5b).



(a) Number of revisits per cell.

(b) Observation angles from different flight IDs.

Figure 5.5: (a) Heatmap showing the number of revisits per cell in the $Paris_{East}$ area (grayscale). (b) Observation angles per cell from different flight IDs in the same area, with colors indicating the angle in degrees.

We also compare the main characteristics of these datasets $Paris_{East}$ and $Paris_{West}$, to state-of-the-art place recognition benchmarks. LoGG3D-Net was originally validated on KITTI (Geiger et al., 2013) and MulRAN (Kim et al., 2020), both collected from instrumented vehicles and organized into sequences with revisit scenarios. In particular, LoGG3D-Net was trained and evaluated on the DCC, Riverside, and KAIST sequences of MulRAN, but not on the Sejong sequences (83k scenes) of this dataset. Closer to our aerial LiDAR setting, CrossLoc3D (Guan et al., 2023) evaluates multiview retrieval on the Oxford RobotCar dataset (Maddern et al., 2017), which consists solely of ground-based acquisitions, and on CS-Campus3D, a hybrid benchmark that combines aerial LiDAR from state mapping agencies with ground-based robot acquisitions over the same area. Finally, Stereopolis II (Paparoditis et al., 2012) is a ground-level dataset of Paris streets.

Table 5.1: Overview of 3D place recognition dataset coverage. Num_{scenes} refers to the total number of point clouds in a dataset.

Dataset	acquisition	area (km^2)	Driving distance (km)	num_{scenes}	environment
LiDAR HD ($Paris_{East}$)	Aerial	9	/	50k	urban+forest
LiDAR HD ($Paris_{West}$)	Aerial	16	/	90k	urban+forest
KITTI (Geiger et al., 2013)	Terrestrial	/	1.7	20k	urban
MulRAN (Kim et al., 2020)	Terrestrial	/	40	150k	urban
Oxford Robotcar (Maddern et al., 2017)	Terrestrial	/	1000	/	urban
CS-Campus3D (Guan et al., 2023)	Terrestrial	5.5	/	7k	urban
CS-Campus3D (Guan et al., 2023)	Aerial	5.5	/	27k	urban
ALITA (urban) (Yin et al., 2022)	Terrestrial	/	120	50 vehicle trajectories	urban
ALITA (campus) (Yin et al., 2022)	Terrestrial	/	50	8x10 vehicle trajectories	campus
Stereopolis II (Paparoditis et al., 2012)	Terrestrial	/	120	/	urban

From Table 5.1, we observe that our initial selection of $Paris_{East}$ and $Paris_{West}$ tiles already provides coverage larger than CS-Campus3D, while remaining comparable in scale to widely used benchmarks such as KITTI and MulRAN. Importantly, the LiDAR HD dataset can be easily scaled to larger regions by incorporating additional tiles from the national acquisition program, offering a clear path toward building continental-scale benchmarks. While the Oxford RobotCar dataset certainly contains a much higher number of

individual scans, these primarily originate from repeated traversals of the same routes, which are designed to capture seasonal and illumination changes. In contrast, our dataset grows by expanding into new geographic areas, thereby increasing not only the number of scenes but also the diversity of environments, which is crucial for assessing scalability in place recognition.

5.3 LoGG3D-Net applied to LiDAR HD

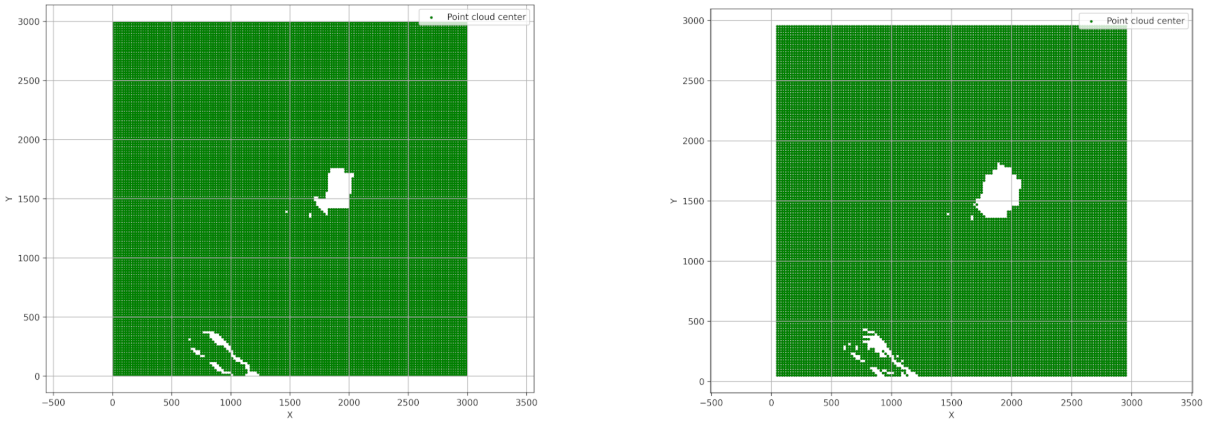
In the previous section, we described the LiDAR HD dataset and its adaptation into a large-scale place recognition benchmark. In section 5.3.1, we present our strategy for using LoGG3D-Net as a point cloud encoder for LiDAR HD data. We begin by detailing the pre-processing and curation of training samples. Then, we evaluate the original LoGG3D-Net descriptor, pre-trained on KITTI to observe the performance drop caused by differences in density, coverage, and acquisition modality. In section 5.3.2, with the objective of applying DSI-3D to aerial-based point clouds, we detail the fine-tuning procedure of the encoder and evaluate it in Section 5.3.3 on independent regions of the Paris area. Finally, we investigate the effect of rotation augmentation during training in Section 5.3.4.

5.3.1 Evaluation of the pre-trained LoGG3D-Net model

In this section, the pre-trained LoGG3D-Net model on KITTI is evaluated on $Paris_{East}$, to determine its generalization capability to new modalities without additional training. For this, we introduce an additional preprocessing step applied to the dataset to remove undesirable samples and obtain an optimal setup for evaluation.

Filtering incomplete point clouds

A filtering step is applied to discard point clouds whose spatial extent, in either the X or Y dimension, is smaller than 98 meters. This step removes edge cases where scenes lack access to neighboring tiles, as well as limiting cases such as water bodies that produce sparse or incomplete point clouds. After this filtering, 46881 and 85731 point clouds are conserved for $Paris_{East}$ and $Paris_{West}$, respectively. Figure 5.6 shows the effect of this filtering on $Paris_{East}$.



(a) Full point cloud dataset (54863 scenes)

(b) Filtered point cloud dataset (46881 scenes)

Figure 5.6: Point cloud center repartition before (a) and after (b) the filtering of incomplete scenes in $Paris_{East}$.

Pre-trained model evaluation

The first experiment aims to assess the retrieval capability of a LoGG3D-Net model pre-trained on ground-based scans. For this, we select the model SO, i.e, the model trained on all the KITTI sequences except sequence 00, as presented in Section 4.2 of Chapter 3. Table 5.2 presents the retrieval performance of an exhaustive search, according to the $F1_{max}$ and $Hits@N$ metrics. The positive and negative thresholds were set arbitrarily to 23 meters, in order to consider only adjacent neighbors (whose geometric distance to the ground truth is equal to 20 m).

Table 5.2: Evaluation on LiDAR HD of the LoGG3D-Net model pre-trained on KITTI.

Pre-trained model	**S0**
Voxel size (m)	0.1
Descriptor size	256
Maximum number of points	35000
Positive threshold	23
Negative threshold	23
$F1_{max}$	0.1352
$Hits@1$	0.0222

These results show that a model trained on ground-based scans generalizes poorly to aerial LiDAR retrieval, as both scores collapsed dramatically. This can be explained by the difference in acquisition modality, resulting in varying point density and coverage of scenes. In order to apply DSI-3D to $Paris_{East}$ and $Paris_{West}$, we first need to fine-tune the point cloud encoder.

5.3.2 Fine-tuning LoGG3D-Net on LiDAR HD

Experiments from the previous section demonstrated that reusing the LoGG3D-Net model pre-trained on KITTI results in poor retrieval performance on LiDAR HD data, most likely due to the modality gap in terms of density and coverage. In this section, to adapt DSI-3D to this new modality, we begin by splitting the $Paris_{East}$, containing nine 1 km^2 LiDAR tiles arranged in a 3×3 grid into two sets, for training and evaluation. Then we fine-tuning LoGG3D-Net using different configurations.

Dataset distribution

We could have fine-tuned LoGG3D-Net on every point cloud of the dataset to obtain highly discriminative representations. While this would demonstrate that DSI-3D can correctly learn docid assignments from descriptors, it would come at the cost of generalizability, limiting the transfer of the fine-tuning to unseen areas of Paris.

Instead, the dataset split utilized is illustrated in Figure 5.7, where the green column is used for training and the remaining tiles in blue serve as the evaluation set. A buffer of two point cloud columns (40 meters wide) was excluded from the training set to ensure independence between the training and evaluation datasets. Since LoGG3D-Net requires quadruplets for training, each point cloud was paired with a positive example defined as the scene sharing the same center coordinates but captured from the viewpoint with the largest angular difference. Negative point clouds were defined as those located more than 59 meters away from the query, ensuring the absence of overlap and preventing negatives from sharing parts of the query scene.

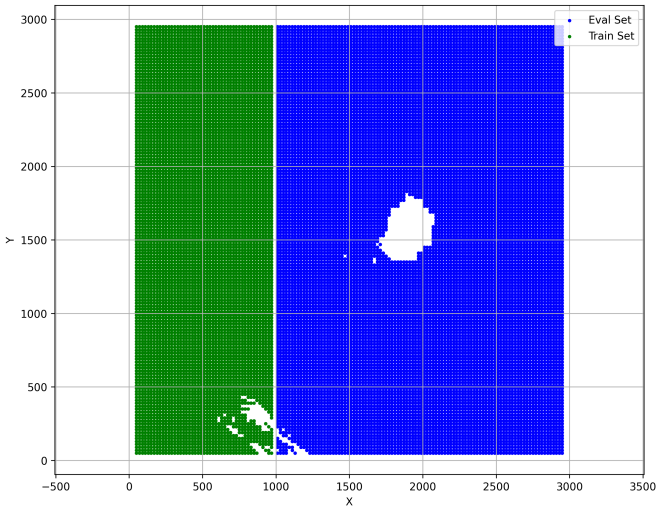
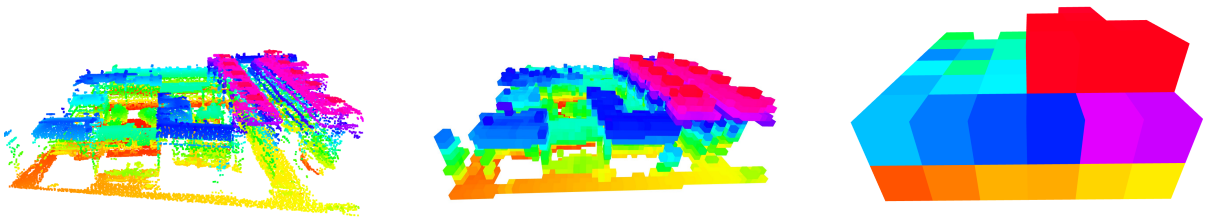


Figure 5.7: Visualization of the LiDAR HD dataset split for $Paris_{East}$, with the training set, containing three tiles (green), and the evaluation set (blue), containing six tiles.

Point cloud preprocessing

Fine-tuning LoGG3D-Net on LiDAR HD data requires several adaptations of its hyperparameters. One key parameter is the preprocessing of point clouds, specifically the number of points considered per scene. By default, LoGG3D-Net retains 35000 points randomly sampled from each point cloud, whereas our LiDAR HD dataset contains, on average, 120000 points per scene. Empirically, we found that increasing the limit to 200000 points in order to preserve all available points yielded the best results. We trained each model for approximately two days.

Another important parameter is the voxel size used by the U-Net backbone, which is set to 10 cm by default. We explore three different voxelization scales for two reasons: (i) to evaluate whether one scale is more discriminative on our dataset, and (ii) to assess whether combining multiple scales by concatenating their representations into a 768-dimensional vector, similar to the feature pyramid in GD-MAE (Yang et al., 2023), could improve performance. Figure 5.8 illustrates the effect of voxel size on point cloud representation.



(a) Original point cloud

(b) Voxelized point cloud (2 m)

(c) Voxelized point cloud (20 m)

Figure 5.8: Visualization of a point cloud under different voxelization settings. (a) Original point cloud without voxelization, (b) voxelized with a 2-meter resolution, and (c) voxelized with a 20-meter resolution.

5.3.3 Evaluation of the fine-tuning

To evaluate the hyperparameter adaptation of LoGG3D-Net on the LiDAR HD dataset, we reuse the metrics presented in Section 4.3.3 of Chapter 3, namely $Hits@N$ and $F1_{max}$, using exhaustive search.

The positive threshold is extended to 23 meters to include direct neighbors, strictly 20 meters apart with 80% overlap, while remaining below the distance to the first diagonal cells (28 meters). Although point clouds within a 56-meter radius of the ground truth still share overlapping geometry, we adopt a stricter evaluation by setting the negative threshold equal to the positive threshold. This threshold is applied consistently to both $Hits@N$ and $F1_{max}$ metrics. During the evaluation, every point cloud descriptor is included in the reference database. However, for each query, the corresponding descriptor is removed to ensure that the model retrieves the best match other than itself across all the scenes.

Table 5.3 summarizes the performance of each training setup. Training A uses the default LoGG3D-Net parameters for the voxel size and the maximum number of points. We observe that using all points with a voxel size of 2 meters yields the best results. This finding is consistent with CrossLoc3D (Guan et al., 2023), which adopts a similar voxelization size of 3 meters.

Table 5.3: Fine-tuned training hyperparameter evaluations of LoGG3D-Net on LiDAR HD. The first row presents the training configurations, the second row the distance threshold used in the evaluation and the last row presents the metrics scores. The best training configuration is in bold.

Training	**A**	**B**	**C**	**D**	**B+C+D**
Voxel size (m)	0.1	0.1	**2**	20	0.1+2+20
Descriptor size	256	256	**256**	256	768
Maximum number of points	35000	200000	**200000**	200000	200000
Data augmentations	Disabled	Disabled	**Disabled**	Disabled	Disabled
Positive threshold	23	23	**23**	23	23
Negative threshold	23	23	**23**	23	23
$F1_{max}$	0.8245	0.8289	**0.9915**	0.9326	0.9545
$Hits@N$	0.6836	0.5497	**0.9829**	0.8737	0.9128

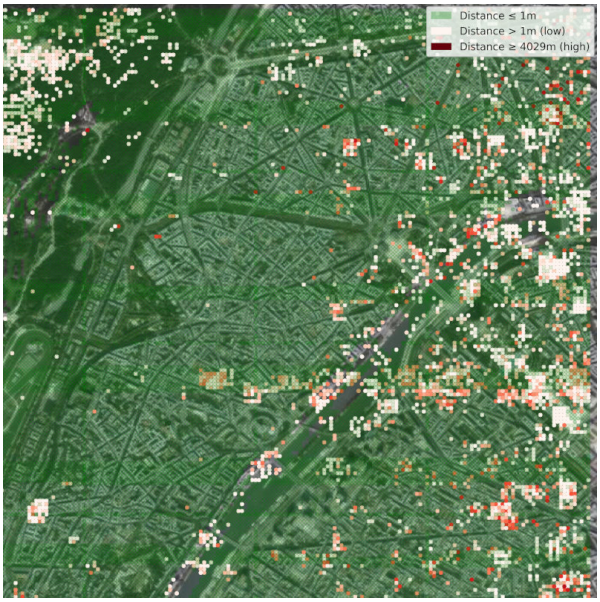
The last column, labeled Training B+C+D, corresponds to the construction of a 768-dimensional descriptor obtained by concatenating the feature vectors from models B, C, and D in that order. Although this multi-scale descriptor achieves better retrieval performance than the individual models trained with voxel sizes of 0.1 m and 20 m, Training C alone surpasses both the mono-scale and the multi-scale configurations. This indicates that the improvements observed with the concatenated descriptor mainly stem from the contribution of Training C, while the additional information from Training B and Training D does not further enhance retrieval performance. Consequently, for the remainder of this thesis, we use the descriptors produced by LoGG3D-Net from Training C.

Table 5.4 shows the evaluation of Training C with an even stricter threshold of 1 m, instead of our previous standard distance thresholds set at 23 m, to measure how often the model retrieves exact revisits, i.e., the same scene observed from a different flight ID. The scores decreased only marginally, indicating that most of the matches were already optimal.

Table 5.4: Fine-tuned training hyperparameter C, evaluated on standard and strict distance thresholds, on LiDAR HD. The first row presents the training configurations, the second row the distance threshold used in the evaluation and the last row presents the metrics scores.

Training	C	C
Voxel size (m)	2	2
Descriptor size	256	256
Maximum number of points	200000	200000
Data augmentations	Disabled	Disabled
Positive threshold	23	1
Negative threshold	23	1
$F1_{max}$	0.9915	0.9909
$Hits@N$	0.9829	0.9816

Figure 5.9 shows the spatial distribution of errors for the selected model for several flights. Each dot corresponds to the center coordinates of a query. Green dots indicate cases where the top-1 candidate is an exact revisit, while the red gradient encodes increasing localization error. White points represent errors of less than one meter, whereas darker shades of red indicate errors of up to three kilometers. We observe that many errors are concentrated within the Bois de Vincennes.



(a) Error repartition on $Paris_{West}$



(b) Error repartition on $Paris_{East}$



(c) Error repartition on $Paris_{West}$ (zoomed)



(d) Error repartition on $Paris_{East}$ (zoomed)

Figure 5.9: Retrieval error repartition of exhaustive search, with descriptors obtained from the model using training C, on $Paris_{West}$ (a) and East (b). Green is exact correspodence (distance = 0 m), while the gradient of red shows the distance error, from white (small distance) to dark red (high distance). (c) shows the errors repartition near Champs-de-Mars while (d) shows the errors repartition near quartier Carnot, in Bois de Vincennes.

While Figure 5.9 may be affected by overlapping points, since multiple queries can share the same pose, this phenomenon is also confirmed when plotting the error distributions separately for each flight ID in Figure 5.10.

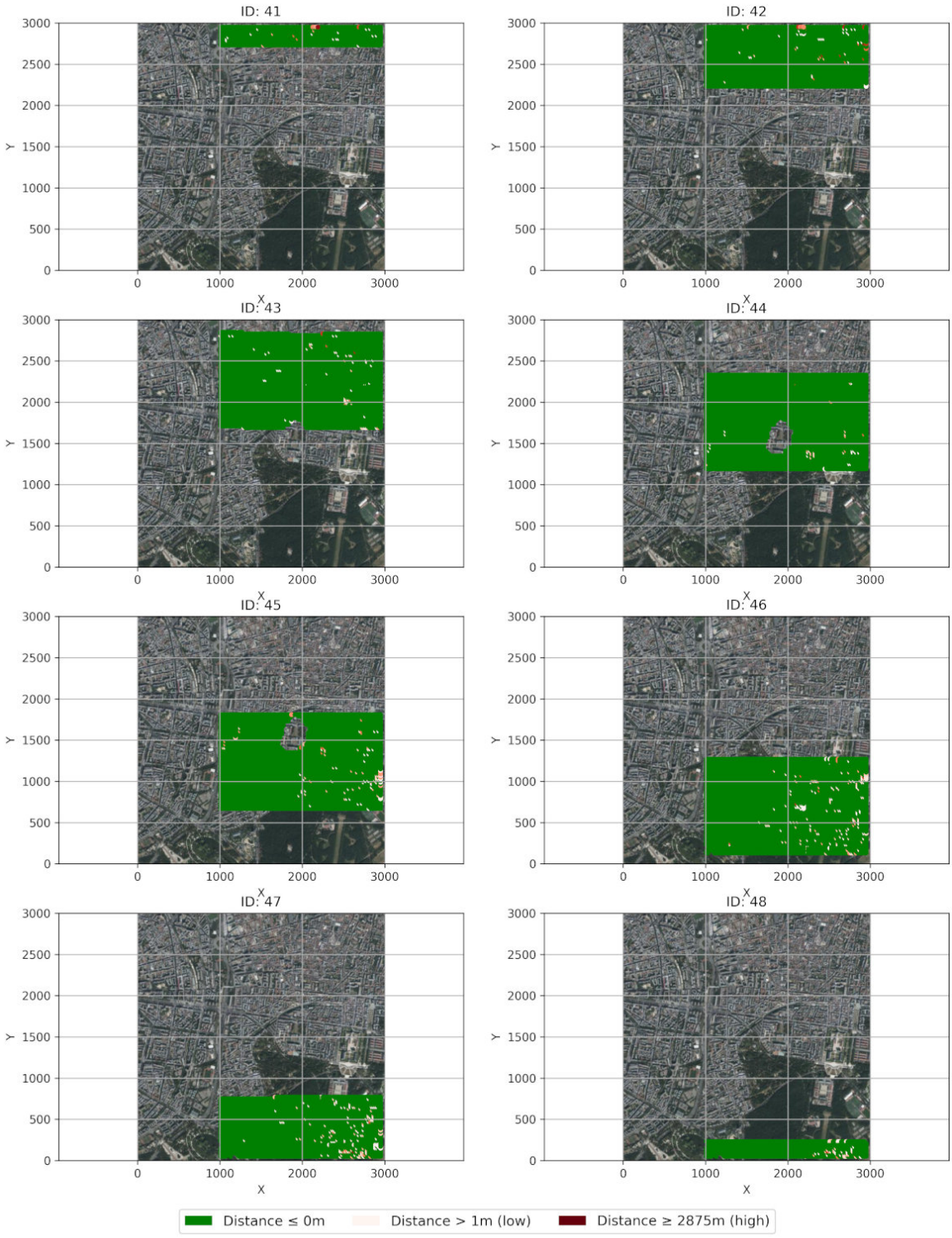


Figure 5.10: Retrieval error repartition of exhaustive search, per flight ID, on $Paris_{East}$. Green are exact correspondence (distance = 0 m), while the gradient of red show the distance error, from white (small distance) to dark red (high distance).

In order to verify the generalization capacity of this fine-tuning, we evaluate it in a different area of Paris, namely $Paris_{West}$ (Figure 5.2a), which includes buildings, forested areas, and water bodies. Since none of its point clouds were used during training, the entire area can serve as a basis for evaluation. The evaluation results are reported in Table 5.5.

Table 5.5: Performance of Training C (fine-tuned LoGG3D-Net) evaluated on the $Paris_{West}$ area. The first row presents the training configurations, the second row the distance threshold used in the evaluation and the last row presents the metrics scores.

Training	C
Voxel size (m)	2
Maximum number of points	200000
Data augmentations	Disabled
Positive threshold	1
Negative threshold	1
$F1_{max}$	0.9616
$Hits@N$	0.9230

We observe that Training C maintains high performance in this larger, unseen area, indicating that the fine-tuning strategy generalizes beyond the training region and could potentially be extended to other parts of the Paris area. In Figure 5.9, we observe that the errors are concentrated in forested areas and near the Seine River, with a higher error density on the eastern side of the city.

In Figure 5.11, we show the error distribution without overlap. A line of errors appears in flight ID 42, up to the Seine, where point clouds from this acquisition are poorly retrieved in IDs 41 and 43. This issue may stem from the training samples from $Paris_{East}$ being insufficiently representative of the specific area surrounding the Eiffel Tower. Nevertheless, the overall results remain strong enough to validate this fine-tuning as a reliable 3D encoder.

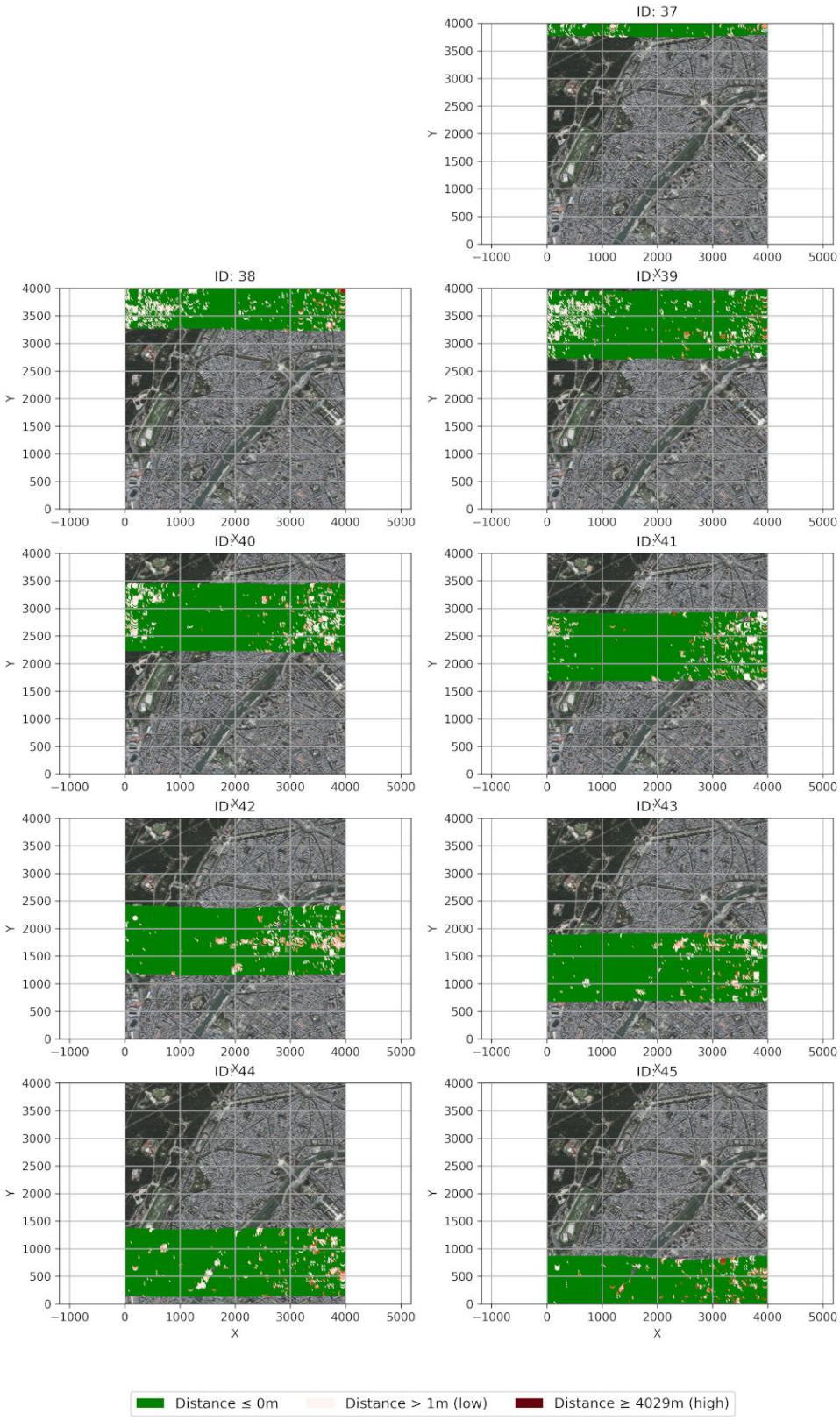


Figure 5.11: Retrieval error repartition for exhaustive search, per flight ID, on $Paris_{West}$. Green are exact correspondence (distance = 0 m), while the gradient of red show the distance error, from white (small distance) to dark red (high distance).

5.3.4 Training data augmentation

In Chapter 3 and 4, we use the LoGG3D-Net model pre-trained on KITTI, which is inherently robust to rotations around the Z-axis because it is trained on augmented data adapted to this transformation (as demonstrated in Appendix A). In this thesis, all the previous experiments have relied on non-augmented data since our objective was to evaluate the efficiency of generative retrieval rather than absolute descriptor robustness. However, we propose to complete the experiments by training a DSI-3D model on such augmented data, considering an additional rotational perturbation in order to find the optimal configuration for the LiDAR HD dataset and estimate the contribution of data augmentation during training on this dataset.

To this end, we introduce random rotations into the training configuration C, following the original LoGG3D-Net data augmentation strategy by applying a random rotation $\theta \in [0, 2\pi]$ along the Z-axis. This new configuration, denoted as C_{bis}, is trained under identical conditions, except for the additional rotational augmentation. Table 5.6 compares the performance of the original configuration C and the rotation-augmented configuration C_{bis}.

Table 5.6: Performance of Training C (fine-tuned LoGG3D-Net), with rotations, evaluated on the $Paris_{East}$ area. The first row presents the training configurations, the second row the distance threshold used in the evaluation and the last row presents the metrics scores. The best training configuration is in bold.

Training	C	C_{bis}
Voxel size (m)	2	2
Maximum number of points	200000	**200000**
Data augmentations	Disabled	**Rotation**
Positive threshold	1	**1**
Negative threshold	1	**1**
$F1_{max}$	0.9909	**0.9970**
$Hits@N$	0.9816	**0.9936**

A clear improvement is observed, as incorporating rotation significantly enhances the model's generalization capability, enabling it to better transfer the learned representations from the three training tiles to the six unseen ones. Although in the following Chapter 6 we use the descriptors obtained from configuration C for both DSI-3D and k-nearest neighbor retrieval methods, we acknowledge that descriptors from configuration C_{bis} are more robust and discriminative.

5.4 Conclusion

In this chapter, we introduced the LiDAR HD dataset for place recognition tasks. We first detailed its main characteristics and data format, then focused on two study areas of 16 km^2 and 9 km^2, which include diverse environments such as urban buildings, forests, and water bodies. The raw tiles were subdivided into a fine, regular grid (20×20 m) and aggregated into overlapping point clouds of 100×100 m to ensure sufficient spatial overlap for revisits. Point clouds located near tile and water surfaces were filtered to retain only scenes covering 98 by 98 meters. We then positioned our dataset relative to existing benchmarks, highlighting its scalability and suitability for large-scale place recognition.

We subsequently motivated the fine-tuning of LoGG3D-Net on this dataset, as the model pre-trained on KITTI does not generalize to LiDAR HD. This new dataset required

a new training protocol, with spatial separation of the training and evaluation sets, as well as a strategy to define the validation set and build the quadruplet for each query based on the multiple acquisitions of the same places, facilitated by the overlapping flight acquisition.

The evaluation demonstrated that the descriptors produced from the fine-tuned model achieved strong retrieval performance on $Paris_{East}$ using exhaustive search and generalized well to $Paris_{West}$, confirming their robustness in unseen areas.

Based on these results, we conclude that LoGG3D-Net, once fine-tuned, constitutes a strong 3D encoder for LiDAR HD data and provides a valid backbone for training DSI-3D in the subsequent chapter.

Chapter 6

Extending DSI-3D

Contents of this Chapter

6.1 Introduction 93
6.2 Mixture of Experts 93
6.2.1 Overview of current methods 94
6.2.2 Adapting MoE for DSI-3D 96
6.3 Re-ranking 106
6.3.1 Related work 106
6.3.2 Re-ranking on LiDAR HD 110
6.4 Conclusion 112

6.1 Introduction

In Chapter 5, we introduced the French LiDAR HD dataset and described how it was adapted for 3D point cloud retrieval. Building on these foundations, the present chapter focuses on extending the DSI-3D framework from Chapter 3, both in terms of scalability to larger land coverage and robustness of the retrieval results.

To this end, we explore two complementary directions. Section 6.2 investigates the use of a Mixture-of-Experts strategy, in which multiple DSI-3D models act as specialized experts that collaborate to retrieve point clouds. Section 6.3 then examines post-processing strategies, such as re-ranking, aimed at refining the similarity ranking and further improving Top-1 accuracy. Together, these approaches contribute to making DSI-3D more adaptable and reliable in large-scale geospatial scenarios.

6.2 Mixture of Experts

A natural continuation of training DSI-3D on KITTI is to extend it to a larger dataset, namely LiDAR HD. While directly training a single model on this larger dataset is feasible, it inevitably leads to a substantial increase in training time. Although the Scaling laws for neural language models (Kaplan et al., 2020; Tay et al., 2021) for Transformers suggest consistent performance gains with increased model capacity, computational resources remain constrained by a fixed number of GPUs and time. Instead, we look for mathematical induction, demonstrating that if two models are trained on separate regions and a routing function correctly assigns each query to its corresponding model, the retrieval coverage can effectively be doubled for a computational cost equivalent to a single model inference plus the routing operation. In parallel, Mixture-of-Experts (MoE)

architectures (Jacobs et al., 1991; Jordan and Jacobs, 1994) are successful in scaling transformer models (Shazeer et al., 2017; Fedus et al., 2022). MoE introduces multiple specialized expert networks within the feedforward layers of the transformer, activating only a subset of them based on a learned gating function. While these architectures are predominantly applied to tasks such as multilingual translation, no prior work explores their use in point cloud retrieval. We propose a MoE-inspired framework, where each expert corresponds to a region-specific DSI-3D model, and a gating network determines which experts to activate based on the input query. Section 6.2.1 provides an overview of MoE methods and the motivations for adopting such an approach in this context. Then, Section 6.2.2 details the training of DSI-3D on LiDAR data using region-specific models, with the objective of enabling a Mixture-of-Experts strategy. Finally, we discuss the improvements introduced during training to address the challenges posed by a more complex dataset: unlike KITTI, which is mostly linear, the French LiDAR HD dataset is fully two-dimensional, as it is not linked to a road path, leading to denser index distributions.

6.2.1 Overview of current methods

The development of foundation models demonstrates that increasingly large architectures can achieve impressive performance across a wide range of tasks (Devlin et al., 2019; Radford et al., 2021; Achiam et al., 2023). However, such models are computationally expensive to train and deploy, and integrating heterogeneous datasets into a single model remains a significant challenge. To address these challenges, MoE architectures (Jacobs et al., 1991; Jordan and Jacobs, 1994; Cai et al., 2025) are presented as an efficient alternative. Unlike conventional models, which activate the entire parameter set for each input, MoE selectively routes the input to a subset of experts deemed most relevant. This allows individual experts to specialize in different domains, thereby improving efficiency and facilitating training on diverse datasets. As a result, MoE provides a scalable framework that is well-suited for handling multifaceted data distributions.

These models become a central component in scaling transformer-based large language models (LLMs), as demonstrated in architectures such as Switch Transformer (Fedus et al., 2022) and GShard (Lepikhin et al., 2020). An MoE layer consists of a set of experts $\{E_1, \ldots, E_N\}$ and a gating network G. In practice, each expert is typically implemented as a feed-forward network (FFN) within a transformer block, while the gating network determines which subset of experts should be activated for a given input. The gating network is usually a linear network followed by a softmax function (Shazeer et al., 2017; Fedus et al., 2022). This design allows the model to dynamically route inputs to the most relevant experts, thereby increasing model capacity without incurring proportional computational costs. Depending on how the gating function selects experts, MoE architectures can be categorized into dense and sparse variants:

Dense MoE

In the dense setting, for an input x, all experts are activated, and their outputs $\{E_1, \ldots, E_N\}$ are combined according to the gating weights $G(x)_i$ (Jordan and Jacobs, 1994):

$$y = \sum_{i=1}^{n} G(x)_i E_i(x) \tag{6.1}$$

where

$$G(x)_i = Softmax(g(x)_i) = Softmax(x.W_g) \tag{6.2}$$

with W_g a trainable weight matrix and $g(x)_i$ as the raw output of the gating function for the expert i.

Sparse MoE

In contrast, sparse MoE activates only a subset of k experts, thereby reducing the effective computational cost (Shazeer et al., 2017). A common formulation adapts Equation 6.2 by selecting the top-k experts according to their gating scores:

$$G(x)_i = Softmax(KeepTopk(H(x), k)) \tag{6.3}$$

where $H(x)$ denotes the noisy gate values. To encourage load balancing across experts, Gaussian noise is added to the gating logits via a trainable projection W_{noise}:

$$H(x)_i = (x.W_g)_i + Gaussian_noise((x.W_{noise})_i) \tag{6.4}$$

Only the top-k values are retained, while the others are set to $-\infty$, ensuring that the corresponding gates $G(x)_i$ become zero:

$$KeepTopK(v, k)_i = \begin{cases} H(x)_i & \text{if } H(x)_i \text{ is in the top k elements of H(x).} \\ -\infty, & \text{otherwise} \end{cases} \tag{6.5}$$

Soft MoE

A limitation of sparse expert activation is that the gate must assign each input discretely to a single expert, making this architecture sensitive to outliers and training instabilities. To address this issue, Soft MoE introduces an intermediate solution in which all input tokens are processed by every expert, similarly to Dense MoE, but are reweighted before being passed through each expert (Puigcerver et al., 2023). As a result, each expert receives a different weighted combination of input tokens, leading to a *soft* or implicit activation rather than a discrete selection. Other approaches, such as Soft Merging of Experts with Adaptive Routing (SMEAR) (Muqeeth et al., 2023), merge expert parameters instead of input tokens by averaging the weights of all experts. This strategy limits computational overhead compared to a Dense MoE.

Auxiliary Loss

As presented in (Shazeer et al., 2017), gating networks in sparse MoE architectures can suffer from load imbalance, where only a small subset of experts receives most of the training samples. This imbalance is problematic, as it prevents many experts from being properly trained and reduces the overall model's capacity and generalization.

To mitigate this issue, an auxiliary loss term L_{Aux} is introduced to encourage a more uniform distribution of queries across experts. The total loss is thus defined as:

$$L_{Total} = L_{main} + \alpha \cdot L_{Aux} \tag{6.6}$$

where L_{main} denotes the main task loss (e.g., the retrieval loss defined in Equation 4.2), and α is a scaling factor that controls the contribution of the auxiliary term, typically set to 0.01.

A common formulation of the auxiliary loss (Shazeer et al., 2017) is based on the coefficient of variation (CV) of the experts' importance values, expressed as:

$$L_{Aux} = CV(Importance(X))^2 \tag{6.7}$$

$$Importance(X) = \sum_{x \in X} G(x) \tag{6.8}$$

where $G(x)$ denotes the gating score assigned to each expert for a given input x. A high CV indicates a strong imbalance among experts, so minimizing L_{Aux} encourages the gating network to distribute queries more evenly.

6.2.2 Adapting MoE for DSI-3D

This section proposes to apply a MoE architecture to our DSI-3D model trained on KITTI in Chapter 4. However, the point cloud repartition from $Paris_{East}$ and $Paris_{West}$ (Section 5.2.1, Chapter 5), being a 2D grid, is vastly different from a mobile mapping dataset such as KITTI (Section 4.2, Chapter 4), which is mostly linear. This denser repartition increases the number of point clouds having close point cloud docids.

For this adaptation, we first define our use of DSI-3D as expert model in Section 6.2.2.1. From this definition of experts, Section 6.2.2.2 motivates our choice of gate strategy and how it impacts expert requirements. To train DSI-3D on a denser dataset, Section 6.2.2.3 explores varying spatial indexing resolutions, directly affecting docid representations. Section 6.2.2.4 presents the training of DSI-3D, following the same principles as Chapter 4. The gate training is discussed in Section 6.2.2.5 and Section 6.2.2.6 evaluates the MoE implementation of DSI-3D facing classical retrieval approaches as well as *global* DSI-3D models, trained on $Paris_{East}$ and $Paris_{West}$.

6.2.2.1 Experts definition

As presented earlier, a Mixture of Experts model requires two components: a gating strategy and a set of experts. The assignment of tasks to experts can be achieved in several ways. Experts may specialize in specific types or aspects of data, such as activation based on point cloud descriptors or their acquisition modalities. In the context of DSI-3D experts, this could translate to experts dedicated to retrieve a specific environment (i.e., roads, urban areas, forests). Task assignment can also be area-specific. For example, each expert can correspond to a local DSI-3D model trained on a specific region. In our setting, we choose to consider each expert as a DSI-3D model trained on a specific region. This design allows greater flexibility for independently training and updating experts, and it also allows for the easy extension of the search area.

6.2.2.2 Gating strategy

Since each expert corresponds to a local DSI-3D model trained on a specific geographic region, experts associated with distant or non-overlapping areas are unlikely to contribute effectively to the generative retrieval of a given point cloud. Only those trained in neighboring or overlapping regions are expected to provide relevant information. Therefore, a sparse activation strategy, in which only a small subset of experts is activated for each query, appears more realistic. Nevertheless, dense activation is also evaluated, in order to assess the MoE robustness to expert noise. Soft MoE could also be a promising direction. However, in our case, the experts are not lightweight FFN but full T5 models, each already trained to generate a docid from a complete point cloud descriptor rather than from individual descriptor tokens. This architectural difference makes a direct adaptation of Soft MoE approaches less straightforward.

Since we choose a sparse gating strategies on pre-trained DSI-3D to specific area, this implies that three important requirements need to be considered:

- The gating network receives only the query representation as its input. It is therefore more efficient if all experts and gates are trained on a shared 3D encoder. However, on the previous dataset (KITTI), each DSI-3D model is trained on a sequence with its own LoGG3D-Net pre-trained model. As a result, they do not share a common representation space, leaving the gating network with no consistent clues about which pre-trained model to select as input. To address this, all experts must operate in the same representation space by strictly using the same 3D encoder.

- All experts have to share the same output space in order to combine their outputs meaningfully. This ensures that their outputs are comparable, allowing the gating mechanism to combine them meaningfully. In other words, each expert must share not only the encoder but also the same indexing reference.
- Finally, each expert has to receive approximately the same number of queries to maintain load balance during training. An imbalance results in only a subset of experts being frequently activated and effectively trained, undermining the purpose of the Mixture-of-Experts approach. In our case, the experts cover areas with nearly equal data representation (differences below 1%), thus avoiding load imbalance issues due to dataset distribution.

To satisfy the requirements outlined above, all experts are constrained to share the same 3D encoder, here the LoGG3D-Net model fine-tuned on LiDAR HD (see Chapter 5), as well as the same output space. This ensures that their representations are comparable and that the gating network can combine their outputs consistently.

Figure 6.1 illustrates the proposed MoE pipeline for retrieval in 3D point clouds. The gating network evaluates the query representation and sparsely activates the experts with the highest gate scores. The outputs of the selected experts are then weighted according to their respective gating probabilities and combined to produce the final prediction. This mechanism allows the model to leverage region-specific expertise while maintaining computational efficiency through selective activation.

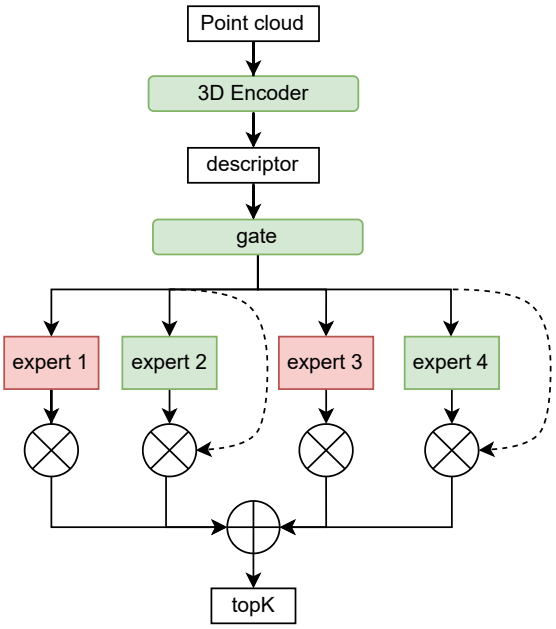


Figure 6.1: Overview of the proposed MoE architecture for retrieval in 3D point clouds. The gating network selects and weights a subset of experts (in green) based on the query embedding. Experts in red are not activated.

6.2.2.3 Spatial indexing resolution

The positionally structured string identifiers derived from a Hilbert curve map a 2D position associated with the point cloud to a distance used as a docid. The 2D position considered for this mapping can take different forms: it may correspond to the metric

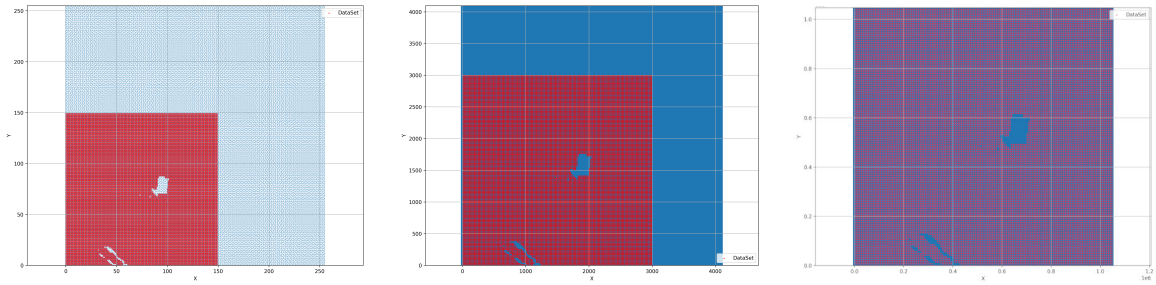
coordinates of the point cloud center, discretized cell coordinates, or extended coordinates scaled to fully exploit the Hilbert curve coverage.

For the dataset $Paris_{East}$ (see Section 5.2.1, Chapter 5), we define three levels of Hilbert-based indexing, corresponding to progressively finer spatial resolutions, as illustrated in Figure 6.2. These three levels form the output space of the docids used by DSI-3D experts.

Cell indexing. At the coarsest level (Figure 6.2a), point clouds are assigned to a 2D grid of size 150 x 150. Here, the Hilbert curve is applied to the corresponding *cell* coordinates of the point cloud within the dataset grid or *matrix*. The grid cell coordinates are encoded into docids using a Hilbert curve at iteration 8, capable of encoding coordinates in the range $[0, 256]$.

Coordinates indexing. At an intermediate level (Figure 6.2b), indexing is applied directly to the raw spatial coordinates of the point clouds, which range from 0 to 3000 m on each axis. Since all point-cloud coordinates are positive integers, there is no need to shift and rescale the dataset. This resolution requires a Hilbert curve at iteration 12 to encode coordinates in the range $[0, 4096]$.

Extended indexing. Finally, at the finest level (Figure 6.2c), we use a high-resolution indexing strategy that maximizes the diversity of the docid space. The point-cloud coordinates are interpolated to obtain a homogeneous sampling on a grid of size 1000000×1000000 before applying a Hilbert curve at iteration 20, providing the finest index granularity to encode coordinates in the range $[0, 1048576]$.



(a) Cell indexing (b) Coordinates indexing (c) Extended indexing

Figure 6.2: Three levels of point cloud indexing based on the Hilbert curve. (a) Cell indexing: indices are assigned to point clouds using their grid cell coordinates, requiring a Hilbert curve at iteration 8. (b) Coordinates indexing: indices are computed directly from the spatial coordinates of point clouds, as in Chapter 3, using a Hilbert curve at iteration 12. (c) Extended indexing: the same scheme as KITTI sequence 22 is reused, with point cloud coordinates interpolated for a homogeneous distribution of identifiers and a Hilbert curve at iteration 20.

In the next section, experts are trained on a subsample of $Paris_{East}$ and $Paris_{West}$. Using any of these three indexation levels on the entire dataset ensures that all experts share the same output space. It also guaranties a shared bijective mapping between docids and point clouds, which enables meaningful contributions from multiple experts. An advantage of indexation (b) is that it can be applied directly to both $Paris_{East}$ and $Paris_{West}$ without any preprocessing.

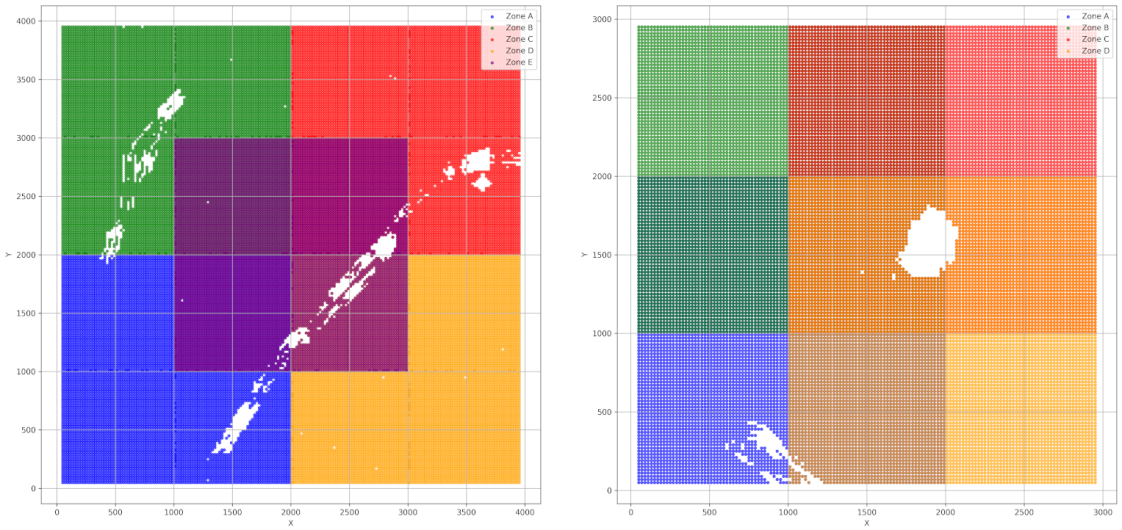
6.2.2.4 Expert training

This section describes the training protocol of local DSI-3D experts. We first present the splitting step of $Paris_{East}$ and $Paris_{West}$ into expert regions. Then, we explore the training hyperparameter configuration to fine-tune a local DSI-3D expert on LiDAR HD. Once the best configuration is identified, it is used for training all remaining experts.

Dataset distribution among the experts.

In Chapter 5, the dataset is split according to spatial separation (see Figure 5.7). To train DSI-3D experts, the training, validation, and evaluation sets are now separated by flight IDs rather than spatially, enabling each model to retrieve point clouds across the entire area. However, we retain the same grid-splitting principle, as presented in Figure 5.3.

Allowing overlap between their training regions further ensures that several experts can provide meaningful contributions to the same query. Figure 6.3 illustrates the spatial distribution of the expert regions, with five covering the $Paris_{West}$ (namely, A_{West} B_{West} C_{West}, D_{West}, E_{West}) and four covering the $Paris_{East}$ (namely, A_{East} B_{East} C_{East} and D_{East}).



(a) $Paris_{West}$ five experts distribution (b) $Paris_{East}$ four experts distribution

Figure 6.3: Distribution of the expert on the 2 areas of Paris. Experts are colored in blue, green, red, yellow (and magenta for $Paris_{West}$), respectively. The additional colors result from the additive synthesis of expert colors, representing areas of overlap.

The dataset division follows the same principle introduced in Section 5.3 from Chapter 5, where sets are separated by flight IDs. The total number of point clouds assigned to each expert, along with their distribution, is summarized in Table 6.1.

Table 6.1: Point cloud distribution per expert.

Area	$\mathbf{A}_{East}$	$\mathbf{B}_{East}$	$\mathbf{C}_{East}$	$\mathbf{D}_{East}$	$\mathbf{A}_{West}$	$\mathbf{B}_{West}$	$\mathbf{C}_{West}$	$\mathbf{D}_{West}$	$\mathbf{E}_{West}$
Training set	9222	9350	9308	9279	9435	9425	9419	9431	9736
Validation set	2118	2294	2279	2239	2595	2255	2436	2266	2539
Evaluation set	9222	9350	9308	9279	9435	9425	9419	9431	9736
Total	20562	20994	20895	20795	21467	21105	21274	21128	22011

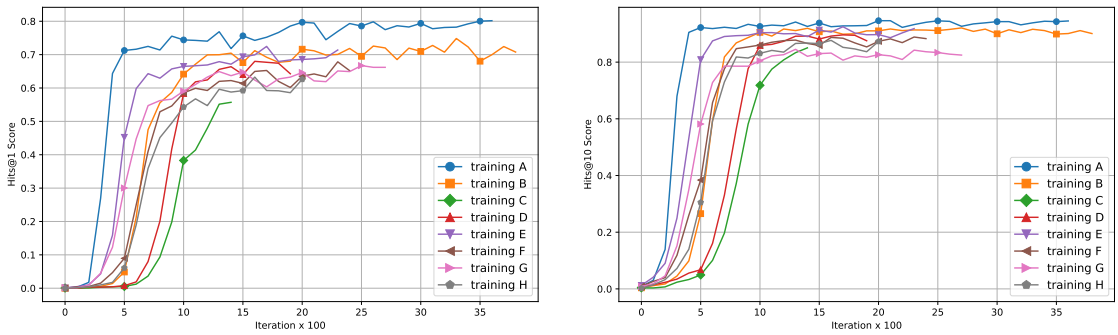
Training hyperparameter fine-tuning. With the dataset divided into three inde-

pendent sets and the 3D encoder already fine-tuned, we proceed to train the individual DSI-3D models that serve as experts. As a starting point, we adopt the best configuration identified in Section 4.5.2 of Chapter 4, named Training T_A; Table 6.2 presents variants of this training configuration. For the subsequent hyperparameter exploration, we restrict training to the A_{East} area in order to systematically evaluate the variations under the same conditions. In this Table, *point cloud indexing* refers to the level of Hilbert curve indexing, where *coord*, *extd*, and *cell* denote coordinate-based indexing, extended indexing, and cell-based indexing, respectively.

Table 6.2: Training hyperparameters of local expert A_{East} on LiDAR HD. Training T_A uses the best configuration found on KITTI (see Section 4.2.2 in Chapter 4), and we explore additional configurations from T_B to T_H.

Training	$\mathbf{T}_A$	$\mathbf{T}_B$	$\mathbf{T}_C$	$\mathbf{T}_D$	$\mathbf{T}_E$	$\mathbf{T}_F$	$\mathbf{T}_G$	$\mathbf{T}_H$
Training samples	9222	9222	9222	9222	9222	9222	9222	9222
Point cloud indexing	coord	coord	extd	coord	coord	cell	cell	coord
Hilbert curve iteration	12	12	20	12	12	8	8	12
Maximum training time	96h	48h	24h	20h	20h	20h	20h	20h
Batch size	1024	256	256	256	256	256	256	256
Learning rate (LR)	$5e-4$	$1e-5$	$1e-5$	$1e-4$	$5e-4$	$1e-5$	$5e-4$	$5e-4$
LR scheduler	cosine	cosine	linear	linear	cosine	linear	cosine	cosine
Warmup steps	100	100	100	100	100	100	100	100
Max epochs	10000	10000	10000	10000	10000	10000	10000	10000
Vocabulary size	939	939	939	939	939	939	939	939
Max token length	10	10	18	18	10	18	5	10
Index padding	no	no	no	no	no	no	no	yes

The validation scores, using *Hits*@1 metric, are computed every 100 training steps. They are showcased in Figure 6.4.



(a) Hits@1 validation score

(b) Hits@10 validation score

Figure 6.4: Validation scores during the training steps for different hyperparameters configuration of DSI-3D on LiDAR HD, measured by *Hits*@1 (a) and *Hits*@10 (b). The tokenizer used for docid encoding and decoding during the training protocol is the same as the one used for DSI-3D trained on KITTI.

A clear observation is that the configuration of training T_A remains the most efficient way to train DSI-3D on LiDAR HD. However, its convergence is slow and costly. Even with this setting, the *Hits*@1 performance of the models is 0.8017, far from the scores of LSH, which has a score of 0.9163, and far below exhaustive search, k-d tree, and HNSW, which reach 0.9930, 0.9930, and 0.9909, respectively.

Before exploring additional hyperparameter configurations, we verified the quality of our descriptors. Appendix B presents an evaluation of the distance ratio across all ex-

perts, confirming that the descriptors are not a bottleneck due to poor descriptiveness.

Expert training on Hilbert curve index vocabulary. Up to this point, we rely on the default tokenizer from GIT (git-base-coco), originally fine-tuned on the Common Objects in Context (COCO) dataset (Lin et al., 2014) to train DSI-3D experts. While this tokenizer is sufficient for training on the relatively linear distribution of point clouds in KITTI (see 4), it becomes a limiting factor when applied to a truly two-dimensional dataset where many point clouds share very similar indices.

To address this limitation, we train a new tokenizer that better reflects the frequency distribution of index digits. We choose to adopt the WordPiece tokenizer (see Section 2.5.3 in Chapter 2), as it closely mimics the token format used in the original GIT model.

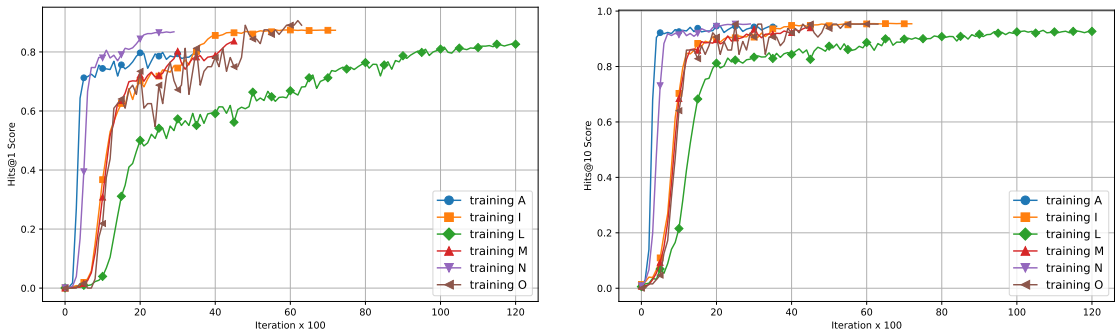
This tokenizer is trained on a new corpus of words, where the most frequent sub-words are assigned entries in the vocabulary, thereby defining the set of tokens. For this purpose, we simulate a 4×4 km with no empty cells and compute the Hilbert-curve indexes for all possible cells. We use this set of indexes to build a balanced corpus covering both LiDAR HD areas. This ensures that the tokenizer vocabulary aligns with the spatial distribution of our data, rather than with natural language artifacts from unrelated datasets.

The new training hyperparameters of local expert A_{East} are summarized in Table 6.3.

Table 6.3: Training hyperparameters of local expert A_{East} on LiDAR HD.

Training	$\mathbf{T}_I$	$\mathbf{T}_L$	$\mathbf{T}_M$	$\mathbf{T}_N$	$\mathbf{T}_O$
Area	A_{East}	A_{East}	A_{East}	A_{East}	A_{East}
Training samples	9222	9222	9222	9222	9222
Coordinate rescale	no	no	no	no	no
Hilbert curve iteration	12	12	12	12	12
Maximum training time	33h	60h	20h	20h	15h
Batch size	128	128	128	256	128
Learning rate (LR)	$5e-4$	$5e-4$	$5e-4$	$5e-4$	$5e-4$
LR scheduler	cosine	cosine	cosine	cosine	cosine
Warmup steps	100	100	100	100	100
Max epochs	100	200	100	100	100
Vocabulary size	1000	300	1000	1000	1000
Max token length	10	10	10	10	10
Index padding	yes	yes	yes	yes	yes

The validation scores of the models trained with the updated tokenizer, computed every 100 steps, are showcased in Figure 6.5.



(a) Hits@1 validation score

(b) Hits@10 validation score

Figure 6.5: Validation scores during the training steps for different hyperparameters configuration of DSI-3D on LiDAR HD, measured by *Hits*@1 (a) and *Hits*@10 (b).

From this study, the configurations T_I and T_O yield the most promising results while keeping the training time manageable. The evaluation of a local DSI-3D trained with the configurations T_O on area A_0 achieves a $Hits@1$ score of 0.8995 and an $F1_{max}$ score of 0.9658.Although longer fine-tuning or extended training might further improve performance, we adopt this configuration to ensure reproducibility across all experts and to focus on evaluating the effectiveness of the spatial MoE approach.

Each expert is trained using the configuration settings T_O and is evaluated based on the checkpoint with the highest validation scores. According to the results from Section 4.4.1 in Chapter 4, local DSI-3D models are trained exclusively with the Hilbert Curve positional indexing. The evaluation metrics, $F1_{max}$ and $Hits@1$, are summarized in Tables 6.4 and 6.5, respectively.

Table 6.4: Evaluation of local DSI-3D experts and state-of-the-art approaches across multiple LiDAR HD expert regions with fine-tuned LoGG3D-Net descriptors, using the $F1_{max}$ metric.

Area	$\mathbf{A}_{East}$	$\mathbf{B}_{East}$	$\mathbf{C}_{East}$	$\mathbf{D}_{East}$	$\mathbf{A}_{West}$	$\mathbf{B}_{West}$	$\mathbf{C}_{West}$	$\mathbf{D}_{West}$	$\mathbf{E}_{West}$
Exhaustive k-NN	**0.9967**	**0.9954**	**0.9946**	**0.9901**	**0.9917**	**0.9524**	**0.9553**	0.9491	**0.9852**
k-d tree	**0.9967**	**0.9954**	**0.9946**	**0.9901**	**0.9917**	**0.9524**	**0.9553**	0.9491	**0.9852**
LSH_{256}	0.9927	0.9902	0.9896	0.9893	0.9849	0.8972	0.9440	0.9497	0.9833
$HNSW_{32}$	0.9965	0.9950	0.9928	0.9853	**0.9917**	0.9425	0.9470	**0.9450**	0.9845
DSI-3D local expert (Hilbert)	0.9658	0.9735	0.9563	0.9362	0.9396	0.8818	0.8693	0.8480	0.9199

Table 6.5: Evaluation of local DSI-3D experts and state-of-the-art approaches across multiple LiDAR HD expert regions with fine-tuned LoGG3D-Net descriptors, using the $Hits@1$ metric.

Area	$\mathbf{A}_{East}$	$\mathbf{B}_{East}$	$\mathbf{C}_{East}$	$\mathbf{D}_{East}$	$\mathbf{A}_{West}$	$\mathbf{B}_{West}$	$\mathbf{C}_{West}$	$\mathbf{D}_{West}$	$\mathbf{E}_{West}$
Exhaustive k-NN	0.9930	0.9905	0.9888	0.9803	0.9810	0.9091	0.9105	0.8978	0.9687
k-d tree	0.9930	0.9905	0.9888	0.9803	0.9810	0.9091	0.9105	0.8978	0.9687
LSH_{256}	0.9163	0.9337	0.8936	0.8278	0.9180	0.6940	0.6638	0.6753	0.8695
$HNSW_{32}$	0.9909	0.9873	0.9808	0.9657	0.9794	0.8910	0.8851	0.8734	0.9634
DSI-3D with Hilbert	0.8995	0.9276	0.8712	0.8169	0.8402	0.6725	0.6674	0.6020	0.7821

With only 20 hours of training, the experts achieve performance that is almost comparable to that of LSH in terms of retrieval quality. The generalization ability of the descriptors can be observed from the results in the $Paris_{West}$ area, where a performance drop is noted for both DSI-3D and LSH, but also, to a lesser extent, for the exhaustive search and HNSW. While additional training or more refined hyperparameter tuning might potentially improve DSI-3D's performance, these results may also suggest a scalability bottleneck inherent in the current training and indexing configuration. In the following section, we investigate whether these specialized experts can maintain higher retrieval performance compared to a larger single DSI-3D model trained over the entire Paris areas, under equivalent total training time.

6.2.2.5 Gate training

The gating network is responsible for determining which expert(s) are activated for a given input. Our design draws inspiration from the MoE approach introduced in GShard (Lepikhin et al., 2020). In its simplest configuration, the gate consists of a single linear layer that maps the input descriptor of dimension $D = 256$ to a vector of logits corresponding to the E available experts:

$$g(x) = x.W_g + b_g, W_g \in \mathbb{R}^{E \times D} \tag{6.9}$$

where b_g denotes the layer bias.

In addition to this simple configuration, we experiment with a deeper architecture inspired by the PyTorch tutorial [1]. This model includes four linear layers of increasing and decreasing dimensionality (e.g., $256 \rightarrow 256 \rightarrow 1024 \rightarrow 256 \rightarrow 4$).

Training strategies

Two training strategies are investigated for the gating network. In the first approach, the gate is trained jointly with the expert models within the Hugging Face trainer pipeline. The pre-trained experts are frozen and encapsulated within a composite model that includes the gating module. This ensures that only the gate parameters are updated during backpropagation. The overall training loss is inspired by Equation 6.6, where L_m corresponds to the GIT cross-entropy loss defined in Equation 3.1, and α is set to 0.01.

$$L_{Total} = L_m + \alpha \cdot L_{Aux} \tag{6.10}$$

In the second approach, the gating network is trained independently as a classifier, where each class corresponds to a specific expert. Because the expert regions may overlap and a single point cloud can be retrieved by multiple experts, the problem is formulated as a multi-label classification task. In this context, the standard categorical cross-entropy loss, which assumes a single valid class per sample ($y = c_i$, with $c_i \in 0, \ldots, N$), is not appropriate. Instead, we employ its multi-label variant, the Binary Cross-Entropy (BCE) loss, where the target y is a Boolean vector indicating expert relevance. This training setup also serves to assess whether a simple linear layer is sufficient for effective routing in a simplified context, without integrating the expert modules. The BCE loss is defined as:

$$loss_{BCE}(x, y) = \frac{1}{N} \sum_{i=1}^{N} -\omega_n [y_i log(p(x_i)) + (1 - y_i) log(1 - p(x_i))] \tag{6.11}$$

where $P(x)$ is the probability associated with observation x and y is the target binary label (0 or 1).

The training and validation sets for both sets are built by aggregating the respective training set of the experts, leading to a total number of descriptors higher than the initial sets, as seen in Table 6.6. This is due to the multiple presence of the same point cloud with different target labels for the gate training as a classifier.

Table 6.6: LiDAR dataset distribution for gate training.

Area	**Paris**$_{East}$	**Paris**$_{West}$
Scenes count	46881	85731
Database size	37159	47446
Validation set	8930	12091

Training the gate as an independent multi-label classifier proves to be more stable and requires less hyperparameter tuning compared to training within the MoE framework

[1] https://docs.pytorch.org/tutorials/beginner/basics/buildmodel_tutorial.html

with frozen experts. Consequently, this configuration is adopted for the remainder of the study. Figure 6.7 presents the gate accuracy and its average Top-1 confidence score on the evaluation set. The training lasts 80 epochs, with a learning rate of 0.002, representing a reasonable training time (around 20 min).

Table 6.7: Performance of the gating network trained as a multi-label classifier.

Train area	**Paris**$_{East}$	**Paris**$_{East}$	**Paris**$_{West}$
Linear layer	1	4	4
Exact activation accuracy	92.16%	99.19%	92.52%
Average activation probability	0.5470	0.5501	0.2731

On $Paris_{East}$, the gate model with 4 linear layers performs better than a single layer. It is then adopted as the gate network for $Paris_{West}$. Overall, the classifier-based gating strategy achieves high activation accuracy.

6.2.2.6 Retrieval evaluation

Independently of the gate training, the performance of the MoE architecture can be evaluated in a manner similar to Chapter 3, by replacing the single DSI-3D model with its MoE counterpart. The gate architecture also enables varying levels of expert activation sparsity, which we investigate by activating one, two, or all experts simultaneously. Tables 6.8 and 6.9 present the retrieval performance of standard retrieval approaches, a single DSI-3D model trained over the entire area, and three MoE configurations using DSI-3D as experts, each evaluated with different indexing strategies. For all DSI-3D training, point clouds are indexed using positional structured identifiers based on the Hilbert curve. Each DSI-3D expert is trained for 20 hours, while the larger DSI-3D models are trained for a duration equivalent to their corresponding cumulative expert training times (i.e., 80 hours and 100 hours, respectively).

Finally, the DSI-3D approach with MoE is not applied to $Paris_{East+West}$ using a Top-2 or dense MoE configuration, since the expected gain from complementary experts is found to be marginal based on observations from the two individual zones. Simultaneously activating experts from two distinct regions may not reasonably be expected to improve performance.

Table 6.8: Evaluation of LiDAR HD retrieval, using the $Hits$@1 metric. The DSI-3D training time is equal to the cumulative training time of the DSI-3D experts.

Area	**Paris**$_{East}$	**Paris**$_{West}$	**Paris**$_{East+West}$
database size	20892	37710	58602
Exhaustive k-NN	0.9849	0.9195	0.9491
k-d tree	0.9849	0.9195	0.9491
LSH_{256}	0.8643	0.7143	0.7500
HNSW_{32}	0.9738	0.8940	0.9209
DSI-3D	0.6493	0.2398	/
DSI-3D with MoE Top-1	0.8281	0.7436	0.7413
DSI-3D with MoE Top-2	0.8285	0.7438	/
DSI-3D with dense MoE	0.8287	0.7438	/

Table 6.9: Evaluation of LiDAR HD retrieval, using the $F1_{max}$ metric. The DSI-3D training time is equal to the cumulative training time of the DSI-3D experts.

Area	**Paris**$_{East}$	**Paris**$_{West}$	**Paris**$_{East+West}$
database size	20892	37710	58602
Exhaustive k-NN	0.9927	0.9593	0.9748
k-d tree	0.9927	0.9593	0.9748
LSH_{256}	0.9888	0.9300	0.9540
HNSW_{32}	0.9900	0.9524	0.9668
DSI-3D	0.8438	0.8271	/
DSI-3D with MoE Top-1	0.8534	0.8207	0.8120
DSI-3D with MoE Top-2	0.8572	0.8219	/
DSI-3D with dense MoE	0.8594	0.8226	/

These results suggest that geographic specialization leads to better performance compared to a model trained across the entire area, for a cumulative training time equivalent to that of experts. State-of-the-art acceleration methods remain faster than DSI-3D for this dataset size. However, with the exception of the k-d tree, all these methods show a performance decrease as the database size increases, a phenomenon not observed with the small KITTI sequences. The retrieval times of both state-of-the-art methods and DSI-3D (with and without a gating mechanism) are presented in Table 6.10. The beam search width is set to 10.

Table 6.10: Retrieval time (in ms), to obtain a Top-10, between standard and Mixture-of-Experts based approaches.

Area	**Paris**$_{East}$	**Paris**$_{West}$	**Paris**$_{East+West}$
database size	20892	37710	58602
Exhaustive search	9.9	20.2	32.5
k-d tree	11.6	16.5	25.9
LSH_{256}	0.7	1.1	1.7
HNSW_{32}	0.6	0.4	0.5
DSI-3D	57.5	59.8	/
DSI-3D with MoE Top-1	85.3	91.0	89.6
DSI-3D with MoE Top-2	114.7	121.4	/
DSI-3D with dense MoE	398.2	534.0	/

From the classical approaches, we observe that the k-d tree can be even slower than an exhaustive search, a behavior explained by the curse of dimensionality (Beyer et al., 1999). LSH and HNSW remain good options for accelerating retrieval as long as the database contains fewer than 40000 descriptors. Regarding the impact of a gating architecture, activating two experts implies only a moderate additional cost on average, as many candidates are filtered out by the gate when the second expert receives a low activation score. In contrast, dense activation, in which all experts are used, requires passing through every DSI-3D model. This results in four or five sequential inferences, significantly increasing the overall retrieval time. Figure 6.6 shows the average retrieval time of the different methods tested on the KITTI sequence 22, $Paris_{East}$, $Paris_{West}$, and $Paris_{East+West}$. Exhaustive and k-d tree searches exhibit linear behavior in relation

to the database size, while LSH and HNSW have a much slower increase. DSI-3D with a Top-1 MoE has an average retrieval time of 0.09s, which includes DSI-3D inference with the gate overhead, whereas a global DSI-3D trained directly on $Paris_{East}$ and $Paris_{West}$ obtains an average retrieval time of 0.06s. This reveals an additional retrieval complexity cost of adding a gating mechanism, measured at 0.03s. Note that here, these inferences are faster than DSI-3D retrieval times from Chapter 4 due to further optimization of the code.

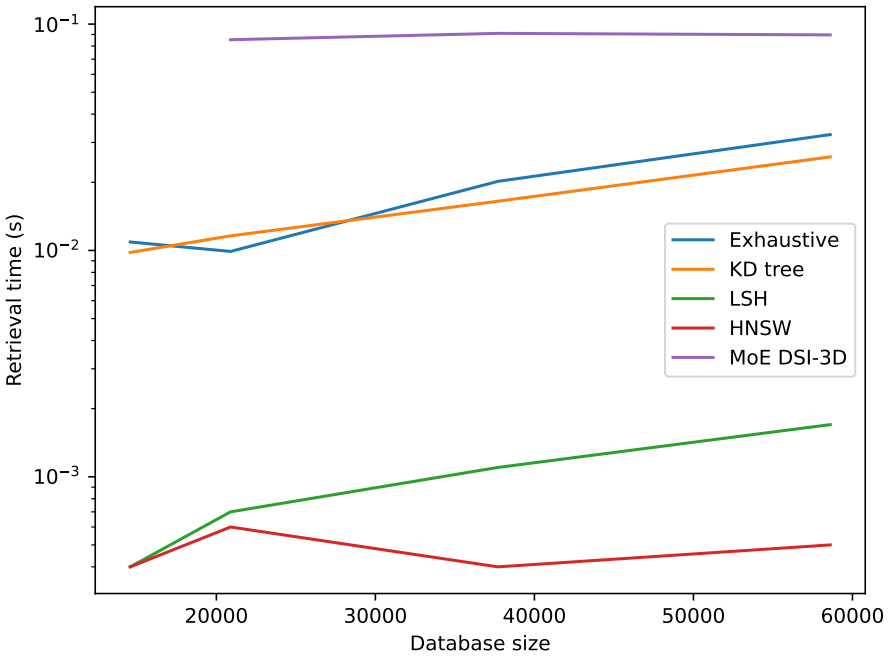


Figure 6.6: Retrieval times (in s) across all database sizes used in this thesis, presented in logarithmic scale.

6.3 Re-ranking

Another way to improve the robustness of retrieval is through a re-ranking stage applied to a subset of the retrieved candidates. In generative retrieval, DSI-3D produces a ranked list of candidate docids given a query. However, the Top-1 prediction may not always correspond to the true match, especially when multiple candidates share similar characteristics and thus have similar descriptors, as experimented with the LoGG3D-Net descriptors on the LIDAR HD dataset through a distance ratio analysis (see Appendix B). Re-ranking the Top-k candidates aims to refine this initial ordering and promote the most relevant items to higher ranks by relying on a more complex analysis, for example, geometric verification or similarity estimation between the query and retrieved candidates.

This section presents work carried out in collaboration with Kerrien Deneuville, a specialized Mastere PPMD[2] student at Geodata Paris, during his master thesis internship (Deneuville, 2025). In Section 6.3.1, we first revisit the state of the art in re-ranking methods, while Section 6.3.2 describes their application to point cloud retrieval.

6.3.1 Related work

This section presents a state-of-the-art review on global and local descriptor-based re-ranking approaches, with a more detailed presentation of SpectralGV and its limitations.

[2]Photogrammétrie, Positionnement et Mesure de Déformations

To address SpectralGV sensibility to point cloud transformations, additional keypoint detection approaches are reviewed.

Global descriptor-based re-ranking

We first present a re-ranking based on global descriptors, which are often used in image retrieval (Radenović et al., 2018; Zhong et al., 2017).

Among global approaches, Query Expansion (Radenović et al., 2018) evolves into a wide variety of descriptor aggregation techniques. The core idea is to use the retrieved first-order neighbors to refine the original query representation. Specifically, features from the most similar candidates are aggregated with the query descriptor to form an updated query vector. This refined query is then used for a second retrieval phase, effectively capturing richer contextual information and improving candidate ranking. Changing the number of candidates used and how they are aggregated produces numerous variations of this approach. Since Query Expansion relies only on aggregated features, it can be generalized to other types of data.

Other variants of global descriptor-based approaches use higher-order neighbors (or neighbors of neighbors). For example, k-reciprocal nearest neighbors (Zhong et al., 2017), originally used for image-based person recognition, constructs new binary features, namely k-reciprocal features, which encode mutual neighborhood relationships between pairs of features. A Jaccard distance is then computed from these new descriptors, and combined to the original distance measurement, re-ordering the candidate set. This process can also be extended to any similarity-based ranking. There are also re-ranking methods directly tailored for 3D place recognition task, such as the Lightweight Transformer Re-Ranking (TReR) approach (Barros et al., 2023). Using only global descriptors in the re-ranking has the advantage of facilitating its application to other loop closure detection models. The re-ranking is achieved with a transformer that receives a list of candidates and returns new ranks similar to those in information retrieval.

Local descriptor-based re-ranking

The re-ranking mechanism can also be enhanced by local information, such as in images or point clouds. An example of re-ranking based on geometric verification for visual place recognition is DEep Local and Global (DELG) (Cao et al., 2020). It is an end-to-end trainable model based on a hierarchical CNN network to extract both global and local features from images. Its current implementation may require some tuning to be applied to other modalities. Global features are used to obtain a query top-k neighbors and use Random Sample Consensus (RANSAC (Fischler and Bolles, 1981)) for robust spatial matching of local descriptors. Another example of vision-based geometric verification is CVNet (Lee et al., 2022), which has a dedicated CNN network, CVNet-Rerank, to re-rank images according to their local features.

An issue with RANSAC and its newer variants is being less effective in re-ranking within a large number of points. To address this, the spectral method for geometric verification (SpectralGV) (Vidanapathirana et al., 2023) proposes a graph-based geometry verification for point cloud re-ranking. Its core idea is to represent each query-nearest neighbor pair with a graph and compute a *fitness* score based on both local descriptor correspondence similarity and their relative distances, without the need for a point cloud registration step. This approach can only be extended to modalities that have relative distances between their local descriptors. Because of its efficiency and scalability, we choose SpectralGV as method of re-ranking scenes from LiDAR HD.

Focus on SpectralGV

Place recognition methods involving retrieval, such as LoGG3D-Net, first compute local descriptors and then encapsulate them into global descriptors, and the top-k is defined based on the latter. SpectralGV proposes exploiting the local descriptors after the initial global descriptor top-k retrieval, and re-rank the candidates list without a direct registration step.

The set of local descriptors of the query is matched with those of the candidates to build a similarity graph. The purpose of this graph is to capture the spatial consistency between query-candidate correspondences, including their spatial neighbors.

Each graph node represents a pair of corresponding points $C_i = (x_i, y_i)$, where x_i is a point from the query point cloud, and y_i is a point from the candidate point cloud. The graph is then represented by an adjacency matrix $M \in \mathbb{R}^{n \times n}$. To control the graph size and computational cost, the number of keypoints is capped by selecting the top n points from both the query and candidate sets. Each element $m_{i,j}$ of M represents the spatial compatibility score of the two correspondences C_i and C_j, which is defined as:

$$m_{i,j} = [1 - \frac{d_{i,j}^2}{d_{thres}^2}]_+, \; d_{i,j} = |\|x_i - x_j\| - \|y_i - y_j\|| \tag{6.12}$$

where d_{thres} is a fixed distance threshold, d_{thres} is a fixed distance threshold, set at 10 m, and $[.]_+$ is the ReLU function. The ReLU function ensures that M do not have negative elements and the distance d_{thres} controls the impact of point distance. If the points are further than the threshold $d_{i,j} \geq d_{thres}$, then the compatibility score $m_{i,j} = 0$. If p_i and p_j have the same positions, then $d_{i,j} = 0$, resulting in the highest compatibility score, $m_{i,j} = 1$. The objective is to identify a subset of correspondences $\hat{C} \subseteq C$ forming a spatially consistent cluster. The most consistent subset is obtained by maximizing the inter-cluster spatial compatibility score:

$$s = \sum_{i,j \in C_{max}} m_{i,j} \tag{6.13}$$

The cluster is then represented by a binary vector v where $v(c_i) = 1$ if c_i belong to the cluster C_{max}. Equation 6.13 can be rewritten as:

$$s = v^T M v \tag{6.14}$$

The sets correspondence is established using Spectral Matching (Leordeanu and Hebert, 2005) to find the most likely combination (see Figure 6.7). According to Spectral Matching, the vector v^* maximizing the inter-cluster score is found by resolving:

$$v^* = arg\ max_v(\frac{v^T M v}{v^T v}) \tag{6.15}$$

The Rayleigh quotient is used to approximate v^* as the principal eigenvector of M and the maximum inter-cluster score is:

$$s^* = v^{*T} M v^* \tag{6.16}$$

These scalar scores s^* are computed for every pair of query-candidates, enabling a new ranking of the candidate sets, as seen in Figure 6.7.

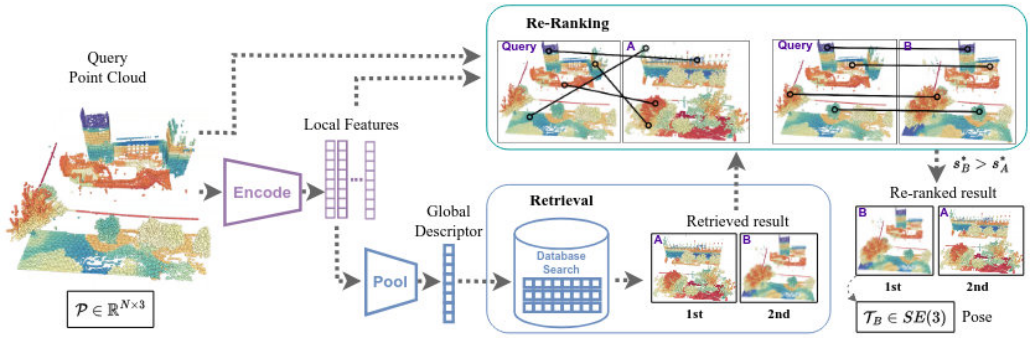
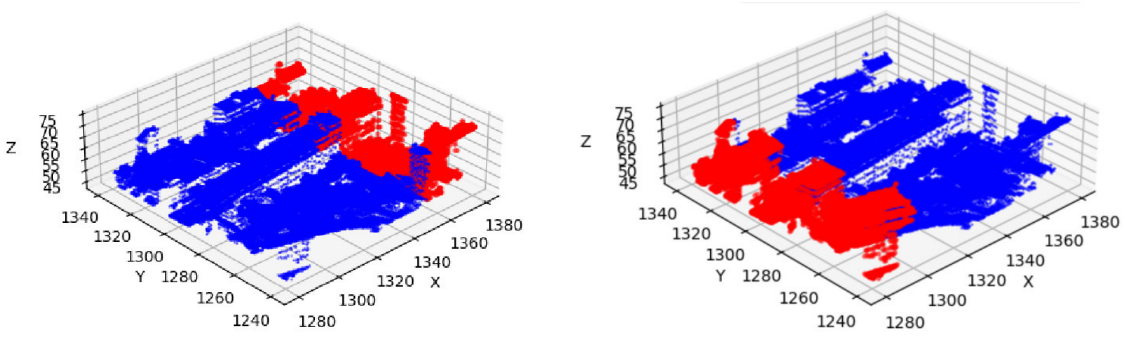


Figure 6.7: SpectralGV method for re-ranking in 3D point clouds. Image taken from (Vidanapathirana et al., 2023).

Limit of SpectralGV

In order to challenge the re-ranking capacity of the model on a truly unknown query, the orientation, more generally the acquisition viewpoint of the point cloud should remain unknown. This reveals a limitation of SpectralGV, as only a subset n of points (15000 points in the implementation of the authors) is considered, due to the high cost of the adjacency matrix. Additionally, the subset of points selected, simply the first n in the list of points in their implementation, is itself naturally altered by the potential transformation existing between the query and the database. For example, in Figure 6.8, Figure 6.8a presents a point clouds from $Paris_{East}$ in blue and the points retained by SpectralGV, forming straight lines along the X-axis in red. However, Figure 6.8b shows that this selection is highly affected by rotation, as the local features considered for the same scene are different, indicating that the selection process is not robust to geometrical transformations.



(a) Original point cloud

(b) Re-oriented point cloud

Figure 6.8: Subset of local descriptors selected (red) by SpectralGV before (a) and after a rotation (b) of 180 degrees. Figure (b) is displayed in its original orientation after SpectralGV points selection. Figures taken from (Deneuville, 2025).

To minimize this problem, we propose to introduce a selection step of the points, invariant to such transformations. We now explore existing robust keypoint detection methods applied to 3D point clouds.

Keypoint detection

Among recent state-of-the-art 3D keypoint detection approaches, there is the Salient Keypoint Detector (SKD) (Tinchev et al., 2021), which identifies salient points using the gradient of local descriptors, point context, and principal component analysis. More recently, D3feat (Bai et al., 2020) is directly trained to detect and encode point features

jointly. The encoding is based on KPconv (Thomas et al., 2019) (see Section 2.3 from Chapter 2) adapted to be invariant to point cloud density. Keypoints are detected according to a saliency score map that identifies features that are dissimilar from their local neighbors.

In their respective articles, these solutions are evaluated according to registration quality between 3D point clouds, in terms of relative translation error (RTE) and of relative rotation error (RRE); the results obtained, taken from the articles, are presented in Table 6.11.

Table 6.11: Comparison of keypoint detectors on KITTI dataset. Results are taken from their respective papers. The best results are in bold.

Detector	RTE (m)	RRE (°)
SKD (Tinchev et al., 2021)	0.31 ± 0.27	1.11 ± 0.89
D3Feat (Bai et al., 2020)	**0.07 ± 3e-3**	**0.24 ± 6e-4**

For the following, we chose D3feat as a keypoint detector since it demonstrates the best performance.

6.3.2 Re-ranking on LiDAR HD

In our improvement of SpectralGV, instead of naively selecting the first points of a point cloud, we replace this step by exploiting D3Feat which highlights a subset of salient keypoints according to its own descriptors. This new selection is then used by SpectralGV to compute compatibility scores with the query. The re-ranking accomplished afterward by SpectralGV is based on the local descriptors obtained from the first k-NN search, here from LoGG3D-Net fine-tuned on LiDAR HD.

Figure 6.9 shows the retrieval and re-ranking performances, measured by the $Hits@1$ score (in %), for query point clouds rotated by an angle θ along the Z-axis (in degrees). The re-ranking is evaluated on the center LiDAR HD tile of $Paris_{East}$, by using the fine-tuned LOGG3D-Net descriptors, which are trained with a voxel size of 0.1 m without rotation augmentation during training, making these descriptors sensible to rotations (following Training B strategy presented in Section 5.3.3).

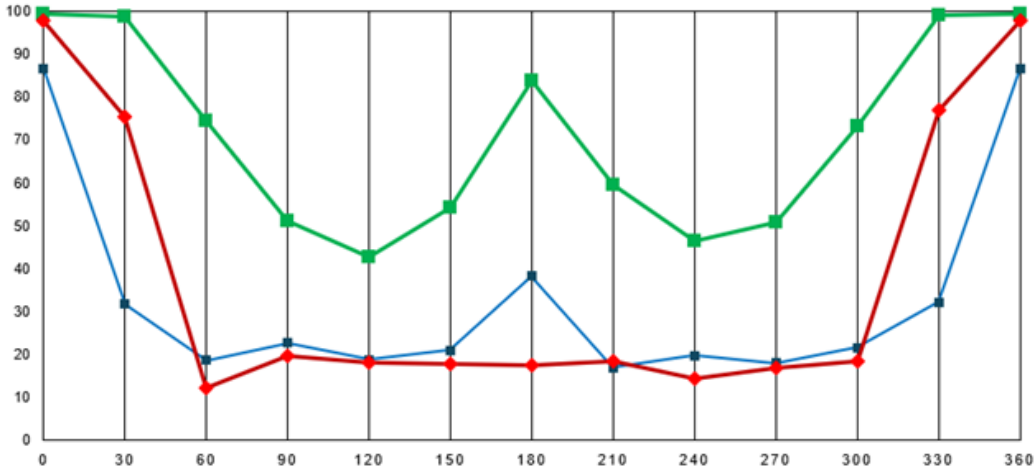


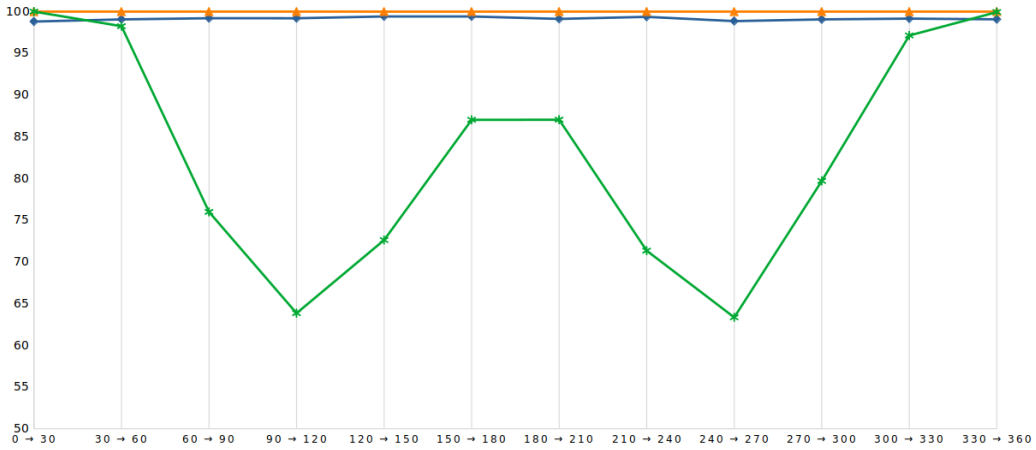
Figure 6.9: SpectralGV and D3Feat re-ranking effect on *Hits*@1 score, according to a rotation along the Z-axis (in degrees). Three configurations are compared with a simple exhaustive search (in blue), an exhaustive search with SpectralGV (in red) and an exhaustive search with SpectralGV+D3Feat (in green). Figure taken from (Deneuville, 2025).

When no rotation is applied to the query ($\theta = 0°$ or $360°$), we first observe that the addition of SpectralGV improves retrieval performance. Then, as expected, introducing rotation causes a performance drop across all methods since here the descriptors are not explicitly trained to be rotation-invariant. The exhaustive search, re-ranked with SpectralGV only, is not noticeably improved because of the rotations, while the addition of D3Feat significantly improves the *Hits*@1 score, achieving an increase of at least 30% under rotated query conditions. Interestingly, even without explicit rotation augmentation during training, a slight symmetry invariance emerges around $\theta = 180°$, although this effect is not captured when using SpectralGV alone.

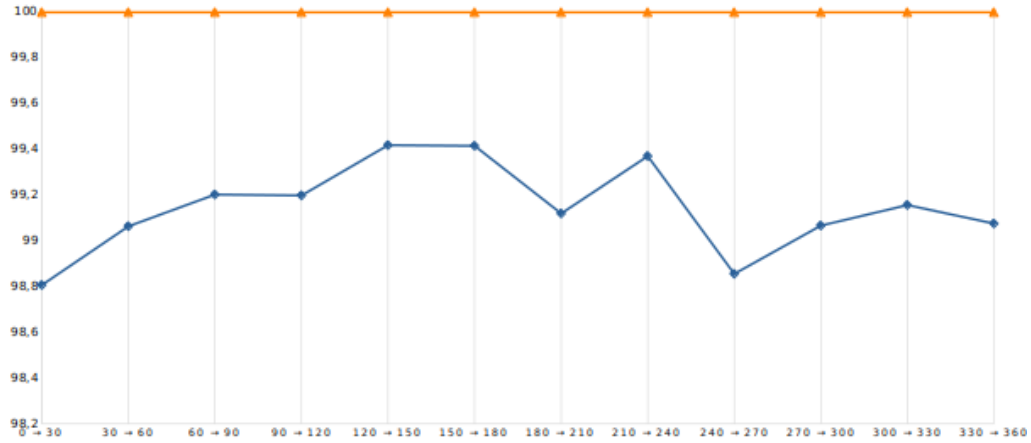
In conclusion, we observe that using a re-ranking approach significantly improves the results of the search for the most similar clouds. We are also able to propose a simple improvement to the state-of-the-art approach, SpectralGV, making it more robust to geometric transformations (in this case, rotation) that may exist between the input cloud and the queried cloud database.

Rotational description invariance

The previous experiment evaluated the benefits of applying D3Feat and SpectralGV to rerank a list of candidates retrieved through an exhaustive search, using descriptors that were not trained with rotated data. Figure 6.10 compares the results of the exhaustive search, re-ranked (in green), with retrieval applied to a descriptor trained to be invariant to rotation. In this figure, the blue curve represents the *Hits*@1 score of an exhaustive search on the rotation-augmented descriptors, while the orange one shows the scores after re-ranking with SpectralGV+D3Feat. For this evaluation, the protocol is slightly modified by querying over the entire dataset and applying a random θ from an interval of 30 degrees.



(a) Rotation augmentation effect on *Hits*@1 score (in %), according to a rotation along the Z-axis (in degrees).



(b) Rotation augmentation effect on *Hits*@1 score (in %), according to a rotation along the Z-axis (in degrees), (zoomed).

Figure 6.10: (a) Two additional configuration are compared with the previous exhaustive search with SpectralGV+D3Feat (in green), an exhaustive search with descriptor trained with rotation augmentation (in blue) and exhaustive search with descriptor trained with rotation augmentation with SpectralGV+D3Feat (in orange). (b) Zoomed figures, keeping only exhaustive search (in blue) and exhaustive search with descriptor trained with rotation augmentation with SpectralGV+D3Feat (in orange). Figures taken from (Deneuville, 2025).

The first observation is that the new descriptors are indeed invariant to rotation. However, re-ranking still improves the results by an average of 0.9%, even if the number of errors is small for an exhaustive search on only one LiDAR tile.

To conclude, while the previous experiments show the benefits of adding a re-ranking pipeline, this experiment again illustrates the gains in robustness to rotation. These ways of improving the retrieval in 3D point clouds were used within an exhaustive search, with no errors in the nearest neighbors returned. They are particularly valuable when applied after an approximate search method such as those studied in the thesis.

6.4 Conclusion

In this chapter, we have presented two approaches that may complement the current implementation of DSI-3D.

First, inspired by the Mixture-of-Experts capacity to scale transformer models while maintaining task versatility, we adapt them for the retrieval of 3D point clouds. This way, each MoE expert is a DSI-3D model trained on LiDAR HD, and a router, or gating network, activates a subset of experts relevant to the query. The two Paris areas were divided into four and five experts, respectively, with equal surface areas and a similar number of point clouds. This has the benefit of keeping the same configuration and training time for each expert, as well as avoiding load imbalance during the router training. Two router training strategies were tested, with the multi-label classifier being the more stable one to train. The dataset change presented a more challenging point cloud retrieval task compared to KITTI, which was primarily linear by the nature of mobile mapping. The fine-tuned tokenizer on Hilbert indices contributes to improved robustness in two-dimensional data distributions, with better token repartition among the search space. An MoE architecture also simplifies the experts training as they can share the same hyperparameters.

The second complement to DSI-3D, although applicable to any 3D retrieval framework, consists in re-ranking the nearest neighbors with geometrical constraints. An additional challenge arises from handling geometrical transformations, here rotations around the Z-axis, as in realistic scenarios where the query point cloud may not share the same orientation convention as the database. To address this issue, a two-step graph-based re-ranking method was introduced, combining D3Feat for salient keypoint detection and SpectralGV for geometric similarity computation.

Conclusion and Perspectives

This chapter reviews the contributions of this thesis and presents perspectives for future work.

Contributions

The first contribution of this thesis lies in the application of information retrieval concepts to the task of point cloud retrieval through the adaptation of the DSI to 3D data. This required bridging the modality gap between the original DSI framework, which generates document identifiers (docids) from textual queries, and 3D point cloud inputs. To achieve this, we designed a point cloud-based architecture by replacing the image encoder used in Vision Transformers (here, GIT) with an off-the-shelf point cloud encoder, namely LoGG3D-Net. A second key adaptation concerns the design of point cloud identifiers. Since GIT generates sequences of tokens autoregressively during beam search (Sabuncuoglu and Bayiz, 1999), only the most promising candidates are kept at each decoding step. Thus, the tokenization of docids has a direct impact on retrieval performance. Poorly structured identifiers may cause the model to prematurely discard valid candidates. To address this, we introduce two positional structured identifiers, spatial indexing schemes that embed geographical proximity within docids, leading to improved retrieval precision as neighboring point clouds are assigned similar identifiers.

The second contribution involves the application of LiDAR HD dataset as a new extensible point cloud dataset for place recognition. Inspired by existing benchmarks such as KITTI (Geiger et al., 2013) and CS-Campus3D (Guan et al., 2023), our dataset covers two distinct regions of Paris: from Saint-Mandé to Vincennes in the east and from Bois de Boulogne to Champ-de-Mars in the west. As no pre-trained model existed for this dataset, we had to precisely fine-tune LoGG3D-Net (Vidanapathirana et al., 2022) at multiple spatial scales to study multi-resolution representations, ultimately selecting a mono-resolution with a voxel size of 2 meters as the optimal configuration.

Finally, **the third contribution extends the DSI-3D framework**. **The first extension** explores the integration of a Mixture of Experts architecture, enabling sparse expert activation during generative retrieval. The LiDAR HD dataset from both Paris areas was further subdivided into sectors of equal surface area, and a dedicated DSI-3D model was trained for each sector. These geographically specialized DSI-3D models were incorporated as experts. A gating network was also trained as a multi-label classifier to determine the most relevant experts for a given query. This approach effectively expands the spatial coverage of DSI-3D while maintaining inference efficiency, offering a scalable solution that improves retrieval performance with minimal increases in computational cost.

The second extension of DSI-3D focuses on a complementary study dedicated to nearest neighbor reclassification strategies based on geometric information. This work was carried out as part of a final-year internship to explore reclassification methods for search results based on local spatial coherence. This study focused on reclassifying candidates

obtained through exhaustive search but can be transposed to a list of candidates obtained through DSI-3D beam searches. An initial approach was to apply a state-of-the-art solution such as SpectralGV, using local cloud descriptors to construct similarity graphs. This approach was then complemented by D3Feat to detect salient points in HD LiDAR scenes. The combination of these two methods improves search robustness, particularly in cases where the orientation of the query cloud is unknown and is therefore randomly defined.

Perspectives

This section outlines potential continuations of this thesis work. The first short-term direction, which is a direct continuation of the present work, is to apply recent generative retrieval strategies, as they directly address DSI design limitations, in particular by identifying better learned docid representations.

Then, the medium-term directions could be to explore solutions to further improve the scalability and models by exploring recent MoE architectures designed for DSI models as well as continual learning solutions to reduce the maintenance costs.

Finally, a long-term perspective could be to extend our model to multi-source and/or multimodal retrieval scenarios.

Generative retrieval

There are several design limitations with our current generative retrieval model. As stated in Relevance-based Identifiers for Prefix-Oriented Ranking (RIPOR) (Zeng et al., 2024), training a generative retrieval model involves two distinct steps. First, the docids are initialized through hierarchical k-means in DSI, or positional structured identifiers in DSI-3D, and then the model learns to map queries to these fixed docids.

However, keeping the docids fixed during learning can become a bottleneck. This effect was observed in Chapter 6, where using a pretrained tokenizer that was not optimized for our docid space made it more difficult for DSI-3D to learn the mapping between queries and docids in the dense LiDAR HD dataset. Another intrinsic limitation of generative retrieval lies in the sequential decoding of docid tokens from left to right, which can prematurely eliminate valid candidates during the beam search. Although positional structure identifiers ensure the proximity of relative docids, a 1D index remains a coarse representation of 2D spatial information.

To address these limitations, RIPOR (Zeng et al., 2024) proposes a relevance-based docid construction in which the model first acts as a dense encoder to find optimal token distributions across documents instead of using *handcrafted* docids, such as positional structure identifiers.

Another proposition of RIPOR is a prefix-oriented ranking optimization that progressively refines each token's decoding steps, instead of only computing a sequence-wise loss at the end of the generation. This is similar to curriculum learning (Bengio et al., 2009), where the model is trained progressively on harder examples. Future work could extend this idea to spatial domains by jointly learning both tokenizer and docid representations in a latent space that better captures spatial structures.

The current docids used for indexing point clouds are composed exclusively of digits to encode 2D spatial information. While this design has the advantage of producing human-readable identifiers, it may limit the expressiveness of the representation. A potential improvement would be to adopt a richer positional structured identifier that includes alphanumeric characters, similar to hexadecimal encoding, to increase the information capacity of each token. This concept could be further extended to include a broader set

of symbols, provided that a consistent mapping function exists between the point cloud attributes and their corresponding docids.

Mixture of experts

Beyond training and representation strategies, scaling the model to cover larger geographic areas introduces additional challenges. Larger coverage increases training time and makes the model more sensitive to urban changes. A Mixture-of-Experts architecture offers a solution by limiting retraining to only a subset of experts rather than to the entire model.

To address the issue of updating MoE architectures on dynamic datasets, MixLoRA-DSI (Huynh et al., 2025) recently proposed an efficient alternative that combines MoE with Low-Rank Adaptation (LoRA) (Hu et al., 2022). This approach allows each expert to be fine-tuned using a small set of additional parameters while keeping the base model frozen. Integrating such a mechanism into DSI-3D would enable continual learning and geographic expansion at minimal computational cost.

A natural extension of our current work would be to design a hierarchical MoE, with multiple gating layers where experts are grouped into higher-level categories, such as regions or cities (e.g, Paris, Lyon). Furthermore, DSI-3D currently always returns a point cloud from its training database, even when the query does not correspond to any known location. Considering the confidence score associated with each retrieval could allow the system to flag uncertain cases. For instance, if a DSI-3D model trained on Paris encounters a query from Lyon, a low confidence score could alert the user (or the model) that no suitable match exists within the indexed area.

Continual learning

Complementary to the MoE, a solution for model updates is continual learning (Thrun, 1995; Parisi et al., 2019), where DSI-3D could learn to retrieve new documents while maintaining its capacity for previously learned ones, mimicking human lifelong learning. This is a solution to avoid retraining a model or an expert for a small number of updates in the database. A potential drawback of integrating new information is catastrophic forgetting (McCloskey and Cohen, 1989), a phenomenon in which newly acquired knowledge interferes with previously learned information. Another drawback is forgetting events (Toneva et al., 2018), where the model can no longer correctly answer queries it previously resolved successfully. DSI++ (Mehta et al., 2022) proposes updating the DSI architecture while addressing issues with a generative memory method to generate new queries from previously seen documents and a loss function that promotes flatter loss minima, thereby reducing forgetting (Mirzadeh et al., 2020). Taking into account that those contributions may lead to easier updates of DSI-3D models, thereby reducing their maintenance costs.

Multi-sourced retrieval

Another promising direction concerns cross-modal retrieval, where the acquisition process may differ between the reference database and the query. Extending DSI-3D to handle multimodal 3D inputs, such as heterogeneous LiDAR datasets, aerial scans, or even image-derived 3D reconstructions, would require learning a shared latent space between modalities, similar to the approach proposed in CrossLoc3D (Guan et al., 2023). Such extensions would make generative retrieval applicable beyond static, homogeneous datasets to real-world use cases. An example is the automated verification of multimedia metadata by retrieving scenes built by Structure-from-Motion (SfM).

Another multimodal extension would involve applying indexing and retrieval to broader, more diverse datasets and query types, including 3D inputs alongside text and image data. This is similar to Multimodal Large Language Models (MLLMs) (Caffagni et al., 2024), in which multiple modalities are represented within a shared embedding space. Such an extension would require a study on information fusion, which can occurs at different levels, and modality alignment mechanisms so the model could effectively interpret these heterogeneous sources.

Appendix A

LoGG3D-Net descriptors robustness

Before exploiting the LoGG3D-Net descriptors in DSI-3D, we first evaluated their robustness under varying perturbations. This step is important because it reveals the robustness properties of the 3D encoder, which directly influences the performance of DSI-3D. We follow the evaluation protocol of the original LoGG3D-Net paper, which considers sequential place recognition for loop closure detection. The perturbations applied include rotation around the Z-axis, occlusion of a random portion of the point cloud, filtering of points beyond a cropping distance, and the addition of Gaussian noise. Importantly, these perturbations are applied only to the query, one at a time, while the database descriptors are formed from the unaltered point cloud descriptors. More precisely:

- **Rotation:** each sequence is traversed once per rotation step, with multiples of 45° applied from 0° to 360°. Only the query is rotated, while the database of point cloud descriptors remains unchanged.

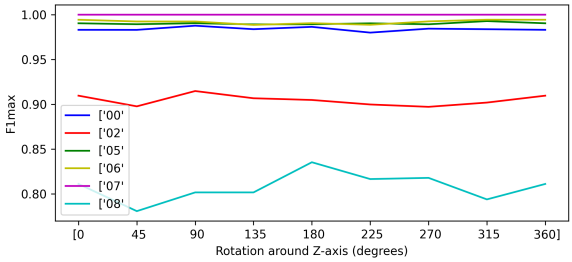
- **Occlusion:** for each sequence and occlusion angle, a random direction is chosen, and a cone-shaped portion of the query point cloud (originating from its center) is removed.

- **Cropping:** a distance threshold is applied at 5 m intervals, from 0 m to 20 m, removing all points beyond the current threshold.

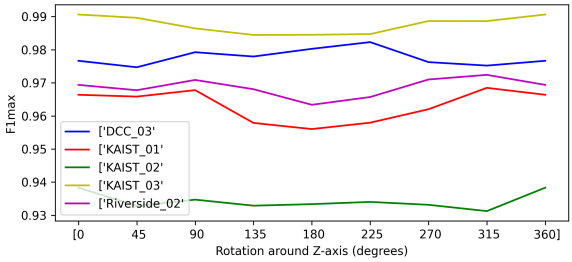
- **Noise:** Gaussian noise is added with standard deviations ranging from 0 cm to 9 cm to every point's coordinates.

The evaluation of descriptor robustness is based on LoGG3D-Net pre-trained models. These models are trained either on the first eleven KITTI sequences using a leave-one-out strategy or on the MulRAN sequences DCC 03, KAIST 01, KAIST 02, KAIST 03, and Riverside 2. Here, DCC refers to Dajeon Convention Center, and KAIST refers to a campus environment sequence. They are evaluated on KITTI sequences 00, 02, 05, 06, 07, and 08. For the MulRAN benchmark, the models are trained on DCC 01, DCC 02, Riverside 02, and Riverside 03, and evaluated on DCC 03, KAIST 01, KAIST 02, KAIST 03, and Riverside 02.

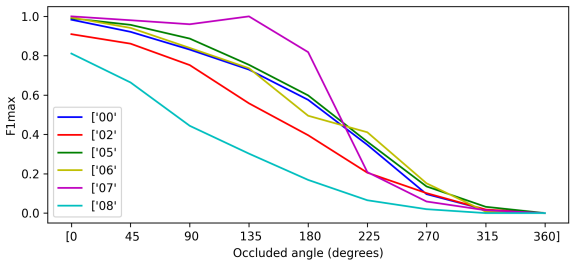
We choose to evaluate those models on both datasets in order to obtain a more general view of LoGG3D-Net descriptors across different datasets. Figure A.1 shows the evolution of the $F1_{max}$ metric under these different perturbations, evaluated on sequences from both the KITTI (Geiger et al., 2013) dataset, and another one, MulRan (Kim et al., 2020).



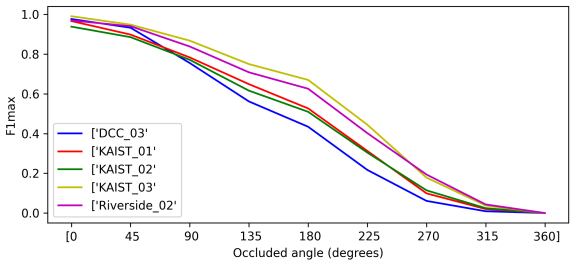
(a) KITTI – Rotation



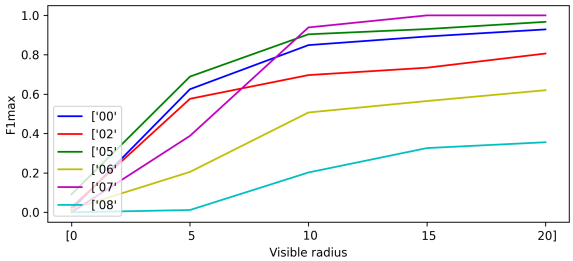
(b) MulRan – Rotation



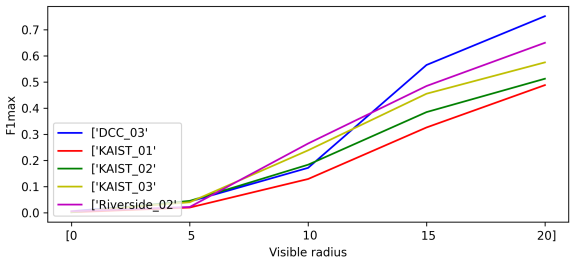
(c) KITTI – Occlusion



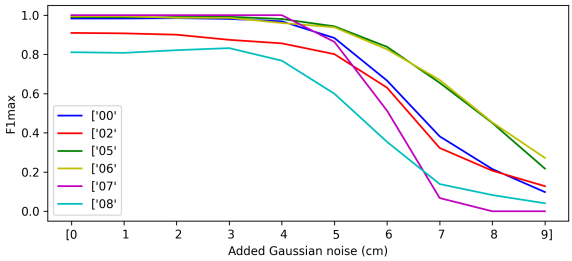
(d) MulRan – Occlusion



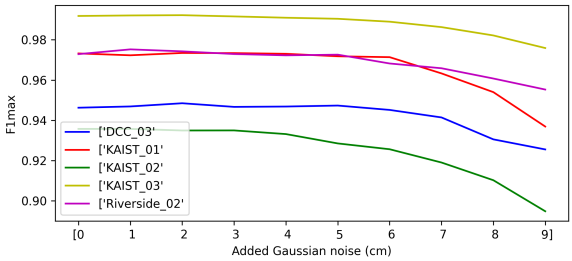
(e) KITTI – Cropping



(f) MulRan – Cropping



(g) KITTI – Noise



(h) MulRan – Noise

Figure A.1: LoGG3D-Net robustness evaluation for several KITTI (left column) and MulRAN (right column) sequences, under different perturbations: rotation around the Z-axis (a, b), partial occlusion of the query (c, d), cropping the query up to a maximum range from its center(e, f), and addition of Gaussian noise (g, h).

From Figures A.1a and A.1b, we observe that the scores globally remain invariant across all rotation angles for both datasets. This suggests that LoGG3D-Net successfully learns rotation-invariant features, a crucial property for place recognition since point clouds are often collected without consistent orientation.

A similar behavior between the two datasets is observed under occlusion, with the $F1_{max}$ score decreasing to around 0.6 when 180° of the query point cloud is removed, as seen in Figures A.1c and A.1d. This is expected since LoGG3D-Net relies on global descriptors, which are more sensitive to global geometry modifications.

Moreover, Figures A.1e and A.1f show that the descriptors exhibit a similar sensitivity to cropping across datasets. Removing points beyond a certain distance significantly alters the scene, especially for point clouds with larger sensing range (e.g, MulRAN) which may loose more scene information. Since KITTI point clouds generally cover shorter distances than those in MulRAN, the performance degradation is less pronounced for KITTI.

The effect of density is also evident with Gaussian noise: in KITTI, performance drops sharply when noise exceeds 5 cm, whereas in MulRAN the scores remain as high as 0.9 even with 9 cm of added noise, see Figures A.1g and A.1h.

Evaluating the effect of these perturbations provides a clearer understanding of LoGG3D-Net descriptors. They are rotation-invariant and, for the KITTI dataset, relatively insensitive to Gaussian noise below 3 cm. Both occlusion and cropping impact retrieval quality, but performance remains stable for occlusions below 45° and for cropping distances of 20 m or more.

In the thesis, DSI-3D models are trained on descriptors extracted from unperturbed point clouds, while inheriting the robustness properties identified above.

Appendix B

Distinctiveness of LoGG3D-Net descriptors

To assess whether the embeddings produced by the LoGG3D-Net model fine-tuned on $Paris_{East}$ (see Section 5.3.2 in Chapter 5) are sufficiently discriminative, we study their distinctiveness on the nine expert regions introduced for MoE in Section 6.2.2.4 of Chapter 6: five from $Paris_{West}$ (A_{West} B_{West} C_{West}, D_{West}, E_{West}) and four from $Paris_{East}$ (A_{East} B_{East} C_{East}, D_{East}).

For each expert region, the distance ratio (Lowe, 2004) is used to measure the distinctiveness of the best match relative to the second-best match, in terms of similarity between LoGG3D-Net descriptors. It is defined as:

$$distance_ratio = \frac{Cosine_similarity(Top\text{-}1, Ground_truth)}{Cosine_similarity(Top\text{-}2, Ground_truth)} \tag{B.1}$$

where $distance_ratio \in [0, 1]$. A ratio close to 0 indicates that the best match is clearly more similar to the query than the second-best one, meaning that the embedding is discriminative. Conversely, a ratio close to 1 suggests ambiguity between potential matches, implying that the representation struggles to distinguish between geometrically similar scenes.

We evaluate the retrieval capabilities of the fine-tuned LoGG3D-Net descriptors using an exhaustive search on the nine expert regions, as described previously. A ratio is computed for each heterogeneous expert region in order to measure the initial distinctiveness of the descriptor subsets on which the local DSI-3D models are trained.

The results are summarized in Table B.1 (distance ratio values of 0.0 and 1.0 are only rounding approximations).

Table B.1: LoGG3D-Net's descriptors distance ratio, fine-tuned on LiDAR HD.

Area	$\mathbf{A}_{East}$	$\mathbf{B}_{East}$	$\mathbf{C}_{East}$	$\mathbf{D}_{East}$	$\mathbf{A}_{West}$	$\mathbf{B}_{West}$	$\mathbf{C}_{West}$	$\mathbf{D}_{West}$	$\mathbf{E}_{West}$
Evaluation set	9222	9350	9308	9279	9435	9425	9419	9431	9736
Mean distance ratio	0.3074	0.2840	0.3424	0.3885	0.3114	0.4667	0.5398	0.5349	0.395
STD distance ratio	0.1784	0.1784	0.1935	0.2167	0.1936	0.2771	0.2474	0.2544	0.2255
min distance ratio	0.0227	0.0260	0.0228	0.0214	0.0	0.0	0.0	0.0	0.0
max distance ratio	1.0	0.9997	0.9998	0.9999	1.0	1.0	1.0	1.0	1.0

The distance ratio across expert regions highlights the following observations:

- On $Paris_{East}$, the western experts A_{East} and B_{East} seem to perform better, likely because they are closest to the training area, whereas D_{East}, which is mostly a forest area, has the highest ratio.

- On $Paris_{West}$, the eastern experts C_{West} and D_{West} covering Paris city perform worse; B_{West} is also affected by the forest area (see Figure 5.9 from Chapter 5). However, A_{West}, located farther from Paris, performs better.

For the $Paris_{East}$ subset, the average distance ratio is 0.33 ± 0.19, while for the $Paris_{West}$ subset, it increases to 0.44 ± 0.24. This suggests that the embeddings in $Paris_{East}$ are generally more discriminative than those in $Paris_{West}$.

This implies that the cosine distance of the second-best match is at least two to three times higher than that of the best match, indicating discriminative behavior. However, a low ratio alone does not imply that bad matches are filtered out. Similar to SIFT, we need to consider the ratio of failed matches. To further investigate the challenging retrieval cases, Table B.2 reports the statistics restricted to queries whose Top-1 retrieval is farther than 1 meter from the ground truth.

Table B.2: LoGG3D-Net's descriptors distance ratio of incorrect matches, fine-tuned on LiDAR HD.

Area	$\mathbf{A}_{East}$	$\mathbf{B}_{East}$	$\mathbf{C}_{East}$	$\mathbf{D}_{East}$	$\mathbf{A}_{West}$	$\mathbf{B}_{West}$	$\mathbf{C}_{West}$	$\mathbf{D}_{West}$	$\mathbf{E}_{West}$
PCD further than 1 meters	64	89	104	183	179	857	843	964	305
Mean distance ratio	0.9244	0.938	0.9339	0.9473	0.9166	0.9373	0.9176	0.9213	0.9290
STD distance ratio	0.0728	0.0640	0.0571	0.0522	0.0852	0.0648	0.0859	0.0900	0.0706
min distance ratio	0.6647	0.6884	0.7170	0.6759	0.4189	0.5338	0.5322	0.3137	0.5645
max distance ratio	1.0	0.9997	0.9998	0.9999	1.0	1.0	1.0	1.0	1.0

On the $Paris_{East}$ subset, only 1% of retrievals are located farther than 1 m from the ground truth, whereas on $Paris_{West}$, this proportion increases to 6%. This difference is likely due to the fact that LoGG3D-Net is fine-tuned exclusively on $Paris_{East}$ LiDAR HD scenes, leading to reduced generalization capability on $Paris_{West}$.

For the subset of incorrect retrievals, the mean distance ratio remains consistently high across both regions, with values of 0.93 ± 0.06 and 0.93 ± 0.07. The distribution of distance ratios shows a clear separation between correctly retrieved and incorrect samples, with poor cases concentrated near 0.9 and good cases mostly below 0.5.

Bibliography

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. (2023). Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Andoni, A. and Indyk, P. (2008). Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Communications of the ACM*, 51(1):117–122.

Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., and Sivic, J. (2016). Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307.

Arandjelovic, R. and Zisserman, A. (2013). All about vlad. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1578–1585.

Asai, A., Kasai, J., Clark, J. H., Lee, K., Choi, E., and Hajishirzi, H. (2020). Xor qa: Cross-lingual open-retrieval question answering. *arXiv preprint arXiv:2010.11856*.

Babenko, A. and Lempitsky, V. (2015). Aggregating deep convolutional features for image retrieval. *arXiv preprint arXiv:1510.07493*.

Bai, X., Luo, Z., Zhou, L., Fu, H., Quan, L., and Tai, C.-L. (2020). D3feat: Joint learning of dense detection and description of 3d local features. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6359–6367.

Barros, T., Garrote, L., Aleksandrov, M., Premebida, C., and Nunes, U. J. (2023). Trer: A lightweight transformer re-ranking approach for 3d lidar place recognition. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 2843–2849. IEEE.

Bay, H., Ess, A., Tuytelaars, T., and Van Gool, L. (2008). Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359.

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. (2009). Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517.

Beyer, K., Goldstein, J., Ramakrishnan, R., and Shaft, U. (1999). When is "nearest neighbor" meaningful? In *International conference on database theory*, pages 217–235. Springer.

Bi, X., Chen, D., Chen, G., Chen, S., Dai, D., Deng, C., Ding, H., Dong, K., Du, Q., Fu, Z., et al. (2024). Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*.

Bosse, M. and Zlot, R. (2009). Keypoint design and evaluation for place recognition in 2d lidar maps. *Robotics and Autonomous Systems*, 57(12):1211–1224.

Bosse, M. and Zlot, R. (2013). Place recognition using keypoint voting in large 3d lidar datasets. In *2013 IEEE international conference on robotics and automation*, pages 2677–2684. IEEE.

Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117.

Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., and Leonard, J. J. (2017). Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332.

Caffagni, D., Cocchi, F., Barsellotti, L., Moratelli, N., Sarto, S., Baraldi, L., Cornia, M., and Cucchiara, R. (2024). The revolution of multimodal large language models: a survey. *arXiv preprint arXiv:2402.12451*.

Cai, W., Jiang, J., Wang, F., Tang, J., Kim, S., and Huang, J. (2025). A survey on mixture of experts in large language models. *IEEE Transactions on Knowledge and Data Engineering*.

Cao, B., Araujo, A., and Sim, J. (2020). Unifying deep local and global features for image search. In *European conference on computer vision*, pages 726–743. Springer.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. (2020). End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer.

Carlevaris-Bianco, N., Ushani, A. K., and Eustice, R. M. (2016). University of michigan north campus long-term vision and lidar dataset. *The International Journal of Robotics Research*, 35(9):1023–1035.

Chen, L., Wu, P., Chitta, K., Jaeger, B., Geiger, A., and Li, H. (2024). End-to-end autonomous driving: Challenges and frontiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Chen, W., Chen, X., Zhang, J., and Huang, K. (2017). Beyond triplet loss: a deep quadruplet network for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 403–412.

Choy, C., Gwak, J., and Savarese, S. (2019a). 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3075–3084.

Choy, C., Park, J., and Koltun, V. (2019b). Fully convolutional geometric features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8958–8966.

Cui, Y., Chen, X., Zhang, Y., Dong, J., Wu, Q., and Zhu, F. (2022). Bow3d: Bag of words for real-time loop closing in 3d lidar slam. *IEEE Robotics and Automation Letters*, 8(5):2828–2835.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

Deneuville, K. (2025). Reconnaissance de lieu à large échelle dans les nuages de points 3d. Master's thesis, ENSG-Géomatique, Université Gustave Eiffel, Champs-sur-Marne, FR.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

Dong, W., Moses, C., and Li, K. (2011). Efficient k-nearest neighbor graph construction for generic similarity measures. In *Proceedings of the 20th international conference on World wide web*, pages 577–586.

Dosovitskiy, A. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Douze, M., Guzhva, A., Deng, C., Johnson, J., Szilvasy, G., Mazaré, P.-E., Lomeli, M., Hosseini, L., and Jégou, H. (2024). The faiss library. *arXiv preprint arXiv:2401.08281*.

Dubey, S. R. (2021). A decade survey of content based image retrieval using deep learning. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(5):2687–2704.

Faloutsos, C. and Roseman, S. (1989). Fractals for secondary key retrieval. In *Proceedings of the eighth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 247–252.

Fedus, W., Zoph, B., and Shazeer, N. (2022). Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

Gage, P. (1994). A new algorithm for data compression. *C Users Journal*, 12(2):23–38.

Gálvez-López, D. and Tardos, J. D. (2012). Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on robotics*, 28(5):1188–1197.

García-Hernández, A., Giubilato, R., Strobl, K. H., Civera, J., and Triebel, R. (2024). Unifying local and global multimodal features for place recognition in aliased and low-texture environments. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3991–3998. IEEE.

Gaydon, C. (2022). Myria3d: Deep learning for the semantic segmentation of aerial lidar point clouds. IGN (French Mapping Agency).

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *The international journal of robotics research*, 32(11):1231–1237.

Gillick, D., Presta, A., and Tomar, G. S. (2018). End-to-end retrieval in continuous space. *arXiv preprint arXiv:1811.08008*.

Gionis, A., Indyk, P., Motwani, R., et al. (1999). Similarity search in high dimensions via hashing. In *Vldb*, volume 99, pages 518–529.

Godil, A. and Wagan, A. I. (2011). Salient local 3d features for 3d shape retrieval. In *Three-Dimensional Imaging, Interaction, and Measurement*, volume 7864, pages 275–282. SPIE.

GOOS, J., Rotterdam, S., and EDWARD VERBREE, D. (2011). Towards a national 3d spatial data infrastructure: case of the netherlands.

Graham, B., Engelcke, M., and Van Der Maaten, L. (2018). 3d semantic segmentation with submanifold sparse convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9224–9232.

Gray, R. (1984). Vector quantization. *IEEE Assp Magazine*, 1(2):4–29.

Groves, P. D. (2011). Shadow matching: A new gnss positioning technique for urban canyons. *The journal of Navigation*, 64(3):417–430.

Guan, T., Muthuselvam, A., Hoover, M., Wang, X., Liang, J., Sathyamoorthy, A. J., Conover, D., and Manocha, D. (2023). Crossloc3d: Aerial-ground cross-source 3d place recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11335–11344.

Harris, C., Stephens, M., et al. (1988). A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Manchester, UK.

Hartley, R. and Zisserman, A. (2003). *Multiple view geometry in computer vision*. Cambridge university press.

Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al. (2022). Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Huynh, T.-L., Vu, T.-T., Wang, W., Le, T., Gašević, D., Li, Y.-F., and Do, T.-T. (2025). Mixlora-dsi: Dynamically expandable mixture-of-lora experts for rehearsal-free generative retrieval over dynamic corpora. *arXiv preprint arXiv:2507.09924*.

Indyk, P. (2004). Nearest neighbors in high-dimensional spaces. *Handbook of discrete and computational geometry*.

Indyk, P. and Motwani, R. (1998). Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Jégou, H., Douze, M., and Schmid, C. (2010). Improving bag-of-features for large scale image search. *International journal of computer vision*, 87(3):316–336.

Jegou, H., Douze, M., and Schmid, C. (2010). Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):117–128.

Jégou, H., Douze, M., Schmid, C., and Pérez, P. (2010). Aggregating local descriptors into a compact image representation. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3304–3311. IEEE.

Johnson, A. E. and Hebert, M. (2002). Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on pattern analysis and machine intelligence*, 21(5):433–449.

Jordan, M. I. and Jacobs, R. A. (1994). Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. (2020). Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Kibriya, A. M. and Frank, E. (2007). An empirical comparison of exact nearest neighbour algorithms. In *European conference on principles of data mining and knowledge discovery*, pages 140–151. Springer.

Kim, G., Choi, S., and Kim, A. (2021). Scan context++: Structural place recognition robust to rotation and lateral variations in urban environments. *IEEE Transactions on Robotics*, 38(3):1856–1874.

Kim, G. and Kim, A. (2018). Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4802–4809. IEEE.

Kim, G., Park, Y. S., Cho, Y., Jeong, J., and Kim, A. (2020). Mulran: Multimodal range dataset for urban place recognition. In *2020 IEEE international conference on robotics and automation (ICRA)*, pages 6246–6253. IEEE.

Knights, J., Vidanapathirana, K., Ramezani, M., Sridharan, S., Fookes, C., and Moghadam, P. (2022). Wild-places: A large-scale dataset for lidar place recognition in unstructured natural environments. *arXiv preprint arXiv:2211.12732*.

Komorowski, J. (2021). Minkloc3d: Point cloud based large-scale place recognition. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1790–1799.

Komorowski, J. (2022). Improving point cloud based place recognition with ranking-based loss and large batch training. In *2022 26th international conference on pattern recognition (ICPR)*, pages 3699–3705. IEEE.

Koniusz, P., Yan, F., Gosselin, P.-H., and Mikolajczyk, K. (2016). Higher-order occurrence pooling for bags-of-words: Visual concept detection. *IEEE transactions on pattern analysis and machine intelligence*, 39(2):313–326.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.

Kudo, T. (2018). Subword regularization: Improving neural network translation models with multiple subword candidates. *arXiv preprint arXiv:1804.10959*.

Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A., Alberti, C., Epstein, D., Polosukhin, I., Devlin, J., Lee, K., et al. (2019). Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Lague, D., Brodu, N., and Leroux, J. (2013). Accurate 3d comparison of complex topography with terrestrial laser scanner: Application to the rangitikei canyon (nz). *ISPRS journal of photogrammetry and remote sensing*, 82:10–26.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. (2019). Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551.

Lee, D.-T. and Wong, C.-K. (1977). Worst-case analysis for region and partial region searches in multidimensional binary search trees and balanced quad trees. *Acta Informatica*, 9(1):23–29.

Lee, S., Seong, H., Lee, S., and Kim, E. (2022). Correlation verification for image retrieval. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5374–5384.

Leordeanu, M. and Hebert, M. (2005). A spectral technique for correspondence problems using pairwise constraints. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 2, pages 1482–1489. IEEE.

Lepikhin, D., Lee, H., Xu, Y., Chen, D., Firat, O., Huang, Y., Krikun, M., Shazeer, N., and Chen, Z. (2020). Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*.

Li, J., Li, D., Savarese, S., and Hoi, S. (2023). Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR.

Liao, Y., Xie, J., and Geiger, A. (2022). Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3292–3310.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. (2017). Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019a). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Liu, Z., Zhou, S., Suo, C., Yin, P., Chen, W., Wang, H., Li, H., and Liu, Y.-H. (2019b). Lpd-net: 3d point cloud learning for large-scale place recognition and environment analysis. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2831–2840.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

Lowry, S., Sünderhauf, N., Newman, P., Leonard, J. J., Cox, D., Corke, P., and Milford, M. J. (2015). Visual place recognition: A survey. *ieee transactions on robotics*, 32(1):1–19.

Lyu, Z., Zhang, J., Lu, M., Li, Y., and Feng, C. (2024). Tell me where you are: Multimodal llms meet place recognition. *arXiv preprint arXiv:2406.17520*.

Maddern, W., Pascoe, G., Linegar, C., and Newman, P. (2017). 1 year, 1000 km: The oxford robotcar dataset. *The International Journal of Robotics Research*, 36(1):3–15.

Malkov, Y., Ponomarenko, A., Logvinov, A., and Krylov, V. (2014). Approximate nearest neighbor algorithm based on navigable small world graphs. *Information Systems*, 45:61–68.

Malkov, Y. A. and Yashunin, D. A. (2018). Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836.

Manning, C. D. (2008). *Introduction to information retrieval*. Syngress Publishing,.

McCloskey, M. and Cohen, N. J. (1989). Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Mehta, S. V., Gupta, J., Tay, Y., Dehghani, M., Tran, V. Q., Rao, J., Najork, M., Strubell, E., and Metzler, D. (2022). Dsi++: Updating transformer memory with new documents. *arXiv preprint arXiv:2212.09744*.

Mirzadeh, S. I., Farajtabar, M., Pascanu, R., and Ghasemzadeh, H. (2020). Understanding the role of training regimes in continual learning. *Advances in Neural Information Processing Systems*, 33:7308–7320.

Muja, M. and Lowe, D. G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP (1)*, 2(331-340):2.

Muqeeth, M., Liu, H., and Raffel, C. (2023). Soft merging of experts with adaptive routing. *arXiv preprint arXiv:2306.03745*.

Nakov, P., Corney, D., Hasanain, M., Alam, F., Elsayed, T., Barrón-Cedeño, A., Papotti, P., Shaar, S., and Martino, G. D. S. (2021). Automated fact-checking for assisting human fact-checkers. *arXiv preprint arXiv:2103.07769*.

Nene, S. A. and Nayar, S. K. (1996). Closest point search in high dimensions. In *Proceedings CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 859–865. IEEE.

Ni, J., Abrego, G. H., Constant, N., Ma, J., Hall, K. B., Cer, D., and Yang, Y. (2021). Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. *arXiv preprint arXiv:2108.08877*.

O'hara, S. and Draper, B. A. (2011). Introduction to the bag of features paradigm for image classification and retrieval. *arXiv preprint arXiv:1101.3354*.

Özyeşil, O., Voroninski, V., Basri, R., and Singer, A. (2017). A survey of structure from motion*. *Acta Numerica*, 26:305–364.

Pan, J., Wang, J., and Li, G. (2024). Survey of vector database management systems. *The International Journal on Very Large Data Bases*, 33.

Paparoditis, N., Papelard, J.-P., Cannelle, B., Devaux, A., Soheilian, B., David, N., and Houzay, E. (2012). Stereopolis ii: A multi-purpose and multi-sensor 3d mobile mapping system for street visualisation and 3d metrology. *Revue française de photogrammétrie et de télédétection*, 200(1):69–79.

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71.

Pion, N., Humenberger, M., Csurka, G., Cabon, Y., and Sattler, T. (2020). Benchmarking image retrieval for visual localization. In *2020 International Conference on 3D Vision (3DV)*, pages 483–494. IEEE.

Produit, T., Blanc, N., Composto, S., Ingensand, J., and Furhoff, L. (2018). Crowdsourcing the georeferencing of historical pictures.

Puigcerver, J., Riquelme, C., Mustafa, B., and Houlsby, N. (2023). From sparse to soft mixtures of experts. *arXiv preprint arXiv:2308.00951*.

Python Software Foundation (2024). Python wiki: Time complexity. https://wiki.python.org/moin/TimeComplexity.

Qader, W. A., Ameen, M. M., and Ahmed, B. I. (2019). An overview of bag of words; importance, implementation, applications, and challenges. In *2019 international engineering conference (IEC)*, pages 200–204. IEEE.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017a). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660.

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017b). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30.

Radenović, F., Tolias, G., and Chum, O. (2018). Fine-tuning cnn image retrieval with no human annotation. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1655–1668.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. (2021). Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. (2018). Improving language understanding by generative pre-training.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Robertson, S. E. and Jones, K. S. (1976). Relevance weighting of search terms. *Journal of the American Society for Information science*, 27(3):129–146.

Rosenblatt, F. et al. (1962). *Principles of neurodynamics: Perceptrons and the theory of brain mechanisms*, volume 55. Spartan books Washington, DC.

Rumelhart, D. E., Durbin, R., Golden, R., and Chauvin, Y. (2013). Backpropagation: The basic theory. In *Backpropagation*, pages 1–34. Psychology Press.

Rusu, R. B., Blodow, N., and Beetz, M. (2009). Fast point feature histograms (fpfh) for 3d registration. In *2009 IEEE international conference on robotics and automation*, pages 3212–3217. IEEE.

Rusu, R. B., Marton, Z. C., Blodow, N., and Beetz, M. (2008). Learning informative point classes for the acquisition of object model maps. In *2008 10th International Conference on Control, Automation, Robotics and Vision*, pages 643–650. IEEE.

Sabuncuoglu, I. and Bayiz, M. (1999). Job shop scheduling with beam search. *European Journal of Operational Research*, 118(2):390–412.

Salti, S., Tombari, F., and Di Stefano, L. (2014). Shot: Unique signatures of histograms for surface and texture description. *Computer vision and image understanding*, 125:251–264.

Salton, G. and Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019). Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Schuster, M. and Nakajima, K. (2012). Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE.

Sennrich, R., Haddow, B., and Birch, A. (2015). Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

Shi, P., Zhang, Y., and Li, J. (2023). Lidar-based place recognition for autonomous driving: A survey. *CoRR*.

Sipiran, I. and Bustos, B. (2010). A robust 3d interest points detector based on harris operator. In *3DOR@ Eurographics*, pages 7–14.

Sivic and Zisserman (2003). Video google: A text retrieval approach to object matching in videos. In *Proceedings ninth IEEE international conference on computer vision*, pages 1470–1477. IEEE.

Steder, B., Ruhnke, M., Grzonka, S., and Burgard, W. (2011). Place recognition in 3d scans using a combination of bag of words and point feature based relative pose estimation. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1249–1255. IEEE.

Stilla, U. and Xu, Y. (2023). Change detection of urban objects using 3d point clouds: A review. *ISPRS Journal of Photogrammetry and Remote Sensing*, 197:228–255.

Stoter, J., Vallet, B., Lithen, T., Pla, M., Wozniak, P., Kellenberger, T., Streilein, A., Ilves, R., and Ledoux, H. (2016). State-of-the-art of 3d national mapping in 2016. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 41:653–660.

Tan, H. and Bansal, M. (2019). Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.

Tang, T. Y., De Martini, D., and Newman, P. (2021). Get to the point: Learning lidar place recognition and metric localisation using overhead imagery. *Proceedings of Robotics: Science and Systems, 2021*.

Tay, Y., Dehghani, M., Rao, J., Fedus, W., Abnar, S., Chung, H. W., Narang, S., Yogatama, D., Vaswani, A., and Metzler, D. (2021). Scale efficiently: Insights from pre-training and fine-tuning transformers. *arXiv preprint arXiv:2109.10686*.

Tay, Y., Tran, V., Dehghani, M., Ni, J., Bahri, D., Mehta, H., Qin, Z., Hui, K., Zhao, Z., Gupta, J., et al. (2022). Transformer memory as a differentiable search index. *Advances in Neural Information Processing Systems*, 35:21831–21843.

Thomas, H., Qi, C. R., Deschaud, J.-E., Marcotegui, B., Goulette, F., and Guibas, L. J. (2019). Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6411–6420.

Thrun, S. (1995). Is learning the n-th thing any easier than learning the first? *Advances in neural information processing systems*, 8.

Tinchev, G., Penate-Sanchez, A., and Fallon, M. (2021). Skd: Keypoint detection for point clouds using saliency estimation. *IEEE Robotics and Automation Letters*, 6(2):3785–3792.

Tolias, G., Sicre, R., and Jégou, H. (2015). Particular object retrieval with integral max-pooling of cnn activations. *arXiv preprint arXiv:1511.05879*.

Toneva, M., Sordoni, A., Combes, R. T. d., Trischler, A., Bengio, Y., and Gordon, G. J. (2018). An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*.

Trotman, A., Puurula, A., and Burgess, B. (2014). Improvements to bm25 and language models examined. In *Proceedings of the 19th Australasian Document Computing Symposium*, pages 58–65.

Ullman, S. (1979). The interpretation of structure from motion. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 203(1153):405–426.

Uy, M. A. and Lee, G. H. (2018). Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4470–4479.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Vidanapathirana, K., Moghadam, P., Harwood, B., Zhao, M., Sridharan, S., and Fookes, C. (2021). Locus: Lidar-based place recognition using spatiotemporal higher-order pooling. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5075–5081. IEEE.

Vidanapathirana, K., Moghadam, P., Sridharan, S., and Fookes, C. (2023). Spectral geometric verification: Re-ranking point cloud retrieval for metric localization. *IEEE Robotics and Automation Letters*, 8(5):2494–2501.

Vidanapathirana, K., Ramezani, M., Moghadam, P., Sridharan, S., and Fookes, C. (2022). Logg3d-net: Locally guided global descriptor learning for 3d place recognition. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2215–2221. IEEE.

Wang, J., Yang, Z., Hu, X., Li, L., Lin, K., Gan, Z., Liu, Z., Liu, C., and Wang, L. (2022). Git: A generative image-to-text transformer for vision and language. *arXiv preprint arXiv:2205.14100*.

Wang, L., Groves, P. D., and Ziebart, M. K. (2012). Multi-constellation gnss performance evaluation for urban canyons using large virtual reality city models. *The Journal of Navigation*, 65(3):459–476.

Xu, T.-X., Guo, Y.-C., Li, Z., Yu, G., Lai, Y.-K., and Zhang, S.-H. (2021). Transloc3d: Point cloud based large-scale place recognition using adaptive receptive fields. *arXiv preprint arXiv:2105.11605*.

Yang, H., He, T., Liu, J., Chen, H., Wu, B., Lin, B., He, X., and Ouyang, W. (2023). Gd-mae: generative decoder for mae pre-training on lidar point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9403–9414.

Yang, J., Li, C., Zhang, P., Xiao, B., Liu, C., Yuan, L., and Gao, J. (2022). Unified contrastive learning in image-text-label space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19163–19173.

Yin, P., Jiao, J., Zhao, S., Xu, L., Huang, G., Choset, H., Scherer, S., and Han, J. (2025). General place recognition survey: Towards real-world autonomy. *IEEE Transactions on Robotics*.

Yin, P., Zhao, S., Ge, R., Cisneros, I., Fu, R., Zhang, J., Choset, H., and Scherer, S. (2022). Alita: A large-scale incremental dataset for long-term autonomy. *arXiv preprint arXiv:2205.10737*.

Yuan, L., Chen, D., Chen, Y.-L., Codella, N., Dai, X., Gao, J., Hu, H., Huang, X., Li, B., Li, C., et al. (2021). Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*.

Zeng, H., Luo, C., Jin, B., Sarwar, S. M., Wei, T., and Zamani, H. (2024). Scalable and effective generative information retrieval. In *Proceedings of the ACM Web Conference 2024*, pages 1441–1452.

Zhang, W. and Xiao, C. (2019). Pcan: 3d attention map learning using contextual information for point cloud based retrieval. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12436–12445.

Zhang, Y., Shi, P., and Li, J. (2024). Lidar-based place recognition for autonomous driving: A survey. *ACM Computing Surveys*, 57(4):1–36.

Zhong, Z., Zheng, L., Cao, D., and Li, S. (2017). Re-ranking person re-identification with k-reciprocal encoding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1318–1327.

Zhuang, S., Ren, H., Shou, L., Pei, J., Gong, M., Zuccon, G., and Jiang, D. (2022). Bridging the gap between indexing and retrieval for differentiable search index with query generation. *arXiv preprint arXiv:2206.10128*.

Résumé détaillé de la thèse en français

Reconnaissance de lieux à grande échelle à partir de nuages de points

Introduction

Cette thèse s'inscrit dans le cadre de l'indexation et de l'efficacité de la recherche de nuages de points 3D, dans un contexte de reconnaissance de lieux à grande échelle. Dans cette étude, nous nous sommes concentrés sur la reconnaissance de nuages de points au sein d'une base de données géoréférencées. L'appariement entre un nuage de points d'une requête et les plus proches voisins, ordonnés par score de similarité, parmi les références disponibles, permet d'estimer sa position.

Les techniques de reconnaissance de lieux présentent de nombreuses applications, telles que la navigation des systèmes autonomes, la correction du positionnement, la détection de changements dans les jumeaux numériques, ou encore la lutte contre la désinformation.

La reconnaissance de lieux se décompose en deux étapes, avec d'abord l'indexation, au cours de laquelle une représentation implicite de la géométrie des scènes, appelée descripteur, est calculée pour chaque nuage de points de référence. La seconde étape, la recherche (*retrieval*), consiste à calculer le descripteur du nuage de points de la requête, puis à le comparer à ceux de la base de données. Par une recherche des plus proches voisins, il est ainsi possible d'identifier le ou les nuages de points de référence les plus similaires à la requête. Cependant, les travaux existants se concentrent majoritairement sur la conception de descripteurs robustes et discriminants, négligeant souvent la phase de recherche. Celle-ci repose encore fréquemment sur des approches exhaustives ou mal adaptées au format des descripteurs considérés.

Les travaux de cette thèse se sont ainsi orientés vers l'accélération de la recherche de nuages de points, notamment par l'adaptation de méthodes issues de la recherche d'information générative, initialement développées pour des données textuelles (Tay et al., 2022). Les objectifs de ce travail se résument comme suit :

- proposer une nouvelle approche s'appuyant uniquement sur les informations géométriques contenues dans un nuage de points de requête, sans recourir à des modalités auxiliaires;
- accélérer la phase de recherche par rapport aux méthodes actuelles de reconnaissance de lieux 3D;
- exploiter les données LiDAR HD [1] à des fins de reconnaissance de lieux à grande

[1] Programme d'acquisition LiDAR 3D à haute densité du territoire français, menés par l'IGN.

échelle, en adaptant la couverture, la densité et les scénarios de revisite;

- concevoir une architecture de recherche hiérarchique, capable de traiter des jeux de données plus grands et adaptée à la mise à jour des modèles.

Les travaux présentés dans cette thèse sont organisés en six chapitres, structurés comme suit :

le chapitre 1 introduit la motivation et le contexte de cette recherche, ainsi que la problématique de la reconnaissance de lieux 3D à grande échelle;

le chapitre 2 propose un état de l'art de la reconnaissance de lieux basée sur les nuages de points, en retraçant l'évolution des approches, depuis les méthodes fondées sur les descripteurs locaux jusqu'aux techniques d'apprentissage profond. Ce chapitre présente également les méthodes d'accélération de la recherche, notamment la recherche approximative des plus proches voisins, ainsi que les avancées en recherche d'information qui ont inspiré la méthodologie proposée dans ce travail;

le chapitre 3 décrit notre première contribution, qui consiste en l'adaptation de l'index de recherche différentiable (DSI) (Tay et al., 2022) aux données 3D. Cette partie inclut la conception de nouvelles stratégies d'indexation adaptées aux nuages de points, ainsi que l'adaptation de transformers de vision servant de lien entre la 3D et le texte;

le chapitre 4 évalue l'application de notre première contribution sur un jeu de données de cartographie mobile. Ce chapitre introduit aussi les choix de conception utilisés pour l'entraînement et l'évaluation de DSI-3D, et s'ouvre sur une étude de l'impact de la variation des hyperparamètres.

le chapitre 5 présente l'adaptation du jeu de données LiDAR haute densité fourni par l'IGN en un jeu de données de référence pour la reconnaissance de lieux à grande échelle. Ce changement de base de données implique aussi un réentraînement de la méthode d'encodage à cette nouvelle modalité. Le nouvel encodage a été validé dans deux zones parisiennes;

le chapitre 6 explore plusieurs stratégies d'amélioration de l'applicabilité de DSI-3D aux jeux de données à grande échelle. Une première extension repose sur l'intégration d'une architecture de type mixture d'experts (MoE) (Jacobs et al., 1991; Jordan and Jacobs, 1994), permettant de partitionner l'espace de recherche en experts spécialisés formés sur différentes régions afin d'améliorer l'évolutivité et la couverture géographique. Une seconde direction vise à renforcer la robustesse de DSI-3D en intégrant des stratégies de reclassement (*re-ranking*);

le chapitre 7 conclut cette thèse en résumant les principales contributions, en discutant des limites des approches proposées et en ouvrant des perspectives pour de futurs travaux de recherche.

Chapitre 2 : État de l'art

Ce chapitre présente l'état de l'art couvrant trois domaines : la reconnaissance de lieux à partir de nuages de points, l'accélération des méthodes de recherche et la recherche d'informations.

Reconnaissance de lieux 3D

La reconnaissance de lieux 3D a d'abord émergé par l'adaptation de descripteurs locaux issus des méthodes de recherche d'images par contenu visuel à la géométrie 3D, notamment pour l'appariement de surfaces et l'alignement de nuages de points. Par la suite, les scènes 3D ont été représentées à l'aide de descripteurs globaux, obtenus par agrégation des caractéristiques locales, ce qui a permis de simplifier l'estimation de la similarité entre deux nuages de points. L'apprentissage profond a ensuite été progressivement intégré à ce domaine, d'abord sous la forme de réseaux convolutifs 3D, puis par l'utilisation de transformers (Vaswani et al., 2017), conduisant à des approches plus performantes. Dans la suite de ce travail, nous avons utilisé LoGG3D-Net (Vidanapathirana et al., 2022) comme encodeur de nuages de points, servant à la fois de base pour notre modèle et de référence pour les méthodes comparatives.

Méthodes d'accélération de la recherche

Dans la littérature de la vision par ordinateur 3D, une limite des différentes méthodes de reconnaissance de lieux est la concentration des avancées sur la phase d'indexation, au détriment de la phase de recherche, souvent mise en œuvre à l'aide d'une simple recherche exhaustive ou, au mieux, avec l'aide d'une structure de type arbre k-D (Bentley, 1975). La recherche exhaustive présente un coût computationnel élevé, augmentant linéairement avec la taille de la base de données, tandis que la méthode des arbres k-D devient inefficace pour les vecteurs de grande dimension (Kibriya and Frank, 2007), ce qui est le cas dans notre contexte. Ce chapitre introduit des méthodes de recherche approximative issues de l'état de l'art des bases de données, reposant notamment sur les tables de hachage (LSH) (Indyk and Motwani, 1998; Douze et al., 2024) et sur des structures de graphe (HNSW) (Malkov and Yashunin, 2018) . Ces approches permettent de sacrifier une partie de la précision des résultats afin d'obtenir une complexité de recherche sous-linéaire. Elles serviront de méthodes de comparaison pour l'évaluation de notre approche.

La recherche d'information textuelle

Enfin, ce chapitre présente également les méthodes de recherche d'information, conçues pour parcourir de très grands ensembles de données, similaires à ceux exploités par les moteurs de recherche. Les premières approches, telles que TF-IDF (Salton and Buckley, 1988) et BM25 (Robertson and Jones, 1976; Trotman et al., 2014), reposent sur une représentation dite «sac de mots», fondée sur la fréquence d'apparition des termes dans un document et leur rareté dans le corpus. Plus récemment, les encodeurs doubles (*dual encoders*) ont été introduits : ils apprennent à associer à chaque document une représentation vectorielle discriminante, permettant d'effectuer la recherche par proximité dans l'espace latent. Les DSI (Tay et al., 2022) constituent une évolution plus récente : ils apprennent, via un transformer, à associer directement une requête textuelle à un identifiant de document (docid). Cette approche permet de réaliser la recherche d'information de manière générative, et c'est cette idée que nous explorons dans le cadre de cette thèse.

Ces constatations ont motivé l'exploration d'une nouvelle approche de recherche générative, DSI-3D, inspirée de la recherche d'information textuelle et appliquée aux données 3D.

Chapitre 3 : DSI-3D, l'adaptation de DSI aux données 3D

À partir des conclusions de l'état de l'art, ce chapitre présente la principale contribution de la thèse, le modèle DSI-3D, qui est une adaptation du DSI à des données 3D afin d'appliquer les concepts de recherche d'information à la reconnaissance de lieux. Afin d'utiliser l'architecture DSI, deux adaptations majeures ont été nécessaires.

Adaptation des transformers de vision

La première concerne la conversion des nuages de points en textes. Pour cela, les transformers de vision ont été utilisés, capables de générer des descriptions textuelles à partir d'images. Notre choix s'est porté sur le modèle GIT (*Generative Image-to-Text*), en raison de son architecture composée d'un encodeur d'image et d'un décodeur de texte. En remplaçant l'encodeur d'image par LoGG3D-Net, nous établissons ainsi une association entre la représentation d'un nuage de points et son identifiant de document, comme illustré en Figure B.1.

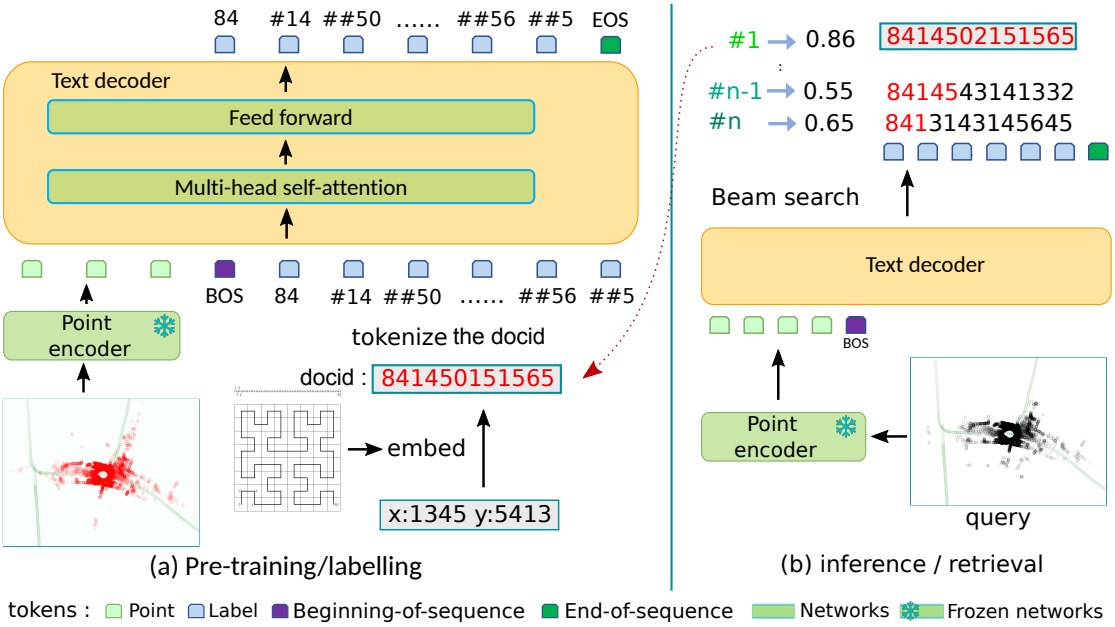


Figure B.1: Étapes de d'indexation et de recherche du DSI, pour des données 3D. (a) L'indexation : pour chaque nuage de points (en rouge), un descripteur est calculé par un encodeur de points gelé (en vert), ici LoGG3D-Net. Ce descripteur est associé à l'index (docid) et que décodeur de texte (en jaune) apprend à générer en fonction du nuage de points encodé. (b) Recherche : pour un nuage de points requête (en noir), une recherche par faisceau est effectuée par le décodeur de texte pour fournir les n solutions les plus probables. Figure modifiée d'après l'article sur la méthode GIT (Wang et al., 2022).

L'indexation de nuage de points

La seconde adaptation porte sur la méthode d'indexation des docids. Le jeu de données de scans choisi est KITTI (Geiger et al., 2013) sur lequel LoGG3D-Net a été pré-entraîné. Étant donné que le modèle DSI effectue une recherche par faisceaux (*beam search*) (Sabuncuoglu and Bayiz, 1999) durant la phase de génération, il est important que des docids correspondant à des nuages géographiquement proches partagent des séquences de tokens similaires afin d'éviter d'éliminer des candidats valides. Le modèle DSI initial propose

plusieurs stratégies, dont une approche naïve, consistant à utiliser des entiers naturels croissants, par exemple en utilisant les index des fichiers KITTI, et une approche hiérarchique, fondée sur un regroupement sémantique des représentations, où les groupes sont obtenus par K-moyennes, et leurs préfixes servent à construire les docids. En complément, nous avons proposé deux nouvelles stratégies de construction de docids, basées sur la position géographique du nuage de points. La première est basée sur l'entrelacement des coordonnées GNSS du centre du nuage, et la seconde repose sur la distance parcourue par une courbe de Hilbert dans l'espace 2D pour atteindre les coordonnées GNSS du centre du nuage.

Chapitre 4 : Évaluation de DSI-3D

Ce chapitre évalue l'approche DSI-3D sur un jeu de données issu du mobile mapping, ce qui a nécessité d'adapter le protocole d'évaluation afin de pouvoir se comparer aux approches classiques.

Présentation du dataset et comparatif des performances

Une présentation détaillée du jeu de données KITTI est faite. Nous y définissons la méthode de classification des réponses, ainsi que la construction d'une séquence 22, regroupant l'ensemble des nuages de points issus des six autres séquences du jeu de données. La méthode d'évaluation a dû être adaptée afin de diviser le jeu de données de manière à ce que les jeux de validation et d'évaluation puissent comporter des revisites positives présentes dans le jeu d'entraînement. Cette contrainte découle du fait que DSI-3D ne peut retourner que des indices correspondant aux éléments sur lesquels il a été préalablement entraîné. Les tables B.3 et B.4 résument les résultats obtenus lors de ces évaluations.

Table B.3: Évaluation sur plusieurs séquences KITTI à l'aide de diverses stratégies d'indexation, basées sur les descripteurs LoGG3D-Net, à l'aide de la métrique $F1_{max}$.

Indexing	**00**	**02**	**05**	**06**	**07**	**08**	**22**
Recherche exhaustive	**0.9956**	**0.9859**	**1.0**	**0.9955**	**1.0**	**0.9975**	**0.9925**
LSH_{256}	0.9945	0.9792	0.9946	0.9815	0.9626	0.9888	0.9850
$HNSW_{32}$	**0.9956**	**0.9859**	**1.0**	**0.9955**	**1.0**	**0.9975**	**0.9925**
DSI-3D avec Label	0.8991	0.8774	0.8831	0.7954	0.8844	0.9044	/
DSI-3D avec Hierar	0.9647	0.9316	0.9684	0.9626	0.9817	0.9684	/
DSI-3D avec GNSS	0.9809	0.9758	0.9891	0.9674	0.9955	0.9851	/
DSI-3D avec Hilbert	0.9933	0.9781	0.9909	0.9817	0.9955	0.9913	0.9819

Table B.4: Évaluation sur plusieurs séquences KITTI à l'aide de diverses stratégies d'indexation, basées sur les descripteurs LoGG3D-Net, à l'aide de la métrique *Hits*@1.

Indexing	**00**	**02**	**05**	**06**	**07**	**08**	**22**
Exhaustive k-NN	**0.9956**	**0.9722**	**1.0**	**0.9910**	**1.0**	**0.9950**	**0.9852**
LSH_{256}	0.9890	0.9572	0.9892	0.9550	0.9820	0.9755	0.9693
$HNSW_{32}$	**0.9912**	**0.9722**	**1.0**	**0.9910**	**1.0**	**0.9950**	**0.9852**
DSI-3D avec Label	0.8132	0.7645	0.7906	0.6306	0.7928	0.8235	/
DSI-3D avec Hierar	0.9319	0.8630	0.9387	0.9279	0.9640	0.9387	/
DSI-3D avec GNSS	0.9626	0.9507	0.9783	0.9369	0.9909	0.9705	/
DSI-3D avec Hilbert	0.9846	0.9550	0.9819	0.9640	0.9910	0.9828	0.9644

Les résultats montrent que la construction des docids à l'aide de la courbe de Hilbert offre les meilleurs résultats dans un contexte de reconnaissance de lieux, où la proximité spatiale entre la prédiction et la vérité terrain est valorisée. On remarque aussi que la seconde méthode d'indexation de position, avec le GNSS, retourne les deuxièmes meilleurs résultats dans un contexte de reconnaissance de lieux où la proximité géographique de la réponse est valorisée. Parmi les méthodes d'accélération de la recherche, la méthode basée sur les graphes, ici HNSW, parvient à préserver des performances plus proches de celles obtenues par une recherche exhaustive. En revanche, DSI-3D atteint des performances comparables à celles obtenues avec une méthode de hachage telle que LSH. La table B.5 présente les temps d'inférence moyen pour chaque séquence KITTI.

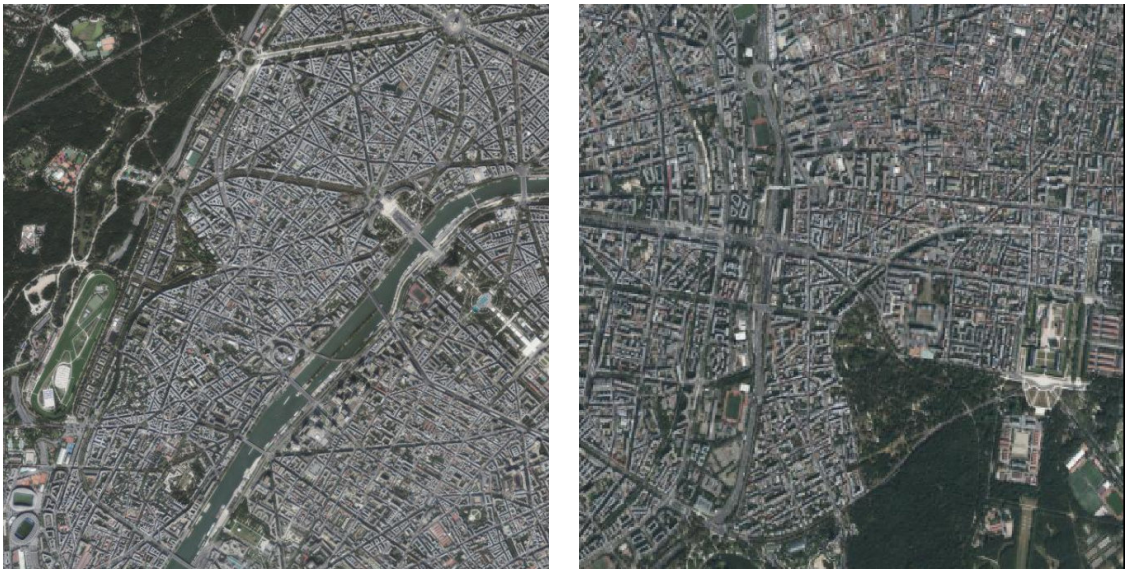
Table B.5: Comparatif des temps de recherche (en ms), pour obtenir 10 candidats, selon différentes méthodes (exhaustive, arbre k-d, LSH, HNSW, DSI-3D) et pour plusieurs séquences avec leurs tailles de base de données.

Sequence	**00**	**02**	**05**	**06**	**07**	**08**	**22**
Taille de la database	3632	3728	2208	880	880	3256	14584
Recherche exhaustive	2.6	2.7	1.6	0.7	0.7	2.4	10.9
Arbre k-d	2.6	2.6	1.6	0.7	0.7	2.3	9.8
LSH_{256}	0.2	0.3	0.2	0.2	0.1	0.3	0.4
$HNSW_{32}$	0.3	0.4	0.3	0.2	0.2	0.3	0.4
DSI-3D label	62.1	63.0	63.2	63.4	65.5	61.5	/
DSI-3D GNSS	92.1	93.5	90.7	95.0	96.8	93.5	/
DSI-3D Hierar	59.7	55.4	54.2	52.1	58.6	59.9	/
DSI-3D Hilbert	93.3	94.5	90.4	86.6	91.7	89.8	95.4

Même si, pour ce jeu de données, les méthodes d'accélération de l'état de l'art demeurent plus rapides, l'un des avantages de DSI-3D réside dans le fait que sa complexité de recherche est indépendante de la taille de la base de données. En effet, cette charge est transférée à la phase d'entraînement, contrairement aux méthodes LSH et HNSW, dont la complexité, bien que sous-linéaire, augmente avec la taille du jeu de données. Après avoir validé le concept de DSI-3D sur le jeu de données KITTI, ses performances sont évaluées sur un ensemble de données plus vaste. Nous nous sommes donc tournés vers le LiDAR HD, couvrant bientôt l'intégralité du territoire français.

Chapitre 5 : LiDAR HD

Les données LiDAR haute densité (LiDAR HD) sont des acquisitions 3D de l'IGN représentant la surface du territoire français, distribuées par tuiles de 1 km^2. L'objectif de ce chapitre est d'adapter ce jeu de données en vue de son utilisation pour la reconnaissance de lieux à l'échelle nationale. Deux zones situées à Paris (Ouest et Est), contenant chacune 16 et 9 tuiles LiDAR, ont été sélectionnées pour cette étude, comme illustré à la Figure B.2.

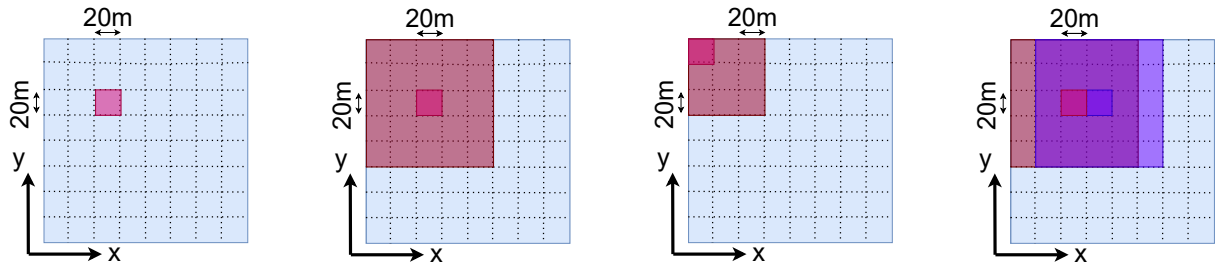


(a) Paris Ouest (16km^2) (b) Paris Est (9km^2)

Figure B.2: Vue aérienne de la couverture spatiale des deux zones sélectionnées de Paris.

Adaptation du LiDAR HD

Ces tuiles ne peuvent pas être utilisées directement pour la reconnaissance de lieux. Il a fallu définir un pas spatial plus fin, c'est pourquoi le découpage spatial des tuiles LiDAR s'effectue selon une grille régulière, avec une taille de cellule de 20 x 20 m (Figure B.3a). Chaque cellule se combine avec ses voisines pour former un nuage de points de 100 x 100 m (Figure B.3a). Les cas limites, dus aux bords des tuiles ou à l'absence de données, sont exclus du jeu de données final (Figure B.3c). L'inclusion de zones de chevauchement entre nuages de points voisins garantit la continuité spatiale des descripteurs (Figure B.3d).



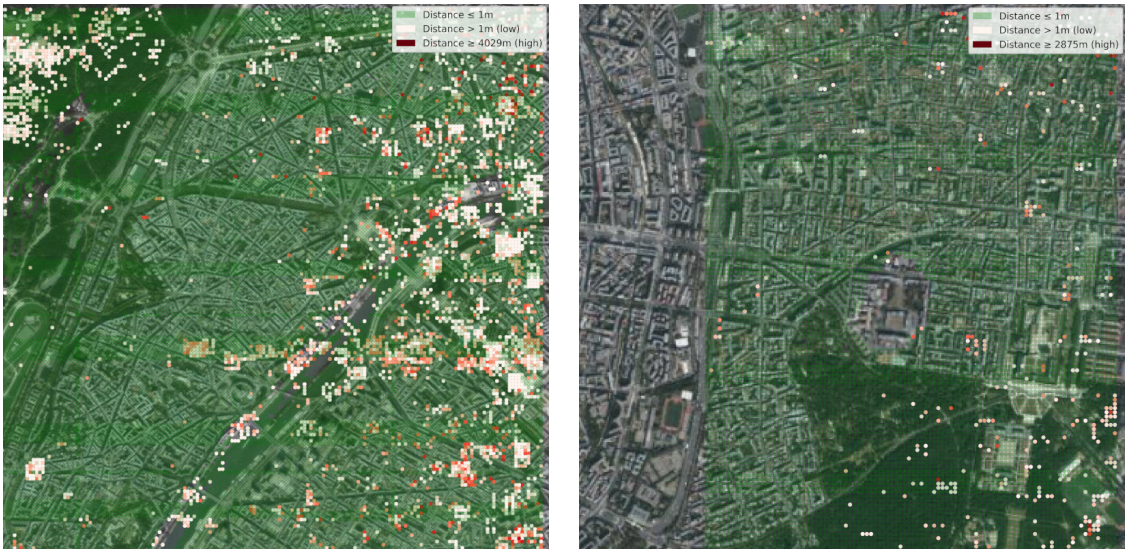
(a) Cellule (b) Nuage de points (c) Cas limites (d) Chevauchement

Figure B.3: Génération de nuages de points à partir de tuiles LiDAR HD. (a) Les tuiles (bleues) sont divisées en cellules de grille de 20 × 20 m (rose). (b) Les cellules voisines partageant le même identifiant d'acquisition sont agrégées pour former un nuage de points de 100 × 100 m. (c) Exemple de cas limites. (d) Le processus (b) est répété pour chaque cellule.

La définition des revisites doit également être mise à jour pour ce jeu de données. Chaque tuile de 1 km^2 a été observée lors de plusieurs acquisitions aériennes, que l'on peut distinguer par leur identifiant de vol. En découpant les tuiles en cellules, nous avons regroupé uniquement les cellules d'un même identifiant de vol pour construire le nuage de points. Cela permet d'obtenir plusieurs observations d'un même lieu, avec différents angles d'acquisition, que l'on considère ici comme des revisites.

Validation des descripteurs de LoGG3D-Net sur le LiDAR HD

Après préparation des données, LoGG3D-Net a été réentraîné sur ce nouveau jeu de données, les modèles pré-entraînés sur KITTI ne se généralisant pas au LiDAR aérien. Les trois dalles verticales à l'ouest de la zone Est ont été utilisées comme jeu d'entraînement, et le reste des dalles pour l'évaluation. De nombreuses configurations ont été explorées, notamment une approche multi-échelle visant à mieux capturer la variabilité spatiale des scènes. L'entraînement a conduit à la production de descripteurs discriminants à échelle fixe, dont les performances ont été validées sur les deux zones sélectionnées. Les résultats de la recherche exhaustive sont visibles sur la Figure B.4, où, en vert, figurent les cas où la réponse renvoyée est la même scène que la requête, sous un identifiant de vol différent. Le gradient de couleur allant du blanc au rouge foncé représente une erreur de distance de plus en plus significative.



(a) Paris Ouest (vue agrandie) (b) Paris Est (vue agrandie)

Figure B.4: Vues agrandies de la répartition des erreurs de recherche lors d'une recherche exhaustive, avec les descripteurs obtenus par le modèle réentraîné, sur les deux zones de Paris. Le vert représente la correspondance exacte (distance = 0 m), tandis que le dégradé de rouge indique l'erreur de distance, du blanc (faible erreur) au rouge foncé (grande distance).

La table B.6 présente les performances d'une recherche exhaustive.

Zone	Paris ouest	Paris est
$F1_{max}$	0.9616	0.9909
$Hits@N$	0.9230	0.9816

Table B.6: Résultat de la recherche exhaustive des descripteurs de LoGG3D-Net réentraîné sur le LiDAR HD, évaluées sur la zone ouest de Paris.

Les résultats montrent que les scores de la recherche exhaustive sont meilleurs pour la partie Est que pour la partie Ouest, ce qui peut s'expliquer par le fait que la zone d'entraînement partage davantage de similarité avec la zone Est en raison de sa proximité. Les scores demeurent néanmoins suffisamment élevés pour que l'on puisse les utiliser pour entraîner nos modèles DSI-3D.

Chapitre 6 : Extension de DSI-3D

Ce chapitre propose, à partir des enseignements des deux chapitres précédents, de définir plusieurs stratégies d'indexation et de recherche dans les nuages 3D pour la reconnaissance de lieux : d'une part, une extension du modèle DSI-3D aux mixtures d'experts, et d'autre part, l'amélioration des résultats de la recherche par reclassement géométrique.

Mixture d'experts appliqué à la reconnaissance de lieux

La première extension concerne l'implémentation d'une architecture de type MoE, dans laquelle chaque expert correspond à un modèle DSI-3D entraîné sur une zone géographique spécifique. Cette activation peut être restreinte aux K experts les plus pertinents, ou bien étendue à l'ensemble des experts, chacun contribuant à la réponse finale. La réponse retournée par le MoE est alors obtenue comme une combinaison linéaire des sorties des

experts activés, pondérées par les scores d'activation calculés par le routeur, comme illustré à la Figure B.5.

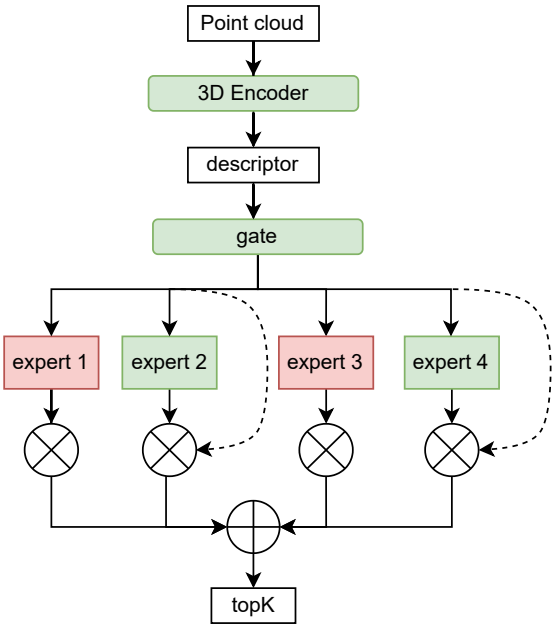
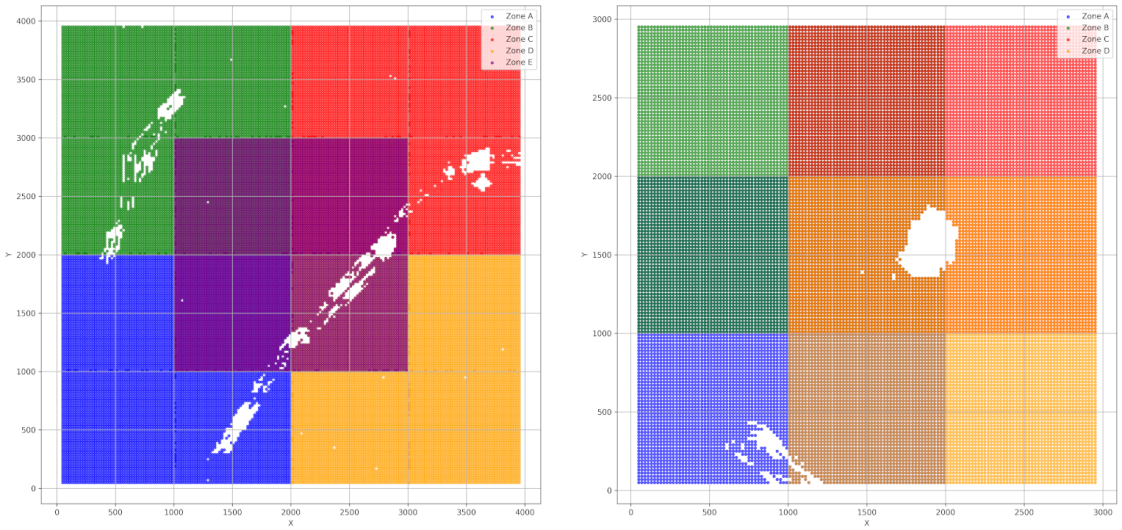


Figure B.5: Architecture proposée de mixture d'experts pour la recherche de scènes 3D. Le routeur sélectionne et pondère un sous-ensemble d'experts (en vert) en fonction du descripteur de la requête. Les experts en rouge ne sont pas activés.

Une distinction importante par rapport aux architectures MoE classiques réside dans la nature des experts. Dans notre cas, chaque expert correspond à un modèle transformer complet, déjà entraîné et figé, et non à un réseau de neurones à propagation directe (*Feed-forward Neural Network*, FFN) à l'intérieur d'un transformer. Le mécanisme de routage est donc externalisé, c'est-à-dire appliqué en dehors du modèle DSI-3D, afin de sélectionner et de combiner les experts pertinents au moment de l'inférence.

Le dataset de chaque zone a été distribué de manière égale entre quatre et cinq sous-ensembles sur lesquels les modèles DSI-3D ont été entraînés. Le chevauchement des zones permet à plusieurs experts d'apporter une contribution pertinente lorsque l'on active plus d'un expert. La figure B.6 illustre la répartition spatiale de ces zones : cinq couvrent l'ouest de Paris et quatre couvrent l'est.



(a) Distribution des experts de Paris Ouest (b) Distribution des experts de Paris Est

Figure B.6: Répartition des experts dans les deux zones de Paris. Les experts sont colorés, dans l'ordre, en bleu, vert, rouge, jaune (et magenta pour Paris Ouest). Les couleurs supplémentaires résultent de la synthèse additive des couleurs des experts, représentant les zones de chevauchements.

Une fois que les experts des deux zones ont été entraînés, les routeurs ont, à leur tour, été entraînés en tant que classifieurs multi-labels afin de déterminer les activations des experts.

L'approche MoE a ensuite été comparée aux méthodes classiques, ainsi qu'à une version de DSI-3D entraînée séparément dans chaque zone. Cette évaluation a été faite à la fois sur les deux zones $Paris_{Est}$ et $Paris_{Ouest}$, ainsi que sur leur combinaison $Paris_{Est+Ouest}$. Les résultats correspondants sont présentés dans les tableaux B.7 et B.8.

Enfin, l'approche DSI-3D avec MoE n'a pas été appliquée à $Paris_{Est+Ouest}$ avec un MoE top-2 ou dense, car le gain attendu des experts complémentaires était marginal d'après les observations des deux premières zones, et l'activation simultanée d'experts issus de deux zones distinctes ne pouvait raisonnablement pas contribuer à améliorer les performances.

Table B.7: Comparaison de la recherche de données LiDAR HD à l'aide de la métrique $Hits$@1, avec des méthodes de l'état de l'art, DSI-3D et ses variantes MoE.

Zone Taille de la database	$Paris_{Est}$ 20892	$Paris_{Ouest}$ 37710	$Paris_{Est+Ouest}$ 58602
Recherche exhaustive	0.9849	0.9195	0.9491
Arbre k-D	0.9849	0.9195	0.9491
LSH_{256}	0.8643	0.7143	0.7500
$HNSW_{32}$	0.9738	0.8940	0.9209
DSI-3D	0.6493	0.2398	/
DSI-3D avec MoE top 1	0.8281	0.7436	0.7413
DSI-3D avec MoE top 2	0.8285	0.7438	/
DSI-3D avec MoE dense	0.8287	0.7438	/

Table B.8: Comparaison de la recherche de données LiDAR HD à l'aide de la métrique $F1_{max}$, avec des méthodes de l'état de l'art, DSI-3D et ses variantes MoE.

Zone	$Paris_{Est}$	$Paris_{Ouest}$	$Paris_{Est+Ouest}$
Taille de la database	20892	37710	58602
Recherche exhaustive	0.9927	0.9593	0.9748
Arbre k-D	0.9927	0.9593	0.9748
LSH_{256}	0.9888	0.9300	0.9540
$HNSW_{32}$	0.9900	0.9524	0.9668
DSI-3D	0.8438	0.8271	/
DSI-3D avec MoE top 1	0.8534	0.8207	0.8120
DSI-3D avec MoE top 2	0.8572	0.8219	/
DSI-3D avec MoE dense	0.8594	0.8226	/

L'approche MoE démontre un intérêt marqué par rapport à l'entraînement d'un modèle unique sur l'ensemble de la zone. En décomposant la tâche en zones géographiques et en spécialisant les experts, le MoE améliore systématiquement les performances en précision ($Hits@1$ et $F1_{max}$) tout en maintenant un coût d'entraînement global identique. Cette répartition régulière facilite la transmission des hyperparamètres d'entraînement entre experts, couvrant une surface et un nombre de scènes comparables, tandis que les modèles DSI-3D nécessitent davantage de réglages. La table B.9 présente les temps d'inférence des méthodes de l'état de l'art, de DSI-3D et de ses variantes MoE.

Table B.9: Comparatif des temps de recherche (en ms) d'un Top-10, selon différentes méthodes (exhaustive, arbre k-d, LSH, HNSW, DSI-3D et MoE+DSI-3D), sur les zones du LiDAR parisiens.

Zone	$Paris_{Est}$	$Paris_{Ouest}$	$Paris_{Est+Ouest}$
Taille de la database	20892	37710	58602
Recherche exhaustive	9.9	20.2	32.5
Arbre k-d	11.6	16.5	25.9
LSH_{256}	0.7	1.1	1.7
$HNSW_{32}$	0.6	0.4	0.5
DSI-3D	57.5	59.8	/
DSI-3D avec MoE top 1	85.3	91.0	89.6
DSI-3D avec MoE top 2	114.7	121.4	/
DSI-3D avec MoE dense	398.2	534.0	/

Les méthodes d'accélération de l'état de l'art demeurent plus rapides que DSI-3D pour cette taille de jeu de données. En revanche, à l'exception de l'arbre k-D, toutes ces méthodes voient leurs performances diminuer avec l'augmentation de la taille de la base, ce qui n'a pas pu être observé avec les séquences KITTI. Le temps d'inférence de DSI-3D avec un MoE est plus lent que celui de sa version classique en raison de l'ajout du temps d'inférence du routeur. L'augmentation du temps d'inférence, avec une activation croissante des experts, est due à l'inférence séquentielle des experts. Une inférence parallélisée des experts activés pourrait réduire le temps d'inférence moyen à celui d'un expert.

Reclassement des candidats

La seconde extension de DSI-3D porte sur une étude complémentaire consacrée aux stratégies de reclassement des plus proches voisins, à partir d'informations géométriques. Ce travail a été réalisé dans le cadre d'un stage de fin d'études (Deneuville, 2025), afin d'explorer des méthodes de reclassement (*re-ranking*) des résultats de recherche basées sur la cohérence spatiale locale. Cette étude s'est consacrée au reclassement de candidats obtenus par une recherche exhaustive, mais elle peut être transposée à une liste de candidats obtenus par la recherche de faisceaux de DSI-3D.

Une première approche a été d'appliquer une solution de l'état de l'art, telle que SpectralGV (Vidanapathirana et al., 2023), utilisant les descripteurs locaux des nuages pour construire des graphes de similarité. Ici, une première liste de candidats est obtenue après une première phase de recherche des k plus proches voisins, basée sur les descripteurs globaux. Ensuite, l'ensemble des descripteurs locaux de la requête est comparé à chaque ensemble de descripteurs locaux des candidats afin de construire un graphe de similarité, exploitant la similarité spatiale entre les points et leurs voisins. La correspondance des ensembles est réalisée à l'aide de *Spectral Matching* (Leordeanu and Hebert, 2005) pour trouver la combinaison la plus probable.

Pour tester la capacité de reclassement face à une requête inconnue, l'orientation du nuage de points ne doit pas être connue. Cela a révélé une limite de SpectralGV en appliquant des rotations aléatoires selon l'axe Z. Seul un sous-ensemble de points (15000 points) est utilisé pour limiter les coûts d'appariement d'ensembles, et leur sélection est affectée par l'orientation du nuage.

Un second état de l'art a été réalisé afin d'étudier les approches de détection de points d'appui (*keypoints*) pour rendre SpectralGV plus robuste aux rotations. Cette étude nous a mené vers D3Feat (Bai et al., 2020) afin de détecter les points saillants des scènes du LiDAR HD. D3Feat est un modèle qui a été entraîné à détecter et à encoder les points saillants à partir de ses propres descripteurs locaux. Leur encodage est fait avec KPconv (Thomas et al., 2019)

La Figure B.7 présente les performances de recherche, mesurées par le score $Hits@1$ (en %) pour des requêtes orientées d'un angle $\theta, \theta \in [0, 360]$ (en degrés). Pour cette évaluation, les descripteurs utilisés ont été entraînés avec une taille de voxel de 0,1 m, sans augmentation de rotation pendant l'apprentissage.

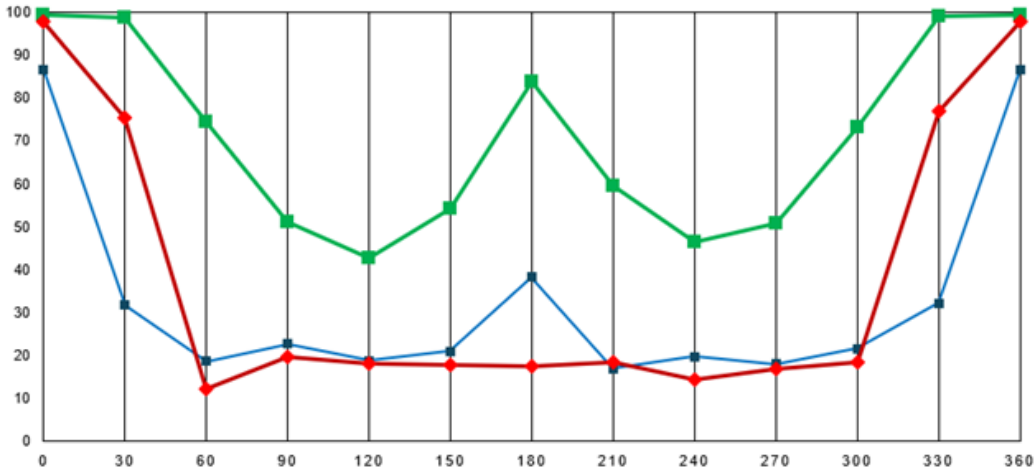


Figure B.7: Effet d'un reclassement avec D3Feat et SpectralGV sur le score $Hits@1$ (en %), selon une rotation d'un angle θ autour de l'axe Z (en degrés sur l'axe des x). Trois configurations sont comparées : une recherche exhaustive simple (en bleu), une recherche exhaustive exploitant SpectralGV (en rouge) et une recherche exhaustive avec SpectralGV amélioré par D3Feat (en vert). Source (Deneuville, 2025).

Pour tous les angles θ, on observe que l'ajout de SpectralGV et de D3Feat (en vert) améliore les performances des recherches par rapport à la recherche exhaustive (en bleu). Le gain de performance peut être estimé à 30% dans les cas où le nuage de points de la requête a été réorienté. Le seul ajout de SpectralGV à une recherche exhaustive n'améliore pas le classement des réponses, voire le dégrade pour les θ proches de 180°.

En conclusion, nous avons pu observer que l'exploitation d'une approche de reclassement améliore notablement les résultats de la recherche des nuages les plus similaires. Nous avons également été en mesure de proposer une amélioration de l'approche de l'état de l'art, SpectralGV, qui permet d'être plus robuste aux transformations géométriques (ici, la rotation) pouvant exister entre le nuage en entrée et la base de nuages interrogée.

Conclusion et perspectives

Les travaux menés au cours de cette thèse se sont concentrés sur l'accélération de la reconnaissance de lieux 3D grâce à une recherche générative. Cette reconnaissance repose uniquement sur la structure géométrique des nuages de points, sans d'autres modalités additionnelles.

La première contribution réside dans l'application des concepts de la recherche d'information à la tâche de reconnaissance de lieux, à travers l'adaptation du modèle DSI aux données 3D. Cela a nécessité de remplacer la modalité textuelle du DSI original, qui génère des identifiants de documents à partir de requêtes textuelles, par des nuages de points 3D. Pour ce faire, une architecture multimodale est employée, en remplaçant l'encodeur d'images utilisé dans les Vision Transformers (ici GIT) par un encodeur de nuages de points existant, à savoir LoGG3D-Net. Une deuxième adaptation a concerné la construction des identifiants des nuages de points. Comme GIT génère les séquences de tokens de manière auto-régressive lors de la recherche de faisceaux, nous avons introduit deux identifiants positionnels, intégrant la position géographique au sein des docids, améliorant ainsi la précision de la recherche, puisque les nuages de points voisins se voient attribuer des identifiants similaires.

La deuxième contribution porte sur la construction d'un nouveau jeu de données dédié à la reconnaissance de lieux, à partir des données LiDAR HD. Inspiré par les jeux de données tels que KITTI et CS-Campus3D, ce jeu de données couvre deux zones distinctes de Paris, de Saint-Mandé à Vincennes à l'est, et du Bois de Boulogne au Champ-de-Mars à l'ouest. En l'absence de modèle préentraîné sur ces données, LoGG3D-Net a été réentraîné et évalué sur la qualité des nouveaux descripteurs.

Enfin, la troisième contribution étend le cadre de DSI-3D grâce à l'intégration d'une architecture de mixture d'experts, permettant une activation partielle des experts disponibles. Le jeu de données LiDAR HD couvrant les deux zones de Paris a été subdivisé en secteurs de surface équivalente, chacun étant associé à un modèle DSI-3D. Ces modèles, spécialisés géographiquement, ont été intégrés en tant qu'experts au sein du MoE. Un réseau de routage a également été entraîné comme classifieur multi-étiquettes afin de déterminer le score d'activation des experts les plus pertinents pour une requête donnée. Cette approche étend la couverture spatiale tout en maintenant un temps d'inférence maîtrisé. Afin de poursuivre cette étude, plusieurs perspectives de recherche peuvent être envisagées :

- optimiser la représentation des docids, en s'inspirant de l'approche proposée par RIPOR, afin que la séquence de tokens capture plus fidèlement la structure spatiale des nuages de points. Une piste consisterait à élargir l'espace des identifiants en autorisant l'usage de caractères alphanumériques ou symboliques, plutôt que de se limiter à des éléments purement numériques, ce qui permettrait d'accroître la quantité d'information encodée dans une séquence de taille fixe;

- étudier des architectures de type Mixture d'Experts hiérarchiques, comportant plusieurs couches de routage où les experts seraient regroupés par zones géographiques (par exemple, par villes, puis par arrondissements). Une telle structuration conserverait la spécialisation locale des experts tout en étendant la couverture de recherche du modèle;

- faciliter la mise à jour du modèle DSI-3D en intégrant des approches d'adaptation à faible rang (Low-Rank Adaptation, LoRA), afin de permettre une actualisation continue sur des jeux de données dynamiques à coût computationnel réduit. Cette approche pourrait être combinée au MoE, en ne réentraînant qu'un sous-ensemble d'experts spécifiquement affectés par les changements du jeu de données;

- étendre DSI-3D à la recherche multi-source, en rendant le système capable de traiter des données issues de modalités hétérogènes, telles que des acquisitions LiDAR terrestres, aériennes, ou des reconstructions 3D dérivées d'images et de vidéos. Cette perspective vise à renforcer la robustesse des représentations et à favoriser la reconnaissance de scènes identiques issues de sources d'acquisition différentes.