HAL open science

# Learning-based algorithms for real-time visual localization of vehicles

Arthur Moreau

## To cite this version:

Arthur Moreau. Learning-based algorithms for real-time visual localization of vehicles. Robotics [cs.RO]. Université Paris sciences et lettres, 2023. English. NNT : 2023UPSLM008 . tel-04122671v2

**HAL Id: tel-04122671**

**https://pastel.hal.science/tel-04122671v2**

Submitted on 8 Jun 2023

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

PSL UNIVERSITÉ PARIS

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à MINES Paris

# Learning-based algorithms for real-time visual localization of vehicles

# Algorithmes d'apprentissage pour la localisation visuelle de véhicules en temps réel


Soutenue par

**Arthur Moreau**

Le 27 avril 2023


École doctorale n°621

**Ingénierie des Systèmes, Matériaux, Mécanique, Energétique**

Spécialité

**Informatique temps réel, robotique et automatique.**

Composition du jury :

| | |
|---|---|
| Valérie Gouet-Brunet<br>Directrice de recherche, LaSTIG, IGN | Rapporteur |
| Patric Jensfelt<br>Professeur, KTH Royal Institute of Technology | Rapporteur |
| Vincent Lepetit<br>Professeur, Ecole Nationale des Ponts ParisTech | Président du jury |
| Arnaud de La Fortelle<br>Professeur, Mines Paris | Directeur de thèse |
| Bogdan Stanciulescu<br>Maître de conférences, Mines Paris | Examinateur |
| Dzmitry Tsishkou<br>Huawei Paris Research Center | Examinateur |

MINES PARIS | PSL

# Abstract

Autonomous driving is expected to revolutionize tomorrow's transportation technologies. Positioning systems are a key component of self-driving vehicles in order to ensure a safe, smooth and reliable navigation. The precise ego position of a vehicle inside of its environment can be recovered by a wide range of sensors and algorithms, but ensuring a high accuracy and reliability in very large environments remains a challenging problem for traditional systems. This thesis aims to solve the map-based relocalization problem with on-board camera sensors and data-driven methods. We build on recent advances on the deep learning area to develop algorithms which learn to relocalize from a large collection of images gathered in the area of interest.

First, we propose to include geometrical clues and uncertainty quantification in direct learning-based visual localization methods and explore their capability to be used as a vehicle localization system in autonomous driving scenarios.

Then, we investigated in different ways the connection between implicit scene representations and visual localization algorithms. By their ability to represent continuously a complex scene in a neural network, these implicit representations can be used to generate photo-realistic synthetic data used to train better algorithms, but also to represent the map of the environment. We show that relevant information captured on roads of several kilometers can be encoded in few megabytes in order to achieve real-time vehicle localization, but also that local features can be learned, stored and rendered by a Neural Field to achieve centimeter-level camera pose estimation in a dense features matching pipeline.

Overall, this work aims to rehabilitate direct learning-based formulations, which are considered to be less precise than classical features-based methods. In the end, the effectiveness of data-driven methods depends on the data and then can be beneficial in some situations, such as autonomous driving.

In addition to that, implicit representations and neural fields, beyond their impressive rendering ability, are rapidly becoming the most effective solution to represent scenes for computer vision and autonomous driving. We have shown some applications of this idea for the localization task and expect to see more in the following years.

## Résumé en Français

La conduite autonome est appelée à révolutionner les transports de demain. Les systèmes de localisation sont un élément clé des véhicules autonomes afin d'assurer une navigation sûre, fluide et fiable. La position précise d'un véhicule dans son environnement peut être déterminée à l'aide de différents capteurs et algorithmes, mais la garantie d'une précision et d'une fiabilité élevées sur des cartes à très grande échelle reste un problème difficile pour les systèmes traditionnels. Cette thèse vise à résoudre le problème de la relocalisation à l'aide de caméras embarquées et de méthodes d'apprentissage automatique. Nous nous appuyons sur les avancées récentes dans le domaine de l'apprentissage profond pour développer des algorithmes qui apprennent à relocaliser à partir d'une grande base de données d'images recueillies dans la zone d'intérêt.

Tout d'abord, nous proposons d'améliorer le raisonnement géometrique et de quantifier l'incertitude dans les méthodes directes de localisation visuelle et nous explorons leur capacité à être utilisées comme système de localisation de véhicules dans des scénarios de conduite autonome.

Dans un second temps, nous avons exploré de différentes manières le lien entre les représentations implicites de scènes et les algorithmes de localisation visuelle. Par leur capacité à représenter de manière continue une scène complexe dans un réseau de neurones, ces représentations implicites peuvent être utilisées pour générer des données synthétiques photo-réalistes utilisées pour entraîner de meilleurs algorithmes, mais aussi pour représenter la carte de l'environnement. Nous montrons que les informations pertinentes capturées sur des routes de plusieurs kilomètres peuvent être encodées en quelques mégabytes afin de réaliser la localisation de véhicules en temps réel, mais aussi que les features locales peuvent être apprises, stockées et rendues par un "Neural Field" pour réaliser une estimation de la position de la caméra au centimètre près dans un algorithme de correspondances locales denses.

Globalement, ce travail vise à réhabiliter les méthodes directes, qui sont considérées comme moins précises que les méthodes classiques basées sur les features. Au final, l'efficacité des méthodes d'apprentissages dépend des données et peut donc être bénéfique dans certaines situations, comme la conduite autonome.

En outre, les représentations implicites, au-delà de leur impressionnante capacité de rendu, deviennent rapidement la solution la plus efficace pour représenter les scènes pour la vision par ordinateur et la conduite autonome. Nous avons montré quelques applications de cette idée pour la localisation dans cette thèse et nous espérons en voir d'autres dans les années à venir.

# Remerciements

Les premiers remerciements liés à cette thèse sont naturellements destinés aux membres du jury, qui ont consacré du temps à la relecture de ce manuscrit et participé à son évaluation. En particulier Valérie Gouet-Brunet et Patric Jensfelt qui ont rapporté la thèse, mais également Vincent Lepetit pour son rôle d'examinateur.

Ces recherches n'auraient pu aboutir sans l'expertise, le soutien et la bienveillance de mes encadrants. Merci à Bogdan et Arnaud, mais aussi à Fabien Moutarde qui a participé au suivi de mon travail. Le travail effectué durant ces 3 années n'aurait pas pu être réalisé sans Dzmitry et Nathan qui ont su me transmettre leur savoir, leur motivation et leur soutien.

L'équipe IoV de Huawei a été un environnement épanouissant pour réaliser mes recherches, à travers son expertise scientifique, mais surtout son ambiance agréable. Moussab, Laurent, Stefano, Quentin, Luis, Yann, Fusang et Arnaud, c'était un plaisir de partager mes journées avec vous. Je n'oublie pas l'équipe ISP de Huawei Nice, où en tant que stagiaire avec Joseph Meehan, j'ai découvert le monde de la recherche qui m'a fasciné et qui, je l'espère, est désormais ma vocation.

La fine équipe des doctorants du CAOR a joué le rôle essentiel de m'emmener me changer les idées le mardi soir dans les tavernes du quartier latin. Une vraie bande de copains qui aide à surmonter les défis d'un doctorat. Je parle bien sûr du gang originel de la V026: Thomas, Camille, Raphaël, Jesus et JoBoule, mais aussi (et j'en oublie sans aucun doute) de Sami, Sofiane, Louis, Hugo, Daniel, Fabio, Angelika et Sascha.

De toute évidence, je n'aurai pu accomplir ces longues années d'études sans le soutien, les conseils, et certaines années la patience, de mes parents. Ils ont su me rendre curieux d'apprendre en toutes circonstances et je les remercie chaudement pour leur soutien inconditionnel.

Les amis les plus anciens, les Popins, sont toujours là et méritent d'être mentionnés. Les années au chemin de la butte n'ont sans doute pas contribuées directement à cette thèse mais ont crée des liens qu'on oublie pas. Plus récemment, j'ai beaucoup apprécié retrouver Federico dans un coin de la butte aux cailles pour se raconter nos ~~vies~~ recherches.

Je termine par le plus grand soutien, ma compagne Ségolène, qui a su endurer la compagnie d'un doctorant préoccupé par ses recherches. Tu as toujours su m'épauler et me donner le sourire, merci. Je suis impatient des nouveaux projets qui nous attendent.

# Contents


**1 Introduction** **3**

1.1 Problem statement: the vehicle localization problem . . . . . 4

1.1.1 Autonomous driving software pipeline . . . . . 4

1.1.2 Vehicle positioning systems . . . . . 5

1.1.3 Image-based positioning systems . . . . . 6

1.2 Objectives: a map-based autonomous driving localization scenario . . . . . 6

1.3 Publications and communications . . . . . 7

1.4 Outline . . . . . 8

**2 Camera Relocalization algorithms: from classical pipelines to learning-based solutions** **11**

2.1 Résumé en Français . . . . . 12

2.2 Coarse localization with Visual Place Recognition . . . . . 13

2.3 Visual Localization with Structure-Based methods . . . . . 14

2.3.1 Local features detection and description . . . . . 14

2.3.2 Local Features Matching . . . . . 16

2.3.3 Camera Pose Estimation with Perspective-N-Points . . . . . 17

2.3.4 Camera Pose Estimation with Direct Features Alignment . . . . . 17

2.3.5 Camera Pose Estimation with Relative Pose Regression . . . . . 18

2.3.6 Advantages and Drawbacks of Structure-Based methods . . . . . 18

2.4 Direct learning-based methods . . . . . 18

2.4.1 Absolute Pose Regression . . . . . 19

2.4.2 Scene Coordinate Regression . . . . . 20

2.5 Discussion: Which solutions for autonomous driving scenarios . . . . . 21

**3 Vehicle localization with Absolute Pose Regression** **25**

3.1 Résumé en Français . . . . . 26

3.2 Motivation . . . . . 27

3.3 Geometric inductive biases in Absolute Pose Regression networks . . . . . 27

3.3.1 Breaking translation invariance with CoordConvolutions . . . . . 27

3.3.2 Learning local confidences with weighted average pooling . . . . . 28

3.3.3 Intrinsic coordinates convolutions for crowd-sourced images . . . 29
3.4 Uncertainty-aware Pose Regression . . . 30
3.4.1 Homoscedatic vs Heteroscedatic uncertainties . . . 31
3.4.2 Joint learning of pose regression and heteroscedatic uncertainty . . . 31
3.4.3 Localization under uncertainty . . . 32
3.5 Absolute Pose Regression with CoordiNet . . . 34
3.6 Experiments . . . 34
3.6.1 Comparison with related methods . . . 34
3.6.2 Evaluation on larger scale datasets . . . 36
3.6.3 Evaluation of localization under uncertainty . . . 37
3.6.4 Ablation studies . . . 39
3.6.5 Localization with crowd-sourced images . . . 40
3.7 Conclusions . . . 41

**4 Literature review: Neural networks as implicit scene representations 43**
4.1 Résumé en Français . . . 44
4.2 Implicit Neural Representations . . . 45
4.3 Neural rendering with Radiance Fields . . . 46
4.3.1 Novel view synthesis . . . 46
4.3.2 Neural Radiance Fields . . . 47
4.4 Improving Neural Radiance Fields . . . 48
4.4.1 Accelerate NeRF training and rendering . . . 48
4.4.2 Improving further rendering quality . . . 50
4.4.3 Neural rendering of dynamic outdoor scenes . . . 52
4.5 Robotics applications of Implicit Neural Representations . . . 53
4.6 Idea: Use implicit scene representations as the visual localization map . . . 54

**5 Visual Localization augmented by NeRF synthesis 57**
5.1 Résumé en Français . . . 58
5.2 Motivation . . . 59
5.3 Related work . . . 60
5.4 Synthetic dataset rendering with LENS . . . 61
5.4.1 NeRF-W training . . . 61
5.4.2 Density volume generation . . . 62
5.4.3 Virtual camera locations . . . 62
5.4.4 Novel view synthesis . . . 63
5.4.5 Camera pose regressor training . . . 63
5.5 Experiments . . . 63
5.5.1 Datasets and implementation details . . . 64
5.5.2 Comparison with related localization methods . . . 65
5.5.3 Ablation studies . . . 65
5.5.4 Exploring camera pose regression on synthetic domain . . . 67

5.5.5 Limitations . . . 68
5.6 Conclusion . . . 68

**6 Large scale localization with an implicit map representation 69**
6.1 Résumé en Français . . . 70
6.2 Motivation . . . 71
6.3 Visual Localization with ImPosing . . . 72
6.3.1 ImPosing localization process . . . 72
6.3.2 Training procedure . . . 74
6.4 Experiments . . . 75
6.4.1 Single scene localization . . . 76
6.4.2 Multi-scene localization . . . 77
6.4.3 Continual learning in the real world . . . 78
6.4.4 Smaller scale benchmarks . . . 79
6.4.5 Efficiency comparison . . . 80
6.4.6 Ablation study . . . 81
6.4.7 Qualitative analysis . . . 82
6.5 Discussion . . . 83
6.6 Conclusion . . . 84

**7 Self-supervised localization features in a radiance field 85**
7.1 Résumé en Français . . . 86
7.2 Motivation . . . 87
7.3 Method . . . 87
7.3.1 Neural rendering of positional descriptors . . . 87
7.3.2 Self-supervised training of features in CROSSFIRE . . . 88
7.3.3 Visual Localization by iterative dense features matching . . . 91
7.4 Experiments . . . 92
7.4.1 Comparison to related methods . . . 92
7.4.2 How good the pose priors need to be ? . . . 94
7.4.3 Ablation studies . . . 95
7.4.4 Efficiency . . . 97
7.5 Limitations and Future Work . . . 97
7.6 Conclusion . . . 97

**8 Conclusion 99**
8.1 Résumé en Français . . . 100
8.2 Summary of contributions and discussions . . . 101
8.3 Future work: a hierarchical vehicle localization algorithm in implicit maps . . . 102
8.4 Perspectives: an autonomous driving solution based on neural scenes representations . . . 103

**9 Bibliography 105**

# List of Figures


2.1 **Localization with image retrieval:** First, the global image descriptors of reference images are computed and stored in a database in an offline step. During navigation, the query image is processed and matched against the database. A localization estimate can be computed by pose averaging on the retrieved images. . . . . 14

2.2 **Classical and learning-based keypoints detection:** We display and compare detected keypoints from the classical Harris corner detector [Harris and Stephens, 1988] (top) and SuperPoint [DeTone et al., 2018] (bottom). . . . . 15

2.3 **Local correspondences between a queries and retrieved images** Inliers 2D-3D correspondences between query images (left) and reference images (right) of the HLoc method [Sarlin et al., 2019]. Local 2D-2D correspondences are upgraded to 2D-3D because the 3D position of reference keypoints are known. . . . . 16

2.4 **Localization with hierarchical features matching pipelines:** After the retrieval step, local descriptors from the top ranked images are matched against the query images, resulting in 2D/3D correspondences, from which a precise camera pose can be computer with Perspective-N-Points + RANSAC. . . . . 17

2.5 **Localization with Absolute Pose Regression:** The whole task in handled by a single neural network which regresses position and orientation of the camera. Before deployment, a scene-specific training is required. . . . . 19

2.6 **Scene Coordinates Regression.** 3D scenes coordinates are extracted from a 3D model (left). SCR regresses the 3D coordinates of the visual content observable in the query image (right). . . . . 21

2.7 **Visual localization classes of methods:** A query image can be localized through different type of algorithms that rely either on local features, global features or direct regression. From [Humemberger et al., 2021]. . . . . 22

3.1 **CoordConvolutions:** additional channels with 2D pixels coordinates are concatenated before the convolution operation. . . . . 28

3.2 **Confidence-Weighted Average Pooling:** One channel of the last feature map is used as confidence weights to perform aggregation of local estimates in a single vector. . . . . 29

3.3 **Visualization of pooling activations:** bottom images are inputs of the model, top images are inputs multiplied by the upsampled confidence map of the pooling layer . . . . . 29
3.4 **Unavoidable failure cases** Dynamic objects generate occlusions that results in failure cases (samples from Oxford RobotCar dataset [Maddern et al., 2017]). . . . . 30
3.5 **Calibrated uncertainties.** Comparison between uncalibrated (left) and calibrated (right) uncertainties, plotted for x (top) and y (bottom) axis. . . . . 33
3.6 **CoordiNet architecture:** the input image is sent to a pretrained image encoder, then pose and uncertainty are predicted from encoder features in two separate decoders. Pose decoder uses Coord convolutions layers. GAP and CWAP refer to Global Average Pooling and Confidence Weighed Average Pooling. . . . . 34
3.7 **Trajectories on Oxford Robotcar.** Left images are Loop results (2014-06-23-15-36-04, 2014-06-26-08-53-56) trained with 2 sequences, right images are Full results (2014-12-09-13-21-02). Middle right is trained with 2 sequences, right is trained with 15. Color map represents errors at a given location: blue is $\sim 1m$ error and red is $> 5m$ error. . . . . 35
3.8 **Samples from Paris (top) and Shanghai (bottom) datasets.** . . . . . 37
3.9 **Localization with uncertainty.** Top lane from left to right: CoordiNet predictions, EKF with CoordiNet poses and fixed covariance. Bottom lane from left to right: EKF with CoordiNet raw uncertainty, EKF with CoordiNet calibrated uncertainty. Colormap is the same as in figure 3.7. . . . . 38
3.10 **EKF on Oxford experiment.** Top lane: CoordiNet trained with 2 runs. Bottom lane: EKF using CoordiNet poses and uncertainty. Colormap is the same as in figure 3.7. . . . 39
3.11 **CoordiNet and EKF trajectories:** CoordiNet sequences of poses (red line) are shown with the uncertainty estimate of the current pose (purple ellipsoid). EKF trajectory (blue line) and ground truth (green line) are also displayed. Figure best viewed in color. . . . . 40
3.12 **Samples from the Brandenburg Gate scene in Photo Tourism dataset [Snavely et al., 2006].** Each image has been captured from varying conditions with a different device. . . 41

4.1 **Implicit Neural Representations:** Data such as images, 3D shapes or sound can be represented in neural network which takes coordinates as input and provides quantities of interest. . . . . 45
4.2 **Novel view synthesis.** Given a sparse set of observations of a scene (left) and their corresponding camera poses (middle), novel view synthesis algorithms are asked to render images of the same scene from unobserved camera poses (right). Figure from [Mildenhall et al., 2020]. . . . . 46
4.3 **Neural Radiance Fields:** NeRF are an implicit scene representation which takes a 3D point and a viewing direction as input, and returns the volume density at this point and a RGB color. Rendering a pixel is performed by ray marching in the 3D space, evaluation of NeRF on a discrete set of sampled points, and differentiable aggregation along the ray by volume rendering. Figure from [Mildenhall et al., 2020]. . . . . 47
4.4 **Instant-NGP architecture.** Intermediate features at a given 3D point are extracted via trilinear interpolation of multi-resolution hash tables of features. Density and color are then decoded in a tiny MLP, enabling fast rendering. Figure from [Müller et al., 2022]. . 49

4.5 **Light Fields Network.** Ray marching and volume rendering are replaced by an implicit representation of the light fielf. Inputs are camera rays, parametrized by Plücker coordinates. Figure from [Sitzmann et al., 2021] . . . . . 50
4.6 **MipNeRF.** NeRF samples points along the camera ray. In reality a pixel captures the light emitted from a conical frustum. Mip-NeRF uses integrated position encodings to capture the variations in resolution and remove aliasing effects. Figure from [Barron et al., 2021]. . . . . 51
4.7 **Rendering of dynamic scenes.** Nerf-W is able to render a scene with controllable appearance from various tourists photographs. More examples shown in Figure 3.12. . . . 53
4.8 **NeRF-W.** The model renders the static scene in one branch and images with transient objects in the other. From [Martin-Brualla et al., 2021] . . . . . 53
4.9 **Implicit scene reconstructions with RGB-D SLAM:** iMAP [Sucar et al., 2021] (left) and NICE-SLAM [Zhu et al., 2022] (right) . . . . . 54

5.1 **Localization failure during an overtaking.** Absolute Pose Regression fails to recover the vehicle position in the left lane because this situation requires to extrapolate outside of the training dataset positions. . . . . 59
5.2 **Novel view synthesis with LENS:** given a 3D scene defined by a set of images labeled with corresponding poses (green cameras), we train a NeRF and render novel views (depicted images) on pose queries distributed across the scene (red cameras). Synthetic and real images are gathered to train a camera pose regression model, which performs twice better when evaluated on test set images (blue cameras) compared to the same model trained only on real training samples. . . . . 60
5.3 **LENS pipeline:** First, NeRF-W is trained with available real images (a.). Then, the trained model is used to detect high density points in the scene (b.). Poses from real images and high density locations are used to generate virtual camera locations (c.) on which novel view synthesis is performed (d.). Combined real and synthetic datasets are used to train a pose regressor (e.). . . . . 61
5.4 **Appearances interpolation with Nerf-W.** We display renderings from the same viewpoints with randomly interpolated appearance embeddings. We observe that resulting appearances look natural and exhibit a good diversity. . . . . 63
5.5 Visualisation of density volumes, virtual camera queries (training poses, test poses,virtual cameras) and example of rendered images on Cambridge Landmarks. . . . . 64
5.6 Translation (tr., left) and angular (ang., right) error relative decrease vs synthetic dataset size. . . . . 66
5.7 Virtual camera locations with (left) and without (right) NeRF volume pruning step. . . . 67

6.1 **High level comparison between Absolute Pose Regression (top left), Image Retrieval (top right) and the proposed algorithm (bottom).** APR entangles the entire visual localization task in a single neural network. IR extracts a global image descriptor from the query image and compares it to a database of descriptors. Our proposal models the reference database as a neural network. . . . . 71

6.2 **Implicit pose encoding for hierarchical image localization.** A set of initial map signatures is compared to the image signature to determine the most probable localization of the camera. The similarity scores guides the selection of a new batch of pose candidates that are used to compute the new map signatures for the second refined localization step. This process is repeated multiple time to predict the final camera pose. . . . . 73
6.3 **Iterative candidates refinement.** At each $k$ step of the localization process, top scored poses are selected to sample the new candidate poses at step $k+1$. From left to right: top scored poses at $k = 0$ to $k = 5$, yellow points are positions of the training example, blue arrows are pose candidates and red arrows are the selected poses among the candidates. . 74
6.4 **Featureless environments and varying weather conditions.** Test is performed on the image on the right, while the network has been trained with 3 recordings with different lightning conditions. Our method is able to provide a coarse localization in these scenarios, where as image retrieval and pose regression competitors fail. . . . . 78
6.5 **Localization accuracy in continual learning setup.** The model is trained each day with new recordings automatically annotated and improves over time. . . . . 79
6.6 **In-device memory usage.** Structure-based methods (black) and image retrieval (blue and purple) use more memory when the reference dataset grows whereas pose regression methods and ImPosing (pink and cyan) storage requirement does not depend on dataset size. . . . . 81
6.7 **From left to right:** median localization errors depending on number of refinements, pose candidates, and final number averaged poses. Training time comparison between pose regression [Moreau et al., 2022a] and ImPosing. . . . . 82
6.8 **Multimodal score distribution in ambiguous cases.** Many road areas present similar structure and appearance, introducing ambiguities in the localization task. In this scenario from the City Loop scene, the model outputs high scores for areas depicted in left and right images, which are very far one from each other. By refining the estimate in further steps, the model is able to solve this ambiguity in most cases. . . . . 83
6.9 **Latent space visualization.** Training poses, colored by the 3 principal components of map descriptors. Poses with similar colors are close in the latent space. Opposite ways of the same road are represented by dissimilar representations. Best viewed in color . . . 83

7.1 **Neural Positional Feature Fields architecture.** We build upon the architecture of Instant-NGP [Müller et al., 2022] enabling fast training and rendering. We use per-image appearance embeddings during training to handle varying illumination across training images. For accurate localization, we add an MLP that is invariant to viewing direction and appearance vector allowing to learn robust localization feature vectors. . . . . 88
7.2 **Training pipeline of Neural Positional Features Fields.** We jointly optimize the neural renderer and the features extractor to obtain robust, scene-specific localization descriptors. We use regularization losses (i.e. TV and SSIM) to increase the consistency of the neural renderer. We propose a two-terms loss that maximizes the similarity between corresponding feature maps while penalizing pixel pairs that are geometrically distant from each other. . . . . 89

7.3 **Similarities of positional features.** We show the dense matching map between one descriptor from the query image (red dots in left images) and the reference descriptors from the neural renderer. Thanks to our training objective, descriptors close (in 3D) to the selected points have high similary whereas others do not match. This behaviour is enforced by our loss function. . . . . 90
7.4 **Localization procedure.** Descriptors are extracted from the query image and matched against descriptors rendered from the localization prior. Depth information provides 2D-3D matches that enable to compute the pose with PnP + RANSAC. This process can be repeated iteratively, by rendering descriptors from the predicted pose. . . . . 91
7.5 **Visualization of rendered views, descriptors and matches in StMarysChurch.** We show on the top row the query image (right), the RGB rendered view from the localization prior (left) and from the 1st estimated pose (middle). The second row represents a PCA visualization of the corresponding descriptors map from the neural renderer (left and middle) and the features extractor (right). The last row displays the inlier matches obtained by our pipeline. . . . . 93
7.6 **Iterative visual localization from an imprecise prior.** Starting from a coarse localization prior, our algorithm estimates the pose of a query image iteratively by comparing image features to descriptors rendered from a neural scene representation. . . . . 94
7.7 **Success and failure cases:** Using dense features field for localization enables to establish accurate correspondences in texture-less areas (left). Failure cases are observed in the presence of dynamic objects (middle), for which the PnP converges on a wrong pool of matches, and ambiguous cases (right) where the CNN mixes up the symmetrical parts of the church due to lack of long-range reasoning. . . . . 95
7.8 **Qualitative comparison of descriptors between the proposed loss and a classical triplet loss.** We visualize the PCA of descriptors from our loss (middle) and a triplet (right) for a given query image (left). . . . . 95
7.9 **Similarities scores between a CNN query pixel and rendered descriptors, depending on $\lambda$.** The left image shows a query pixel from a test sample while other images show a colormap of the similarity between rendered descriptors from the same viewpoint and the query pixel. Yellow color indicates a high similarity. . . . . 96
7.10 **Localization accuracy depending on descriptor head inputs.** We compare the final accuracy on the "Chess" scene with and without the viewing direction as descriptor input in the neural renderer. . . . . 96

8.1 **Hierarchical visual localization with implicit maps.** A first camera pose is estimated by ImPosing and then refined by dense features matching in a Neural Positional Features Field. . . . . 102

# List of Tables


3.1 **Absolute Pose Regression methods on Oxford RobotCar dataset.** CoordiNet is compared to other APR methods. The pose considered is the raw network output without post processing methods. Empty cells correspond to results non reported on the related papers. . . . . 35

3.2 **Absolute Pose Regression methods on Cambridge Landmarks dataset.** . . . . . 36

3.3 **Results on large scale datasets.** . . . . . 37

3.4 Ablation study on Shanghai dataset (errors in meters/degrees). . . . . 39

3.5 **Results on Photo Tourism dataset, with and without camera-aware convolutions.** . . 41

5.1 **6DOF localization errors of visual localization methods.** We report median translation/orientation error (meters/degrees). TransPN and DirectPN stand for TransPoseNet and DirectPoseNet. Superscripts numbers refer to the relative improvement (green) or deterioration (red) in percentage brought by synthetic data. . . . . 66

5.2 median translation (m) and orientation (°) errors depending on synthetic dataset size . . . 66

5.3 Localization errors comparison for density volume and appearances embedding ablations on Shop Facade scene. . . . . 67

5.4 Median translation (m) and orientation (°) errors in Fire scene from 7scenes. seq3-4 refers to methods using images from sequences 3 & 4 as training data. . . . . 67

5.5 Approximate time and memory requirements comparison between structure-based methods and ours. * denotes a scaling time and memory consumption according to the scene size. † structure-based methods need to load the features vocabulary and access to the 3D points cloud during localization. . . . . 68

6.1 Localization error on Oxford RobotCar and Daoxiang Lake datasets. . . . . 76

6.2 Median localization error on 4Seasons dataset. . . . . 77

6.3 Small-scale datasets (median localization error in meters/degrees). . . . . 80

6.4 **Qualitative comparison between methods.** We compare the properties of visual localization class of methods w.r.t. storage requirement, capability to operate in large maps (scalability), latency and accuracy. *IR* stands for Image Retrieval, *PR* for Pose Regression, *SCR* for Scene Coordinate Regression, *DB* for database and *NN* for neural networks weights. Storage of IR databases are detailed in [Song et al., 2022]. . . . . 80

7.1 **6-DoF median localization errors of learning-based visual localization methods.** NPFF is the best performing method on indoor scenes and second best in outdoor. . . . . 94
7.2 Median error w.r.t. prior strategy and No. of iterations. . . . . 95

Chapter 1

# Introduction

## Contents


**1.1 Problem statement: the vehicle localization problem** . . . . . **4**

1.1.1 Autonomous driving software pipeline . . . . . 4

1.1.2 Vehicle positioning systems . . . . . 5

1.1.3 Image-based positioning systems . . . . . 6

**1.2 Objectives: a map-based autonomous driving localization scenario** . . . . . **6**

**1.3 Publications and communications** . . . . . **7**

**1.4 Outline** . . . . . **8**

The starting point of this PhD thesis is the perspective of autonomous driving systems: the deployment of such a technology at scale will require advanced computer vision algorithms to solve complex tasks in real-time without human intervention. This field has made significant progress in the last decade thanks to machine learning approaches, which paves the way for learning the entire driving task from data. The research conducted in this thesis aims to develop this direction and explores deep learning solutions for autonomous driving. More specifically, we focus on mapping and localization algorithms from visual sensors. It consists in analysing video streams from cameras mounted on vehicles to understand the surrounding environment of the vehicle, build a map of the area and then being able to re-localize inside of this map during a new navigation scenario. Determining the ego-position of the vehicle is crucial to ensure the safe and smooth navigation of a car. This thesis proposes several innovative contributions about image-based relocalization algorithms with learning-based approaches. This task involves several challenges, such as recovering meaningful information about the 3D world while images are a 2D signal, dealing with a dynamic outdoor environment where lightning conditions and moving objects change over time but also developing systems which are computationally tractable in order to operate in real-time embedded into a vehicle.

First, we introduce the problem from a practical point of view in section 1.1: We briefly present the classical pipeline of autonomous driving, we discuss the different options for positioning systems of vehicles, define the background concepts, and highlight the main challenges induced by this task. Then, in section 1.2, we define a scenario for autonomous driving based on high definition maps and discuss the scientific perspectives, the industrial constraints and some technical specifications which would enable to develop visual localization methods that can be deployed at scale for autonomous vehicles. We define objectives and formulate hypothesis which have been evaluated through the research work of this thesis. Finally, section 1.3 presents the publications and communications that emerged from the work of this thesis.

## 1.1 Problem statement: the vehicle localization problem

### 1.1.1 Autonomous driving software pipeline

Driving is a highly complex task that requires to understand correctly the surrounding environment and to react appropriately to the actions of the other road users. Automating this process is usually tackled by a combination of different subtasks described below:

- **Sensors:** Automated vehicles are equipped with a sensors stack which capture raw information about the surrounding environment in real-time. This can include one or multiple cameras, GNSS, lidar and radar. Each sensor has its own advantages and drawbacks: cameras provide a dense photometric signal but the information is projected to a 2D plane, while a lidar captures accurate 3D information but is limited to a sparse signal in a close range. This thesis focuses on algorithms which use only camera images.

- **Perception:** The raw data captured by sensors is processed by perception algorithms in order to provide useful information about the surrounding environment. Perception contains geometric ("where") and semantic ("what") information. The scene understanding tasks involved in vehicle

perception can include (but are not limited to): detection and tracking of other roads agents such as vehicles [Betke et al., 2000, Mahmoud et al., 2023] and pedestrians [Dollar et al., 2011, Liu et al., 2019], detection of lane markings and road geometry [López et al., 2010], traffic sign recognition and classification [Zaklouta and Stanciulescu, 2014, Tabernik and Skočaj, 2019], pixel-wise semantic segmentation of the scene [Xie et al., 2021a, Chen et al., 2017], pixel-wise depth prediction [Bhat et al., 2021, Godard et al., 2019] or semantic scene completion [Roldao et al., 2022]. Overall, the goal of perception is to obtain an accurate intermediate representation of the environment which is simplified and interpretable compared to the raw sensors measurements, such that it can be used in later steps.

- **Localization:** An other requirement of automated driving is to know the ego position of the vehicle at any time. This position is defined w.r.t. a map of the environment, which can be pre-computed before deployment (relocalization approach) or built in real-time during navigation in the Simultaneous Localization and Mapping (SLAM) scenario. Vehicle positioning algorithms can leverage data from various sensors [Kuutti et al., 2018], but the more accurate and robust solution is to use several of them through sensor fusion [Levinson et al., 2007, Fayyad et al., 2020]. This thesis, though, will not address sensor fusion but rather try to maximize the accuracy and robustness of image-based solutions, which can later be fused with other signals. Similar to perception, localization is an intermediate step used in later computation, but its accuracy is crucial to ensure safety and smoothness of the vehicle trajectory.

- **Prediction:** Once a representation of the surrounding environment has been built, the autonomous vehicle needs to reason about the future evolution of the scene. It consists in predicting the future trajectory and actions of the other road agents. Premise work in this field rely on physical models, time series extrapolation and causality principles [Lefèvre et al., 2014]. Since then, prediction of vehicles trajectory [Zhao et al., 2021, Gilles et al., 2021], pedestrians intentions [Gesnouin et al., 2021] and trajectory [Achaji et al., 2022] rely on machine learning methods that use the information coming from perception and localization as input.

- **Path planning and control:** Finally, given the surrounding environment and its probable future, the vehicle needs to plan its future trajectory to reach the target destination, while being sure to avoid any possible collision, respect road legislation and ensure a smooth navigation for the passengers.

A potential failure in the localization step will result in erroneous input data for the prediction and path planning tasks which can cause collisions with other road agents in the worst case and a jerky and uncomfortable vehicle trajectory in the best case.

### 1.1.2 Vehicle positioning systems

The most common way to localize a vehicle is to use GNSS technology, however depending on the cost, its localization accuracy is limited ($\approx 5m$) and not sufficient for an autonomous driving system. In addition to that, areas such as tunnels or underground car park are GPS-denied while localization is still

needed. Plenty of other sensors can be used to build positioning or improve the accuracy of existing ones with sensor fusion.

Wheel odometry estimates the motion by integrating the number of turns of the vehicle wheels over time [Merriaux et al., 2014]. This is a cheap solution because the information is usually available in the CAN bus of the vehicle, but not very precise when used alone ($\approx 5m$) and accumulates drift over time. Inertial Measurement Units (IMU) use accelerometers and gyroscopes to capture the movement of the vehicle and the position can also be recovered by integrating these signals [Yi et al., 2007]. In urban areas, WiFi signals can be triangulated to recover the vehicle position [Dinh-Van et al., 2017, Nguyen et al., 2016, Hernández et al., 2017]. Radar is an alternative solution [Ward and Folkesson, 2016]. Lidar can be used to estimate the vehicle odometry [Dellenbach et al., 2021] or absolute position [Deschaud, 2018, Bârsan et al., 2018, Phillips et al., 2021] with a high precision but at a high cost. Finally, image-based localization are an emerging, but yet well-studied, technology which probably offers the best trade-off between sensor cost and localization accuracy. We present these techniques in section 1.1.3.

### 1.1.3 Image-based positioning systems

There are two main different strategies to estimate the position of a vehicle from camera images:

- **Visual Simultaneous Localization and Mapping (vSLAM):** In this scenario, we don't know the environment on which we drive beforehand. As a result we need to build a map of the environment during the navigation and localize new images in this map at the same time. SLAM usually leverages visual odometry [Yousif et al., 2015, Tang et al., 2018], which consists in estimating the relative pose between 2 images captured at consecutive timesteps. It can be approached by features-based methods [Mur-Artal et al., 2015, Sumikura et al., 2019, Li et al., 2020] or direct approaches [Engel et al., 2014, Tateno et al., 2017, Yang et al., 2020]. It is compatible with different sensors setup: monocular RGB or RGB-D camera and stereo cameras.

- **Mapping and Relocalization:** Alternatively, one can first collect data in the area of interest, build a map of the environment offline before deployment, and then estimate the camera pose of images captured during navigation w.r.t. the reference map (relocalization). This scenario is also referred as map-based localization [Chalvatzaras et al., 2022]. While collecting data and computing a map before deployment is a constraint, it enables to build very precise maps and then to provide more accurate and reliable localization systems. Alternatively, vSLAM maps can be used in relocalization mode.

This thesis addresses the mapping and relocalization scenario, and in particular the relocalization algorithms with a single RGB camera. Section 1.2 gives more insights on the problem that this industrial PhD wants to address to develop vehicle localization systems, and chapter 2 presents the prior literature on relocalization algorithms.

## 1.2 Objectives: a map-based autonomous driving localization scenario

We aim to develop a mapping and localization system able to operate for autonomous driving at scale in an industrial scenario. We set as objective the following specifications given some assumptions:

- We assume that lots of data can be collected in target environments, either by the service provider or gathered from the service users in a crowd-sourced way.
- We can afford extensive offline computation on servers before deployment, such as mapping algorithms and training machine learning models. On the other hand, the relocalization algorithm has to be compatible with real-time computation at high frequency in the vehicle.
- We need a robust system able to operate on public roads, with kilometers-scale maps, changing lightning conditions, and interactions with other road users.
- We want a "self-learning" system, in the sense that the localization service can be deployed in an area and improve over time while more data is collected without relying on human annotation in the loop.
- Finally, we need an accurate enough localization algorithm (sub-meter accuracy), able to ensure safety and robustness of the autonomous driving software pipeline.

This scenario is quite different than the public visual localization benchmarks which usually use a small amount of reference data. State-of-the-art features-based localization algorithms are very accurate but struggle to scale with larger datasets (see chapter 2). This is the main motivation for developing direct learning-based methods during this thesis.

## 1.3 Publications and communications

This PhD has been conducted in the center for robotics of Mines Paris, PSL University in the context of an industrial collaboration with Huawei Technologies France, in the Internet of Vehicles (IoV) team of Paris Research Center. The main publications and communications of this thesis can be synthesized as follows:

- A. Moreau, N. Piasco, D. Tsishkou, B. Stanciulescu, A. de La Fortelle. LENS: Localization enhanced by NeRF synthesis. 5th Annual Conference on Robot Learning (CoRL 2021).
- Best oral presentation for "Visual-based localization enhanced by NeRF synthesis" at Journée des Jeunes Chercheurs en Robotique (JJCR 2021).
- A. Moreau, N. Piasco, D. Tsishkou, B. Stanciulescu, A. de La Fortelle. CoordiNet: uncertainty-aware pose regressor for reliable vehicle localization. Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV 2022).
- A. Moreau, T. Gilles, N. Piasco, D. Tsishkou, B. Stanciulescu, A. de La Fortelle. ImPosing: Implicit Pose Encoding for Efficient Visual Localization. Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV 2023).

- A. Moreau, N. Piasco, M. Bennehar, D. Tsishkou, B. Stanciulescu, A. de La Fortelle. NPFF: Self-Supervised Neural Positional Features Fields for Visual Localization. Under submission at CVPR 2023.

The research conducted during the thesis also led to the filling of 3 European patents which are currently under examination at the time of publication:

- Mobile device with reliable localization module trained from crowd sourced rendered views
- Apparatus and methods for visual localization with compact learned map representation
- Visual Positioning System based on descriptors equiped Neural Fields

## 1.4 Outline

This thesis is laid out in eight chapters:

**Chapter 1: Introduction.** We briefly introduce the context of the research of this thesis, including the autonomous vehicle software pipeline, vehicle positioning systems and more specifically the distinction between SLAM and map-based localization. Then, we define the objective of the research by taking into account industrial specifications.

**Chapter 2: Camera relocalization algorithms: from classical pipelines to learning-based solutions.** We present a literature review of visual localization algorithms, focusing on single RGB queries. We present existing classes of algorithms: visual place recognition with image retrieval, structure-based methods relying on 3D models and local features and finally direct learning-based methods. We describe their overall approach, extensions and respective benefits and limitations. We conclude by discussing the applicability of these solutions for autonomous driving and highlight that direct learning-based methods are a promising direction in this scenario.

**Chapter 3: Vehicle localization with Absolute Pose Regression.** This chapter describes research which has been conducted to develop vehicle localization systems with Absolute Pose Regression algorithms. We propose to include geometry inductive biases and heteroscedatic uncertainty quantification in a fully-convolutional architecture. During navigation, we filter the predicted positions coupled with their uncertainties over time with an EKF. We explore the capability of our system CoordiNet [Moreau et al., 2022a] on public benchmarks and large-scale driving datasets. We observe a satisfying localization accuracy when the training data is abundant, but also some limitations which have led to the next contributions.

**Chapter 4: Literature review: Neural networks as implicit scene representations.** We review the recently introduced Implicit Neural Representations, which represent data points, such as a 3D scene, in coordinate-based neural networks. We discuss more in details Neural Radiance Fields (NeRF) and related neural rendering techniques. Specifically, we present NeRF improvements which reduce training and rendering times, use more advanced scene formulations, and deal with dynamic scenes. Finally, we represent robotics applications of implicit scene representations and introduce the general guideline idea of following chapters: use such representations as the visual localization map.

**Chapter 5: Visual Localization augmented by NeRF synthesis.** We present a simple technique to improve Absolute Pose Regression with NeRF generated data. Given a sparse set of captured reference

frames, we train a NeRF model, use it to synthesize photorealistic views uniformly distributed around the scene, and train an Absolute Pose Regressor with a large training dataset composed on both real and synthetic images. Using our method LENS [Moreau et al., 2022b], we observe a large accuracy improvement on public benchmarks where pose regression usually under performs. Ablation experiments show that the improvement not only comes from a larger quantity of data, but rather from a better distribution of the training dataset.

**Chapter 6: Large scale localization with an implicit map representation.** The well-established solution to address visual localization in large-scale maps is to use visual place recognition algorithms, based on image retrieval. This chapter investigates the idea of replacing the image retrieval database by a MLP that implicitly represents the visual content of kilometers-scale driving maps. In this formulation, the global descriptor of "any" $SE(3)$ pose can be evaluated. As a result, localization is not tackled by comparing to a discrete and finite database, but rather by iterative dense sampling of camera pose candidates. We confirm the benefits of our method ImPosing [Moreau et al., 2023a] on very diverse driving scenarios as well as its ability to scale up to large-data regimes.

**Chapter 7: Self-supervised localization features in a radiance field.** This chapter extends the idea of implicit maps representations to structure-based localization algorithms. We use local features matching to address precise camera pose estimation. However, we replace the commonly used sparse 3D model by a Neural Field able to render scene-specialized local features. This scene representation presents several advantages such as compactness, the ability to perform dense features matching and to refine the pose an arbitrary number of times. We describe our system CROSSFIRE [Moreau et al., 2023b], how to train it in a self-supervised way and evaluate its properties in indoor and outdoor scenarios.

**Chapter 8: Conclusion.** Finally, we summarize our contributions and explain how they could be combined together in a hierarchical visual localization system. We conclude the manuscript by discussing the perspective of autonomous driving solutions based on implicit representations.

Chapter 2

# Camera Relocalization algorithms: from classical pipelines to learning-based solutions

## Contents


**2.1 Résumé en Français** . . . . . **12**

**2.2 Coarse localization with Visual Place Recognition** . . . . . **13**

**2.3 Visual Localization with Structure-Based methods** . . . . . **14**

2.3.1 Local features detection and description . . . . . 14

2.3.2 Local Features Matching . . . . . 16

2.3.3 Camera Pose Estimation with Perspective-N-Points . . . . . 17

2.3.4 Camera Pose Estimation with Direct Features Alignment . . . . . 17

2.3.5 Camera Pose Estimation with Relative Pose Regression . . . . . 18

2.3.6 Advantages and Drawbacks of Structure-Based methods . . . . . 18

**2.4 Direct learning-based methods** . . . . . **18**

2.4.1 Absolute Pose Regression . . . . . 19

2.4.2 Scene Coordinate Regression . . . . . 20

**2.5 Discussion: Which solutions for autonomous driving scenarios** . . . . . **21**

## 2.1 Résumé en Français

Nous présentons une revue de la littérature sur les algorithmes de localisation visuelle prenant une unique image en entrée.

Nous présentons les classes d'algorithmes existantes : image retrieval, méthodes basées "features" reposant sur des modèles 3D et, enfin, les méthodes d'apprentissage direct. Nous décrivons les différentes approches, leurs extensions et leurs avantages et limites respectifs.

Nous concluons en discutant de l'applicabilité de ces solutions pour les scénarios de conduite autonome et soulignons que les méthodes basées sur l'apprentissage direct sont une voie prometteuse dans ce scénario, grâce à leur capacité de passer à l'échelle lorsque de grands jeux de données sont disponibles.

This chapter presents a literature review of prior work on Camera Relocalization methods, also known as Image-based Localization. The task is to estimate the 6-DoF camera pose $(t, q) \in SE(3)$ of a query image $I$, where $t$ is a translation vector and $q$ is a unit quaternion. Camera poses are defined in the absolute coordinate system of a pre-computed map of the environment. We consider as available a reference dataset of posed images $(I_k)$ collected in the target area. Optionally, some methods can make use of an additional 3D model of the scene. We restrict ourselves to algorithms that operate only with a single query camera image as input.

First, we review the different existing approaches to solve this geometry problem. Section 2.2 presents Image Retrieval solutions, Section 2.3 introduces Structure-based approaches, including 2D-3D features matching pipelines, direct alignment methods and hierarchical approaches based on relative pose regression. Section 2.4 reviews the literature on direct-learning based methods, i.e. end-to-end deep learning approaches. Finally, Section 2.5 discusses the most relevant solutions for vehicle localization at large scale and paves the way for the research direction of the thesis. We refer to [Piasco et al., 2018] for an extensive review of this field.

## 2.2 Coarse localization with Visual Place Recognition

Visual Place Recognition (VPR) [Lowry et al., 2015, Pronobis et al., 2006] is a well-studied problem related to Visual Localization. It can be defined as "the ability to recognize one's location based on two observations perceived from overlapping field-of-views" [Garg et al., 2021]. Instead of directly computing a camera pose, the goal is to compare the query image to a large database of geo-referenced images and retrieve the most similar examples (i.e. the images which observe the same scene).

The VPR task is usually cast as an Image Retrieval (IR) problem, described in Figure 2.1. The image is encoded in a single global image descriptor which summarize the visual content in a compact vector. In an offline step, the descriptor of each reference image $(I_k)$ is computed and stored. Then during deployment, the query image $I$ is processed and its global image descriptor is matched against the reference database through nearest neighbour search. The global image descriptor was originally obtained by first computing sparse local descriptors such as SIFT [Lowe, 1999], followed by aggregation based on "visual words", which are clusters of the descriptors space. Bag Of Words [Sivic and Zisserman, 2009] counts the occurrences of each visual word, Word Spatial Arrangement [Penatti et al., 2014] takes into account their relative positions, while VLAD (vector of locally aggregated descriptors) [Jégou et al., 2010, Arandjelovic and Zisserman, 2013] and DenseVLAD [Torii et al., 2015a] sum the residuals w.r.t. clusters centroids. Recently, encoding an image into a global descriptor has been tackled by Convolutional Neural Networks followed by a pooling layer [Chen et al., 2022c]. The CNN is directly optimized for the task of place recognition with euclidian distance [Arandjelović et al., 2016], triplet loss [Gordo et al., 2017] or a listwise loss [Revaud et al., 2019]. The resulting representation can be improved by taking additional modalities into account such as depth information [Piasco et al., 2019b, Piasco et al., 2021], or semantics [Weng et al., 2021].

A camera pose approximation of a query image can be computed with the reference poses of the most similar retrieved images, for example with pose averaging [Torii et al., 2011]. These retrieved images can be re-ranked based on geographic constraints [Sattler et al., 2016]. However, these algorithms do not provide a precise estimation of the camera location but a coarse localization information. The accuracy

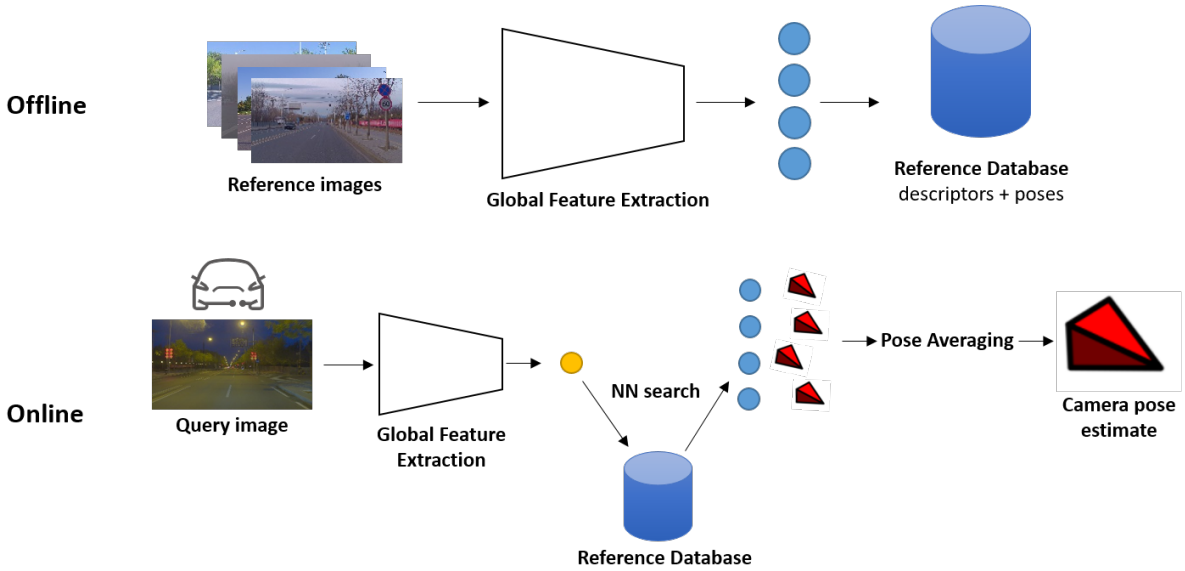


Figure 2.1: **Localization with image retrieval:** First, the global image descriptors of reference images are computed and stored in a database in an offline step. During navigation, the query image is processed and matched against the database. A localization estimate can be computed by pose averaging on the retrieved images.

depends on the spatial density of the reference images in the database, but using larger databases involves higher storage requirements and computation time during the localization process. As a result, when a sub-meter pose accuracy is required, the classical solution is to use image retrieval as a first step and to refine the coarse localization with other solutions [Humenberger et al., 2022]. These methods are presented in section 2.3.

## 2.3 Visual Localization with Structure-Based methods

Precise camera pose estimation can be computed from local features, a 3D model of the scene and projective geometry principles. This section reviews these algorithms, referred as "structure-based" in the literature. These pipelines usually combines several independent steps from the query image to the camera pose estimate. We assume to have access to a Structure-from-Motion 3D model, for example computed with COLMAP [Schönberger and Frahm, 2016]. It contains a sparse 3D point cloud of the scene. Each point has been observed and triangulated in at least 2 references images. These observations (the 2D locations of triangulated keypoints and their corresponding descriptors) are also stored.

When the target area is large, the first step often consists in an image retrieval step (described in section 2.2). Then local features are detected, described and then compared to the 3D model by 2D-3D features matching or direct features alignment. This hierarchical architecture is described in Figure 2.4.

### 2.3.1 Local features detection and description

Local features are points of interest in an image. They are characterized by a 2D location $(x, y)$ in the image coordinate system and a visual descriptor, i.e. a latent vector which represent the visual content

around the point. Local features are useful when the points they represent can be repeatedly detected and consistently described in various images captured from different viewpoints and lightning conditions. This property enables to establish local correspondences (see section 2.3.2) which are inputs of many geometry-based 3D computer vision algorithms [Hartley and Zisserman, 2003], including camera pose estimation. Points detection and description can be computed one after the other or jointly. This process is usually referred to (local) features extraction.

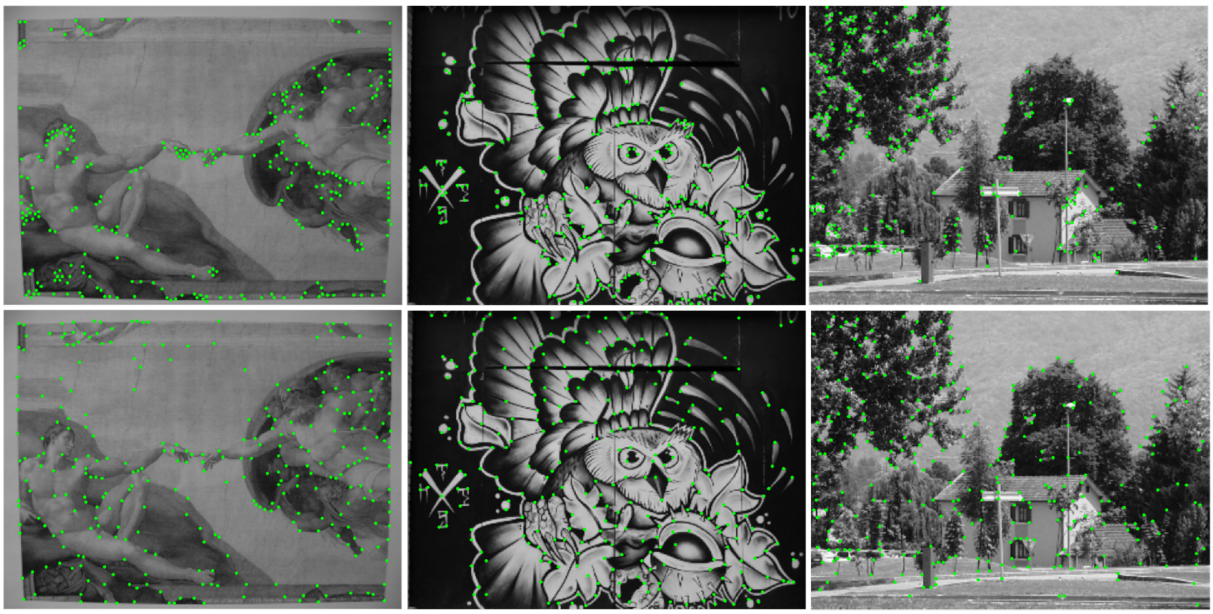

Figure 2.2: **Classical and learning-based keypoints detection:** We display and compare detected keypoints from the classical Harris corner detector [Harris and Stephens, 1988] (top) and SuperPoint [DeTone et al., 2018] (bottom).

**Detection:** Keypoints detection consists in locating "interesting" points in an image. Points of interest do not have a formal definition, but detectors usually aim to select areas with rich visual information, such as edges, corners or highly textured areas. Keypoints evaluation is a well-studied problem, where "repeatability" is an important factor [Schmid et al., 2000]. Harris corner detector [Harris and Stephens, 1988] has been a popular keypoint extractor before machine learning. Difference of Gaussian (DoG) has also been widely used [Lowe, 1999]. Since then, this task has been tackled by supervised learning from human annotated keypoints [Rosten and Drummond, 2006] or images captured from the same viewpoint with different appearances [Verdie et al., 2015]. SuperPoint [DeTone et al., 2018] replaces ill-defined human annotations by pre-training with synthetic data followed by a self-supervised objective based on homographic adaptation. In order to perform real-time computation, SuperPoint processes the full image in a CNN which outputs downscaled features maps with 65 channels. These channels correspond to keypoint probabilities of a 8x8 pixel grid with a "no keypoint" dustbin. The discrete set of detected keypoints on the full-sized image can be extracted from these features maps by non-maximum suppression. A qualitative comparison between classical and learning-based detected keypoints is shown in Figure 2.2

**Description:** Descriptors are a vector representation of the visual content of a local image patch. While hand-crafted descriptors such as SIFT [Lowe, 1999] and SURF [Bay et al., 2006] have shown great success, the focus has shifted in recent years to learn features description from large amounts of

visual data. Many learning-based formulations [Choy et al., 2016, Jahrer et al., 2008, Simo-Serra et al., 2015, Zagoruyko and Komodakis, 2015, Tian et al., 2019, Dusmanu et al., 2019] rely on siamese convolutional networks trained with pairs or triplets of images/patches supervised with local correspondences. Features extractors can be trained without annotated correspondences by augmenting 2 versions of the same image or using weak supervision. SuperPoint [DeTone et al., 2018] uses homographies while Novotny et al. [Novotny et al., 2018] leverage image warps. In a recent work, CAPS [Wang et al., 2020b] have shown that accurate correspondences between different views can be obtained using weak supervision through the use of relative camera poses.

### 2.3.2 Local Features Matching

The core idea of local features matching algorithms is to associate image pixels (2D coordinates in the image) with points from the 3D model (3D coordinates in the scene). Such a correspondence is called a 2D-3D match, and having several of them enables to compute the camera pose with projective geometry principles. Given a sparse set of 2D keypoints with descriptors extracted from the query image and a 3D model, we present the different approaches to obtain 2D-3D matches.

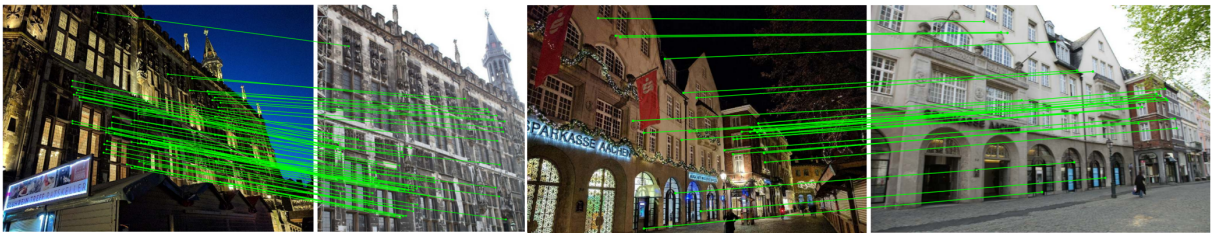

Figure 2.3: **Local correspondences between a queries and retrieved images** Inliers 2D-3D correspondences between query images (left) and reference images (right) of the HLoc method [Sarlin et al., 2019]. Local 2D-2D correspondences are upgraded to 2D-3D because the 3D position of reference keypoints are known.

With $n$ 2D keypoints and $m$ 3D points, the simplest approach to establish correspondences is to compute the $n \times m$ similarity matrix between all pairs. Then, a match can be defined by argmax similarity on either rows or columns. More robustly, only matches consistent in both directions can be kept (mutual matching). Lowe's ratio test [Lowe, 2004] can also be used to reject ambiguous matches. However, the brute force approach (computing the full similarity matrix) is not tractable for large scenes with many 3D points. Efficient data structure can be used such as vocabulary trees [Nister and Stewenius, 2006]. Active Search [Sattler et al., 2012] proposes a prioritized search strategy which enables fast and accurate features matching still considered as a strong baseline. Hierarchical approaches, such as HLoc [Sarlin et al., 2019] or MegLoc [Peng et al., 2021], are the current state-of-the-art: query features are not compared to the full 3D point cloud, but rather matched against the keypoints detected in reference images retrieved by VPR. Thus, the search space is highly reduced and the problem is brought back to 2D-2D sparse image matching (see Figure 2.3). Alternatively PnLP [Piasco et al., 2019a] proposes to extend 2D-2D matches to 2D-3D via relative depth estimation. Matching can be tackled by classical strategies aforementioned, but the current best approach, named SuperGlue [Sarlin et al., 2020] uses a Graph Neural Network (GNN) combined with an optimal matching layer. The GNN learns to encode 2 sets of keypoints with

descriptors coming from 2 different images by attentional aggregation, and the optimal matching layer finds the optimal partial assignment with the Sinkhorn algorithm. In contrast with simpler strategies, the matching process does not take only descriptors similarity to define matches, but also uses geometrical cues.

### 2.3.3 Camera Pose Estimation with Perspective-N-Points

The Perspective-N-points problem (PnP in short) determines the 6-DoF pose of a calibrated camera, given $n$ observed 3D points and their corresponding 2D projections in an image. It is a projective geometry problem where each 2D-3D correspondence point is a constraint. This minimal problem is well-defined starting from $n = 3$ points [Haralick et al., 1991]. Many solvers have been proposed to tackle related problems [Larsson et al., 2017b, Larsson et al., 2017a, Kukelova et al., 2011, Bujnak et al., 2008] problem and offer different trade-offs between accuracy and efficiency. Notably, EPnP [Lepetit et al., 2009] is a popular choice because of its $O(n)$ complexity.

The set of points provided by features matching pipelines always contains a portion of false (outliers) and true (inliers) correspondences. To avoid the negative impact of outliers on the PnP computation, a RANSAC [Fischler and Bolles, 1981] formulation is used to detect inliers, resulting in a robust camera pose estimation process.

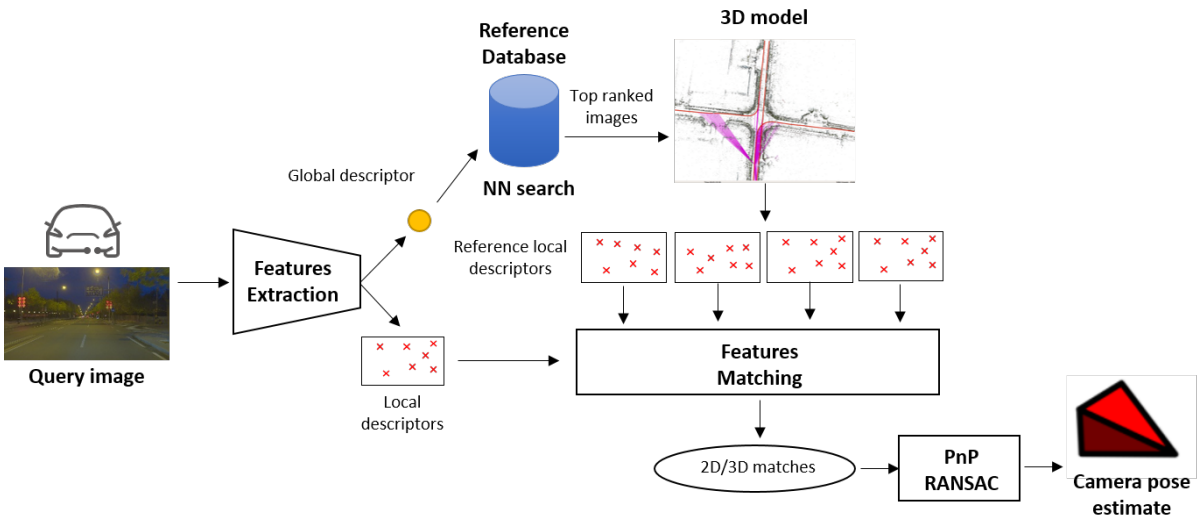


Figure 2.4: **Localization with hierarchical features matching pipelines:** After the retrieval step, local descriptors from the top ranked images are matched against the query images, resulting in 2D/3D correspondences, from which a precise camera pose can be computer with Perspective-N-Points + RANSAC.

### 2.3.4 Camera Pose Estimation with Direct Features Alignment

Camera pose estimation can also be computed by finding the transformation which aligns the query image and the 3D model. It can be performed by iteratively optimizing a cost function which measure the reprojection error at the current camera pose estimate. Image alignment was originally performed directly on image pixels intensities [Lucas and Kanade, 1981, Baker and Matthews, 2004]. Today, related visual localization methods rely on aligning learned local features. GN-Net [Von Stumberg et al.,

2020] learns weather-invariant deep features tailored for alignment with Gauss-Newton optimization, using images correspondences from visual SLAM. Recently, PixLoc [Sarlin et al., 2021] learns multi-scale features end-to-end for the visual localization task, by unrolling the Levenberg-Marquadt alignment optimizer. Replacing pixel intensities by deep features enable to increase robustness to appearance changes and to extend the convergence basin of these algorithms, which are known to be accurate but highly sensitive to the camera pose initialization.

### 2.3.5 Camera Pose Estimation with Relative Pose Regression

Finally, given a query and retrieved images in the reference database, the camera pose estimation problem can alternatively be solved by learning-based methods. Instead of relying on local features and epipolar geometry, hierarchical relative pose regression methods [Laskar et al., 2017, Balntas et al., 2018, Ding et al., 2019] leverage CNN to compute the relative camera pose between the two images. In this scenario, references RGB images need to be stored in memory instead of the 3D model of the scene, resulting in very large memory consumption in wide maps or dense datasets. Moreover, features-based methods and geometric solvers or alignment algorithms perform better than current state-of-the-art for learned relative pose regression.

### 2.3.6 Advantages and Drawbacks of Structure-Based methods

The 2D-3D features matching pipelines report the best localization accuracy on visual localization benchmarks [Sattler et al., 2018]. By determining local correspondences between the query image and the 3D map, a centimeter-level camera pose can be computed with well-established projective geometry algorithms.

However, structure-based methods present a high computational cost and large storage requirements. As a result, computing these algorithms in an embedded device is not trivial and requires many engineering efforts. The storage problem can be reduced with compression methods [Camposeco et al., 2019]. An other approach [Lee et al., 2021] consists in server-side computation by transferring the visual data from the device to a server which runs the localization algorithm.

## 2.4 Direct learning-based methods

To circumvent the complexity and drawbacks of structure-based solutions, direct deep learning methods have been developed. In this paradigm, the reference database collected in the environment is used to train a scene-specific neural network to provide localization information from images. Two main formulations exist: Absolute Pose Regression (APR) predicts translation and orientation of the camera directly from an image and Scene Coordinate Regression (SCR) predicts the world coordinates of the observed content. Because camera poses and scene coordinates are defined in the coordinate system of a given scene, these models require a new training before deployment on a new map. On the other side, this training enables to memorize the visual content of the map in the network weights such that no reference data such as an Image Retrieval (IR) descriptors database or a 3D model need to be stored during deployment. We present below the prior literature on these direct learning-based methods. This thesis contributed in this field in several ways, presented in the following chapters.

### 2.4.1 Absolute Pose Regression

Absolute Pose Regression (APR) or Camera Pose Regression is the simplest formulation for direct learning-based visual localization: a Deep Neural Network takes the query image as input and outputs the camera pose. In other words, the problem is tackled with end-to-end supervised regression. PoseNet [Kendall et al., 2015] is the pioneering work. It uses an encoder-decoder architecture where the encoder is a CNN pretrained on ImageNet [Deng et al., 2009] and the decoder regresses the pose with fully connected layers. Camera pose is modelled by a 7-dimensional vector, composed of 3 translation components and 4 rotations components which describe a quaternion. The neural network is optimized by minimizing a weighted sum of the translation and orientation errors. The architecture is illustrated in Figure 2.5.

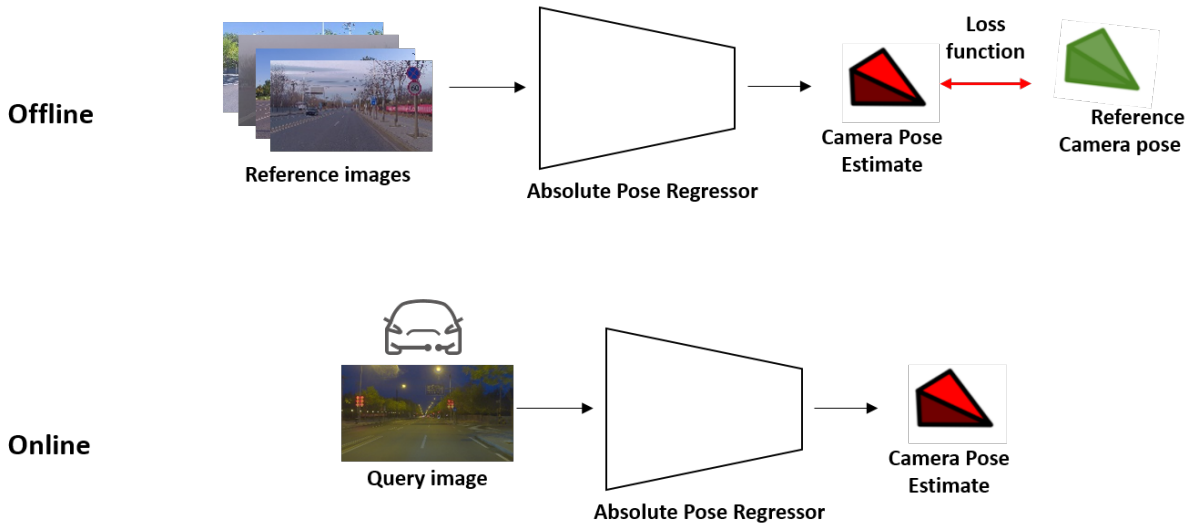


Figure 2.5: **Localization with Absolute Pose Regression:** The whole task in handled by a single neural network which regresses position and orientation of the camera. Before deployment, a scene-specific training is required.

Since then, many improvements have been proposed. Many works have focused on the neural network architecture. Concerning the encoder, the original GoogLeNet [Szegedy et al., 2015] has been replaced by more recent backbones such as ResNet [He et al., 2016]. The original decoder architecture is limited because the encoder features map are processed by a global average pooling which discard the spatial information about the features. Several solutions have been proposed, such as up-convolutions in a Hourglass architecture [Melekhov et al., 2017]. Atloc [Wang et al., 2020a] uses an attention-based module before the fully-connected layers. TransPoseNet [Shavit et al., 2021b] replaces CNN architecture by transformers.

Another research direction tries to leverage spatio-temporal constraints using consecutive video frames. This has first been done with a Long-Short Term Memory (LSTM) module [Walch et al., 2016]. VidLoc [Clark et al., 2017] and MapNet [Brahmbhatt et al., 2018] tackle it with siamese networks and relative pose supervision. GRNet [Xue et al., 2020] formulates the problem with Graph Neural Networks.

Concerning the loss function, the original PoseNet paper [Kendall et al., 2015] rely on a hand-crafted weighting between translation and orientation losses. However, it has been observed that the optimal

weighting is scene-specific and depends on the scene scale and the distribution of training poses. Consequently, the only way to obtain optimal localization accuracy with this loss function formulation is by manual hyperparameters tuning, which is expensive and time consuming. To solve this problem, authors from PoseNet have proposed a new way to optimize APR networks: instead of manually searching for a reasonable weighting of the losses, GeoPoseNet [Kendall and Cipolla, 2017] proposes to learn these weights dynamically during the training using homoscedatic uncertainties [Kendall et al., 2018], which improves significantly the accuracy.

Finally, BayesianPoseNet [Kendall and Cipolla, 2016] proposes to add uncertainty quantification on APR models via the Monte Carlo Dropout [Gal and Ghahramani, 2016] technique. RVL [Huang et al., 2019] uses Prior Guided Dropout on the input image for the same purpose. The posterior distribution of camera poses can also be quantified by a encoder-decoder probabilistic framework where the encoder performs variational inference and a MLP decodes poses after sampling in the latent space [Zangeneh et al., 2023]. Because APR sometimes provide unreliable localization predictions in difficult cases such as camera occlusion, uncertainty quantification enables to filter out these outliers predictions. This thesis made contributions on this aspect, presented and discussed in section 3.4.

Despite this amount of work, the localization accuracy of APR on standard visual localization benchmarks is lower than structure-based methods and on par with image retrieval baselines [Sattler et al., 2019]. The main reason for this limited accuracy is the lack of geometrical reasoning in the camera pose estimation process. During our research, we have conducted many experiments to evaluate and improve APR accuracy and present results which rehabilitate Absolute Pose Regression methods, by showing that they present a competitive accuracy when the training dataset is large and diverse.

The main advantage of Absolute Pose Regression is its computational efficiency during deployment. Once trained, a single forward pass on the network is sufficient to obtain a localization estimate. Storage requirements are also very compact. The original PoseNet paper [Kendall et al., 2015] reports a runtime of 5ms on a NVidia Titan Black GPU and 50MB for the stored weights. In addition to that, these metrics do not depend on the number of training images, whereas it grows linearly for image retrieval and quadratically for structure-based methods [Wu, 2013].

### 2.4.2 Scene Coordinate Regression

Scene Coordinate Regression (SCR) methods combine a direct learning-based approach with a geometric camera pose computation. Instead of predicting directly the camera pose, SCR regresses the 3D location of the local content observed in the image. In other words, Scene Coordinates Regression uses machine learning techniques to learn local correspondences between 2D image pixels and 3D scenes coordinates (see Figure 2.6). These 2D-3D correspondences enable to compute the camera pose by solving the Perspective-N-Points problem (see Section 2.3.3), which exhibits a higher precision than Absolute Pose Regression.

Seminal work on Scene Coordinate Regression rely on RGB-D images and use random forest to map RGB-D patches to the 3D coordinates [Shotton et al., 2013]. Since then, the scene coordinate regression pipeline has been adapted to RGB images processed by fully convolutional networks [Massiceti et al., 2017].

The natural approach to train these models is to supervise the scene coordinates with a pre-computed

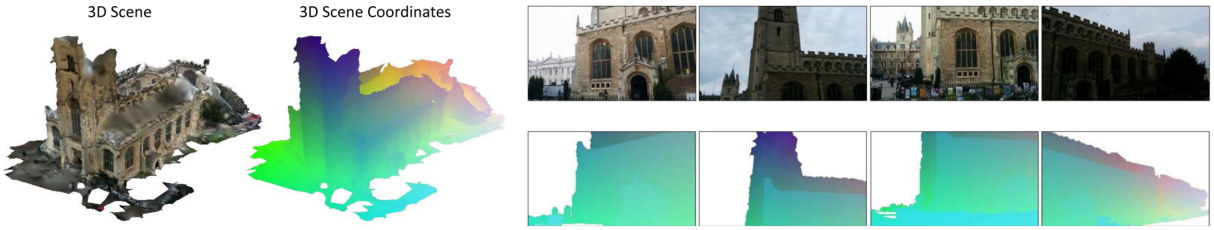


Figure 2.6: **Scene Coordinates Regression.** 3D scenes coordinates are extracted from a 3D model (left). SCR regresses the 3D coordinates of the visual content observable in the query image (right).

3D model of the scene. Because the PnP+RANSAC step is not differentiable, end-to-end learning from pixels to camera pose is usually not feasible. Replacing RANSAC by a differentiable counterpart based on probabilistic modelling, DSAC [Brachmann et al., 2017], enables to circumvent the problem and improve the accuracy. With this formulation, the model is even capable of learning the 3D model by itself only from camera pose supervision [Brachmann and Rother, 2018], at the cost of a reduced accuracy.

This class of methods exhibit higher accuracy than absolute pose regression and the efficiency enables real-time computation, however Scene Coordinate Regression approaches are limited to relatively small environments [Brachmann and Rother, 2021]. This problem can be mitigated by ESAC [Brachmann and Rother, 2019], which uses mixtures of expert to improve scaling to large environments.

## 2.5 Discussion: Which solutions for autonomous driving scenarios

In previous sections, we presented the different existing approaches for visual localization, which are summarized in Figure 2.7. Overall, two main classes of methods can be defined.

From one side, multi-step pipelines which involve (local and global) features extraction and features matching against a reference database before camera pose estimation. By combining the well-established features-based geometry techniques of classical computer vision with the latest learning-based modules (e.g. SuperPoint [DeTone et al., 2018] and SuperGlue [Sarlin et al., 2020]), they present state-of-the-art accuracy on all existing visual localization benchmarks.

On the other side, direct deep learning approaches rely on deep neural network to solve the entire task end-to-end. These more recent techniques are currently less popular than structure-based approaches for different reasons: APR is inaccurate, SCR does not scale and both require scene-specific training while one of the main advantages of deep learning approaches is their generalization properties.

However, considering the scenario described in Section 1.2, i.e. a growing amount of collected reference data and real-time deployment in the vehicle, we argue that direct learning-based approaches are better suited than structure-based methods to solve the vehicle localization problem at scale. We develop below several arguments to support this claim.

**Scaling up with large datasets.** Existing visual localization benchmarks [Sattler et al., 2016] [1] have primarily focused on aspects like long-term localization (Oxford RobotCar [Maddern et al., 2017]) or appearance change (Aachen Day-Night, 4seasons). The proposed reference datasets are usually limited to a few thousand images collected in 1-5 independent sequences. This data regime is sufficient to gen-

[1] visuallocalization.net

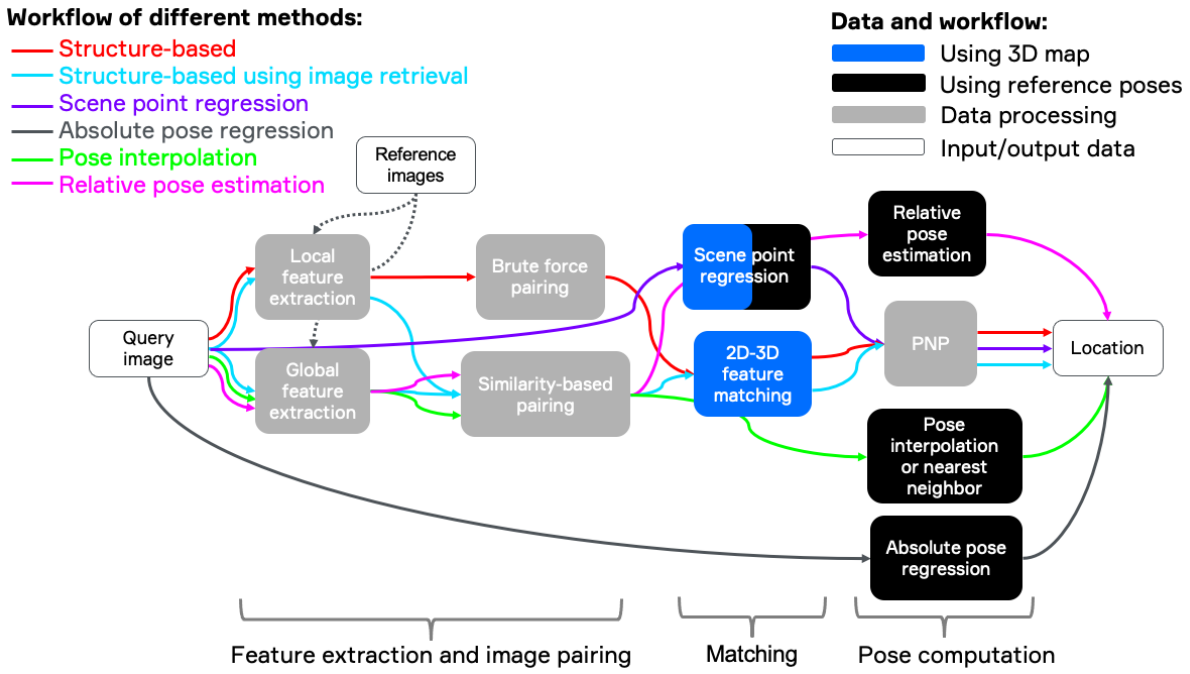


Figure 2.7: **Visual localization classes of methods:** A query image can be localized through different type of algorithms that rely either on local features, global features or direct regression. From [Humemberger et al., 2021].

erate high quality 3D reconstructions and perform accurate relocalization with structure-based methods. However, in the perspective of autonomous driving deployment, where a larger amount of data can easily be gathered, the effectiveness of visual localization methods has not been explored properly. In this scenario where the size of databases stored by structure-based methods increase, the computational cost of usual features matching pipelines can become intractable in practice. On the other hand, learning-based methods, which are not competitive on current benchmarks, should naturally benefit from large datasets and perform better. While additional computation will only occur on the offline training step, the online relocalization step will not be impacted and will remain efficient. One could argue that techniques could be developed to maintain compressed databases in high data regimes but we believe that pushing forward learning-based methods offers more perspectives due to the recent success of deep learning over the whole computer vision field.

**No assumption on visual content.** Because features-based methods rely on establishing local correspondences between observed keypoints, they make the assumption that a sufficient number of points of interest with rich visual features exist in the image. This is usually the case in urban areas where buildings and urban furniture offer rich and repeatable features. Countrysides, tunnels or parkings sometimes present uniform and low textured areas where establishing accurate local correspondences can be difficult, and for which the structure-based methods can totally fail. CNN rather rely on the global context in the image and then are more robust to this type of environments.

**Online efficiency.** Deep Neural Networks are expensive to train but inference is lightweight compared to more complex pipelines. Assuming that the provider of the autonomous driving service has access to a large amount of computation before deployment is reasonable. As a result, the trade-off of-

fered by direct learning-based methods (scene-specific training offline but efficient computation online) is well suited for the autonomous driving scenario.

The research conducted during this PhD thesis has first focused on evaluating the accuracy of Absolute Pose Regression on autonomous driving scenarios. Then, several contributions have been proposed to improve APR accuracy and its applicability to real-world scenarios. Chapter 3 first introduces several ways to inject geometry reasoning in Absolute Pose Regression models and then proposes a new way to add uncertainty quantification and develop a simple vehicle localization system based on temporal filtering of APR predictions.

Chapter 3

# Vehicle localization with Absolute Pose Regression

## Contents


**3.1 Résumé en Français** . . . . . **26**

**3.2 Motivation** . . . . . **27**

**3.3 Geometric inductive biases in Absolute Pose Regression networks** . . . . . **27**

3.3.1 Breaking translation invariance with CoordConvolutions . . . . . 27

3.3.2 Learning local confidences with weighted average pooling . . . . . 28

3.3.3 Intrinsic coordinates convolutions for crowd-sourced images . . . . . 29

**3.4 Uncertainty-aware Pose Regression** . . . . . **30**

3.4.1 Homoscedatic vs Heteroscedatic uncertainties . . . . . 31

3.4.2 Joint learning of pose regression and heteroscedatic uncertainty . . . . . 31

3.4.3 Localization under uncertainty . . . . . 32

**3.5 Absolute Pose Regression with CoordiNet** . . . . . **34**

**3.6 Experiments** . . . . . **34**

3.6.1 Comparison with related methods . . . . . 34

3.6.2 Evaluation on larger scale datasets . . . . . 36

3.6.3 Evaluation of localization under uncertainty . . . . . 37

3.6.4 Ablation studies . . . . . 39

3.6.5 Localization with crowd-sourced images . . . . . 40

**3.7 Conclusions** . . . . . **41**

## 3.1 Résumé en Français

Ce chapitre décrit les recherches menées pour développer des systèmes de localisation de véhicules à l'aide d'algorithmes de Pose Regression.

Nous proposons d'inclure des biais inductifs géometriques et la quantification de l'incertitude dans les réseaux de neurones utilisés. Pendant la navigation, nous filtrons temporellement les positions prédites et leurs incertitudes avec un EKF.

Nous explorons la capacité de notre système CoordiNet sur des benchmarks publics et de grands jeux de données. Nous observons une précision de localisation satisfaisante lorsque les données d'entraînement sont abondantes, mais aussi certaines limitations qui ont conduit aux contributions introduites dans les chapitres suivants.

## 3.2 Motivation

The research conducted in this thesis started with the simplest possible formulation: Absolute Pose Regression. This approach emerged with PoseNet [Kendall et al., 2015] during the rise of Deep Learning methods. Since then, many improvements have been proposed [Brahmbhatt et al., 2018, Walch et al., 2016, Melekhov et al., 2017, Huang et al., 2019, Wang et al., 2020a] but the approach remained unpopular in the visual localization field [Sattler et al., 2019]. This approach is indeed significantly less accurate than features-based methods (see section 2.3) in standard benchmarks and lacks geometrical reasoning. However, in the perspective autonomous driving, with real-time computation constraints and a growing amount of collected data, replacing complex multi-step pipelines by a deep neural network for the localization task is an appealing solution. Previous literature on this topic is limited to comparison of methods on limited scale benchmarks.

This chapter explores the capability of Absolute Pose Regression to be used as a vehicle positioning system. First, we propose several architectural improvements in section 3.3 to enhance the geometrical reasoning of APR networks. Then, we leverage uncertainty quantification to filter out unreliable predictions from the network over time in section 3.4. These contributions are combined in a new pose regression model, named CoordiNet [Moreau et al., 2022a], presented in section 3.5. Finally, we evaluate the proposed model in section 3.6. We start by comparing CoordiNet to relevant competitors on academic benchmarks. Then, we explore further the capability of this model on larger-scale autonomous driving datasets available publicly or specifically collected for this purpose in Paris and Shanghai. We conclude in section 3.7 by observing both benefits and limitations of Absolute Pose Regression in the context of autonomous driving, paving the way for the next contributions of the thesis.

## 3.3 Geometric inductive biases in Absolute Pose Regression networks

Visual localization is by nature a geometry problem. State-of-the-art solutions usually decompose the problem into small independent sub tasks combining features extraction with geometric reasoning such as pose estimation with PnP. In contrast with this idea, Absolute Pose Regressors rather rely on over-parametrized deep learning models. With this formulation, the Pose Regressor network needs to learn the entire task end-to-end from available data without prior knowledge of real-world geometry principles involved in the localization problem. Prior literature [Sattler et al., 2019] has stated that this is one crucial limitation of APR models and the main reason of their poor accuracy. In addition to that, Absolute Pose Regression usually assumes that all training and test images are captured by a unique calibrated camera, but the crowded-source data scenario described in section 1.2 could involve training with heterogeneous cameras.

In this section, we focus on improving the formulation of Absolute Pose Regression CNNs by embedding geometrical inductive biases in their architecture. The benefits of the proposals of this section are measured with ablation studies in section 3.6.4.

### 3.3.1 Breaking translation invariance with CoordConvolutions

The architectural improvements of Convolutional Neural Networks (CNN) are usually measured on image classification benchmarks such as ImageNet [Deng et al., 2009]. One main reason of CNN success

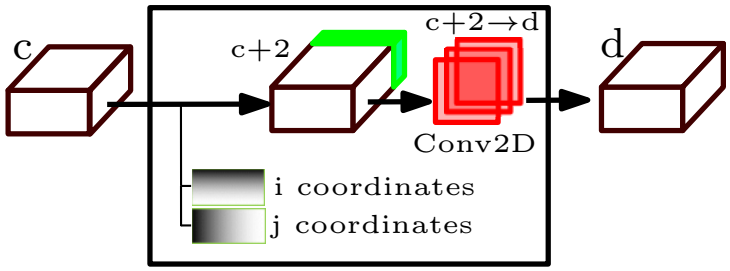


Figure 3.1: **CoordConvolutions:** additional channels with 2D pixels coordinates are concatenated before the convolution operation.

for image processing tasks is their translation invariance property: instead of learning dense connections between all pixels at each layers, we restrict to local convolution kernels applied on features maps in a sliding window fashion. As a result, the same processing is applied to each image local patch without taking into account their position in the 2D image space.

Because the image classification output does not depend on the object location, translation invariance is a beneficial property which increases robustness. However, in the visual localization problem, it is clear that the 2D position of visual content in the image is a very important feature to estimate the camera pose, because explicit solvers take 2D-3D local correspondences as input.

Interestingly, fully convolutional neural networks have been observed to be unable to solve very simple toy problems related to pixels 2D location [Liu et al., 2018], such as supervised coordinate classification which consists in taking a black image with one white pixel as input and asking the network to predict the coordinate of the white pixel. The authors propose a simple fix to resolve this problem: the CoordConv layer, which acts as a replacement of the traditional convolutional layer.

CoordConv concatenates 2 additional channels to the input tensor, that contain (x,y) pixel coordinates in the image coordinate system before applying the convolution, as illustrated in Figure 3.1. This way the convolutional layer can take the spatial position into account in the processing. The additional computational cost brought by this layer is negligible in deep neural networks.

We observed that including CoordConv in the last layers of an Absolute Pose Regressor was significantly improving the accuracy, as shown in section 3.6.4.

### 3.3.2 Learning local confidences with weighted average pooling

The output of Absolute Pose Regressors is the camera pose, usually represented by a 1-dimensional vector of 7 components (3 for translation and 4 for orientation). When the output of a CNN is such a vector, for instance in image classification, a global pooling is used to aggregate the 3-dimensional $H \times W \times D$ tensor into a vector of size $D$. Common solutions use simple heuristics such as Global Average Pooling (GAP) or Global Max Pooling (GMP). With the addition of the previously introduced CoordConv, we expect the feature maps to contain diverse features distributed across the spatial dimensions. Thus, we hypothesize that using GAP or GMP layers after CoordConv is counter-productive for the accuracy because they lack expressiveness to aggregate spatial information distributed across the image.

We take inspiration from a CNN architecture, FC4 [Hu et al., 2017] designed for the Auto White Balance task. In this paper, a variant of Global Average Pooling, named Confidence-Weighted Average

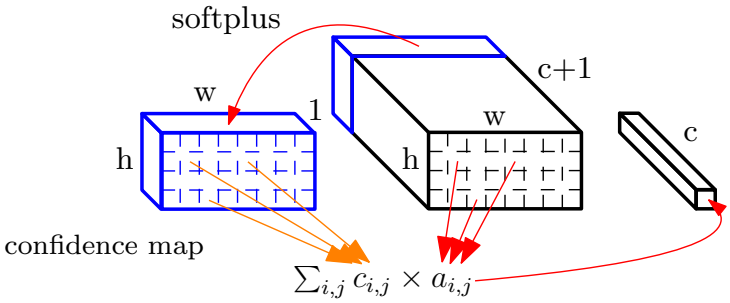


Figure 3.2: **Confidence-Weighted Average Pooling:** One channel of the last feature map is used as confidence weights to perform aggregation of local estimates in a single vector.

Pooling (CWAP) is introduced. In order to output a d-dimensional output vector, CWAP uses features map with $d+1$ channels. The last channel is used as a local confidence map used to perform a weighted average pooling. Softplus activation is applied to the confidence map to ensure that confidence weights are positive. These weights are predicted according to previous layer activation so we can compare this computation to a low-cost self-attention mechanism. This layer is illustrated in Figure 3.2

Examples of confidence map activations are shown in figure 3.3. We observe that on small scenes of Cambridge Landmarks, the pooling always highlights the same object regardless of camera pose (here front of Kings College). On larger scenes, there is no common object visible in all the scene. In this case, the pooling masks areas where dynamic objects appear.

### 3.3.3 Intrinsic coordinates convolutions for crowd-sourced images

Publicly available datasets commonly used for visual localization [Maddern et al., 2017, Kendall et al., 2015, Shotton et al., 2013] are recorded with a single camera. During our experiments, we noticed that an APR model was not able to predict an accurate pose from images captured by a different camera, or even from images from the same camera but resized or cropped.

However, being able to adapt to new cameras would be a great practical advantage, for example for being able to train from crowded-sourced data, or deploy the algorithm on a new sensor stack. We assume

Figure 3.3: **Visualization of pooling activations:** bottom images are inputs of the model, top images are inputs multiplied by the upsampled confidence map of the pooling layer

a pinhole camera model and that all cameras are calibrated (i.e. we have access to the intrinsics matrix $K$). In practice, this is not a constraint because these parameters are estimated by the SfM mapping process [Schönberger and Frahm, 2016].

One solution is to condition the network on the camera parameters. Inspired by CoordConv (see sec 3.3.1), we propose to inject local information about camera rays as additional channels in convolution layers. More specifically, we compute for each pixel $(i, j)$ 2 coordinates $r_i, r_j$ which correspond to the tangents of the angles between the camera ray and the ray reaching the optical centre $(cx, cy)$:

$$r_i = (i - c_x)/f_x$$

$$r_j = (j - c_y)/f_y$$

where $(f_x, f_y)$ are the focal lengths. Similar to the CoordConv, we concatenate these coordinates to latent feature maps before computing the convolution. Because these coordinates parameterize the orientation of camera rays (i.e. deviation from the camera orientation), they do not depend anymore on the specific camera which have been used to capture the image. Using this formulation prevents the network to overfit on a specific camera, and enables to perform inference on a never observed (calibrated) camera.

We conduct experiments where we replace all convolutions in a APR network by the proposed camera-aware layer on section 3.6.5. After a preliminary research phase, we discovered that a highly similar method has been developed for the monocular depth estimation task and published under the name CAM-Convs [Facil et al., 2019].

## 3.4 Uncertainty-aware Pose Regression

During the vehicle navigation, unexpected situations can arise that cause the visual localization algorithm to fail: strong occlusions of the camera, difficult lightning conditions such as sun flares, modification of the environment such as roadworks, etc. Such occlusions are shown in Figure 3.4. In order to make a robust localization system, one needs to know when the predicted pose is not reliable.

Computing uncertainty coupled with pose regression is a common way to handle this problem [Kendall and Cipolla, 2016]. However, widely used approaches to compute uncertainty for pose regression have limitations for practical applications. They generate multiple hypothesis for each single image at inference time, and then compute mean and variance to estimate pose and uncertainty [Kendall and Cipolla, 2016, Huang et al., 2019]. This increases a lot computational complexity because several inferences are

Figure 3.4: **Unavoidable failure cases** Dynamic objects generate occlusions that results in failure cases (samples from Oxford RobotCar dataset [Maddern et al., 2017]).

required. For practical applications, one needs uncertainty to be estimated jointly with the pose regression and such estimated uncertainty should be highly correlated to a potential level of errors of regressed poses.

To address this requirement, we propose to predict uncertainty from activations of a hidden layer, so it can be jointly computed at inference time with pose regression and learn to associate potential failures with content of input image.

### 3.4.1 Homoscedatic vs Heteroscedatic uncertainties

In Bayesian Deep Learning, there are 2 types of uncertainty one can model [Kendall and Gal, 2017]:

- Epistemic uncertainty (or model uncertainty) is the uncertainty on the weights of the network. It measures the degree of knowledge of the network about input data. A simple way to approximate it is Monte Carlo Dropout [Gal and Ghahramani, 2016] (MCD): train a network with dropout layers and keep them active at inference time. Several inference on the same data will provide a sample of results. The mean is used as prediction and the variance is interpreted as epistemic uncertainty. Bayesian PoseNet [Kendall and Cipolla, 2016] uses this method to estimate uncertainty of pose regression. RVL [Huang et al., 2019] proposes a prior guided dropout on input image: it removes areas where dynamic objects appear and allows to generate a Monte Carlo sample too.

- Aleatoric uncertainty (or data uncertainty) is the variance of the network output which can be caused by noise in input data. We can choose between heteroscedatic aleatoric uncertainty which can vary with input data and homoscedatic uncertainty which measures the overall variance in the outputs of a task as it is not dependent on input data. Heteroscedatic can be predicted directly from the network, as shown in [Lakshminarayanan et al., 2017, Kendall and Gal, 2017], by using a maximum likelihood loss function. This formulation can be extended to multivariate uncertainty, as shown by [Russell and Reale, 2021] where correlation between outputs variables is learned. Finally, HydraNet [Peretroukhin et al., 2019] combines epistemic and aleatoric uncertainties to provide consistent orientation estimates.

We decide to learn heteroscedatic uncertainty as an auxiliary task during the training of our model. Thus, we obtain an uncertainty estimate at test time for less computation than with MCD.

### 3.4.2 Joint learning of pose regression and heteroscedatic uncertainty

Pose regression problem can be modeled by the equation $Y = f_W(I)$, where $f$ is the neural network with weights $W$, $Y$ is the predicted pose and $I$ is the input image. In Bayesian Deep Learning [Kendall and Gal, 2017], $W$, $I$ and $Y$ are considered as random variables and uncertainty is the variance of these variables.

In our problem, $Y \in SE(3)$. We decompose the 6-DoF pose with a 3D translation vector $[T_x, T_y, T_z]$ and a unit quaternion $[Q_x, Q_y, Q_z, Q_w]$ for rotation.

For translation, we model $T_x, T_y, T_z$ as 3 independent Gaussian variables centered on the actual pose with variances $\sigma^2_{T_x}, \sigma^2_{T_y}, \sigma^2_{T_z}$. Our framework predicts these $\sigma^2$ uncertainties, which represent the expected noise in the output pose, depending on input image. As we made it data dependent, it is called

heteroscedatic uncertainty. In practice, each uncertainty is learned relatively to a loss function. In our case, we want to have one uncertainty estimate for each translation component, so we use 3 separate $L_1$ translation losses $L_{T_x}$, $L_{T_y}$, $L_{T_z}$.

For rotation, we can not use the same formulation, first because individual components of a unit quaternion are clearly not independent, but also because the 3D rotation group $SO(3)$ is not euclidean. As a result, we optimize rotation with a single loss function $L_R$, resulting in a single rotation uncertainty estimate $\sigma_R^2$. An other manner to learn uncertainty for rotation have been proposed in [Peretroukhin et al., 2019], but we found out that in practice our 1-dimension uncertainty estimation work well and leave the integration of multivariate rotation uncertainty estimation in pose regression for future work. Our choice for $L_R$ is the geodesic distance between rotations, defined as the minimal angular difference between 2 rotations:

$$L_R = cos^{-1}((tr(M_{pred}M_{GT}^{-1}) - 1)/2) \tag{3.1}$$

where $M_{pred}$ and $M_{GT}$ are predicted and reference 3D rotations, converted to rotation matrices. [Zhou et al., 2019] has shown that this optimization objective performs better than $L_2$ loss.

Finally, we combine these 4 loss functions by minimizing the negative log-likelihood of our model. We do not learn $\sigma^2$ directly but use $s = \log\sigma^2$ for numerical stability, following [Kendall and Cipolla, 2017]. Our final optimization objective becomes:

$$L_\sigma(I) = \sum_{i\in[T_x,T_y,T_z,R]} L_i(I)e^{-s_i(I)} + s_i(I) \tag{3.2}$$

This loss function is actually the same than the *learnable weights pose loss* [Kendall and Cipolla, 2017], except that uncertainty values are outputs of the network instead of free scalar values (homoscedatic uncertainty).

To minimize this loss function, the network needs to learn accurate poses in order to decrease $L_i$ losses. When a challenging image is provided, the network can predict high uncertainties in order to reduce the weights of regression losses in the objective. The second term acts as a penalization term to avoid infinite uncertainties. The best way to minimize this cost is to predict uncertainties proportional to loss values.

This method also has desirable effects for training. When used with homoscedatic uncertainty, each individual loss will contribute to the final loss with approximately the same weight. As our uncertainties vary with input data, this property is extended at a data level : each training sample will contribute to the batch loss with an approximately equal weight, whereas usually large errors contribute more. Contrary to the object classification problem where samples having larger error should be main target for optimizing the network, we want every sample to be considered equally important in the optimization process.

### 3.4.3 Localization under uncertainty

At test time, we fuse together the regressed pose with learned uncertainties in order to filter out failure cases and obtain a smooth and temporally consistent trajectory. This is a desirable property in autonomous driving and robotics applications, because localization could be directly used by the planning algorithm to compute control command.

We propose to use an Extended Kalman Filter (EKF) with an omnidirectional motion model for this fusion step. Integration is done by providing only the absolute pose measurement given by the network to the filter. We attach a simplified diagonal covariance matrix $\Sigma$ to each measurement, defined by:

$$\Sigma = I_6 * \left[\sigma_{T_x}^2 \sigma_{T_y}^2 \sigma_{T_z}^2 \sigma_R^2 \sigma_R^2 \sigma_R^2\right]^t .$$

This formulation is limited to represent uncertainty in $SE(3)$. First because in practice the covariance between variables can be non-zero. Another limitation discussed earlier is the use of a one dimensional rotation uncertainty $\sigma_R^2$. We tried to use more sophisticated formulations, where non-diagonal coefficients are learned, inspired by [Russell and Reale, 2021, Peretroukhin et al., 2019], but observed a lower pose regression accuracy with this formulation. One reason could be that vectors in equation 3.2 are replaced by matrices, leading to a lower numerical stability during training. This model of covariance matrix in $SE(3)$ could be improved as a future work. However, we show in 3.6.3 that our proposal is sufficient in practice to reach our target: a consistent trajectory where outliers are filtered.

**Uncertainty calibration:** During the evaluation of our method, we observed that learned uncertainties often underestimate the actual error (see 3.9). This is caused by overfitting: at the end of the training procedure, the model performs very well on training images and the uncertainty layer learns a distribution of errors which does not represent the actual distribution in test conditions. To mitigate this effect, we propose a 2 steps training procedure: available training data is split in a training set and a validation/calibration set. We first train the model with the training set with the procedure described in 3.4.2, and then fine-tune the uncertainty layer on the calibration set while all other layers are freezed. This enable to calibrate uncertainties on examples representative of test conditions. Since the use of a validation set is a common machine learning practice, this calibration step does not make our method more data-intensive than others. We show in figure 3.5 the benefit of uncertainty calibration for proper uncertainty estimation on a test sequence.

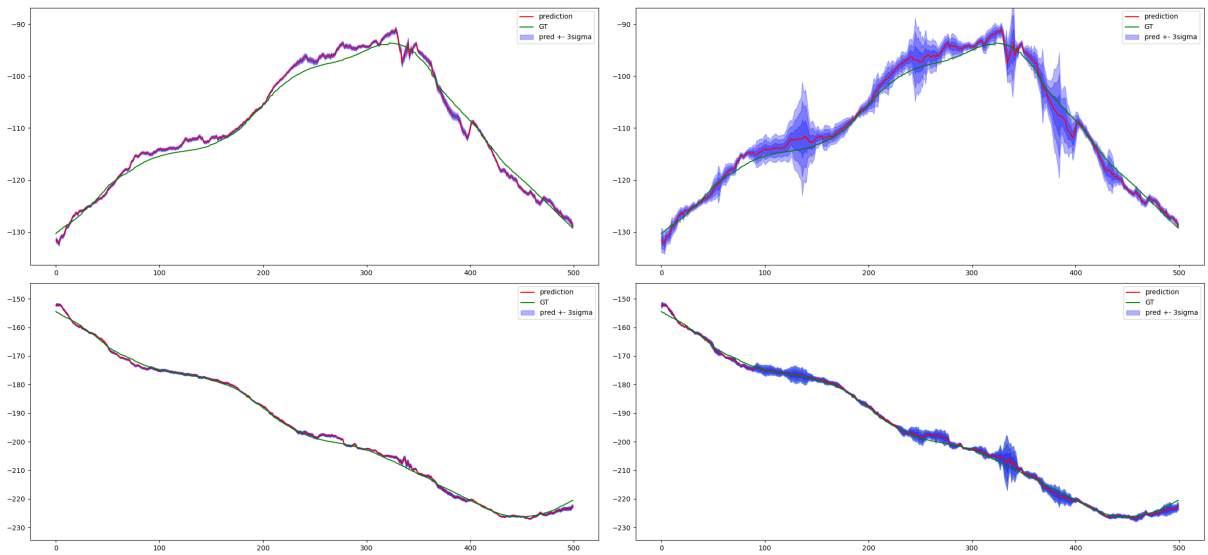


Figure 3.5: **Calibrated uncertainties.** Comparison between uncalibrated (left) and calibrated (right) uncertainties, plotted for x (top) and y (bottom) axis.

## 3.5 Absolute Pose Regression with CoordiNet

This section introduces our proposed Absolute Pose Regression model, named CoordiNet, which combines the geometric inductive biases presented in section 3.3 and the heteroscedatic uncertainty quantification presented in section 3.4. The fully convolutional architecture of the model is detailed in Figure 3.6.

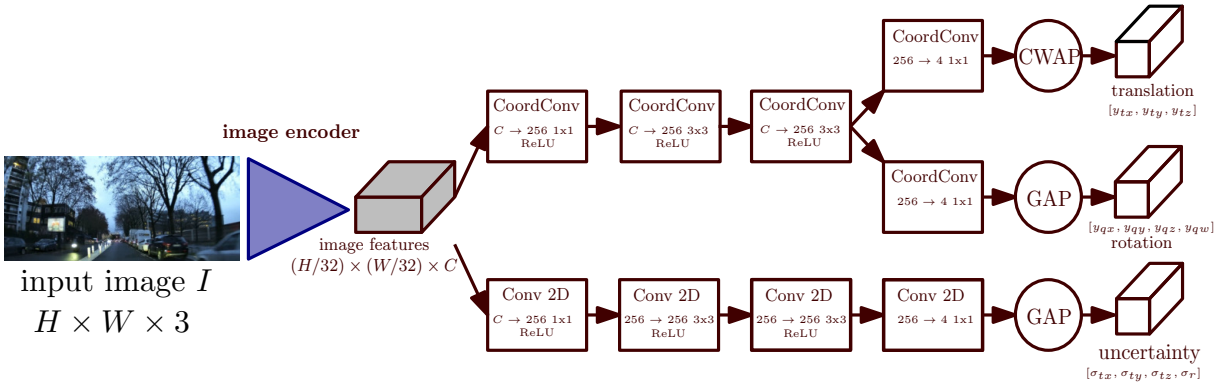


Figure 3.6: **CoordiNet architecture:** the input image is sent to a pretrained image encoder, then pose and uncertainty are predicted from encoder features in two separate decoders. Pose decoder uses Coord convolutions layers. GAP and CWAP refer to Global Average Pooling and Confidence Weighed Average Pooling.

The RGB input image is first processed by a backbone encoder, we used ResNet [He et al., 2016] and EfficientNet [Tan and Le, 2019] in our experiments. Then, unlike PoseNet [Kendall et al., 2015], our decoder does not use fully-connected layers but CoordConv layers and CWAP pooling (see section 3.3). We actually train 2 separate decoders, one for the pose estimation and the other for uncertainty quantification. This is actually an important design choice because it enables to fine-tune the uncertainty decoder while keeping the other parameters of the network freezed during the calibration step described in 3.4.3.

## 3.6 Experiments

### 3.6.1 Comparison with related methods

We compare CoordiNet to state-of-the-art Absolute Pose Regression networks. We use both median error and mean error as metrics. Mean error has the drawback of being corrupted by outliers (which can be very large because we estimate absolute pose in kilometers scale maps), while median error is usually more relevant to measure the accuracy of an algorithm.

#### 3.6.1.1 Oxford Robotcar:

First, we compare our method to related works on Oxford Robotcar dataset [Maddern et al., 2017]. This is an autonomous driving dataset collected with a camera mounted on the top of a vehicle. 2 separate scenes are considered : Loop is a 1km map in residential areas, while Full goes into the entire city in a 9.8 kms long course. The camera pose groundtruth provided by [Maddern et al., 2020] presents a meter-level accuracy and is sometimes erroneous. While many recordings are available in the original dataset,

the relocalization benchmarks done in the research community only use minimal train and test datasets. We reproduced the experiments done by RVL [Huang et al., 2019]. The model is trained on 2 scenes (Full and Loop) using only 2 training sequences in each case. While Full is tested on one sequence, 2 sequences are used for Loop. This benchmark is particularly challenging because the area is very large and APR methods can fail to generalize with only 2 different sequences in the training set.

We resize input images to $256 \times 455$. We train our model with ResNet34 encoder for a fair comparison with previous methods, and keep a fixed learning rate of $1e^{-4}$. We compare CoordiNet to other monocular methods, but also to sequential methods using several images as input.

Table 3.1: **Absolute Pose Regression methods on Oxford RobotCar dataset.** CoordiNet is compared to other APR methods. The pose considered is the raw network output without post processing methods. Empty cells correspond to results non reported on the related papers.

| | **Mean error comparison** | | | **Median error comparison** | | |
|---|---|---|---|---|---|---|
| **Method** | **Loop** | **Full** | **Average** | **Loop** | **Full** | **Average** |
| **CoordiNet** | 4.15m / 1.44° | 14.96m / 5.74° | 9.56m / 3.59° | 2.27m / 0.86° | 3.55m / 1.14° | 2.91m / 1.00° |
| AD-MapNet | 6.45m / 2.98° | 19.18m / 4.60° | 12.82m / 3.79° | | | |
| AtLoc+ | 7.53m / 3.61° | 21.0m / 6.15° | 14.27m / 4.88° | 4.06m / 1.98° | 6.40m / 1.50° | 5.23m / 1.74° |
| AtLoc | 8.73m / 4.63° | 29.6m / 12.4° | 19.17m / 8.52° | 5.36m / 2.10° | 11.1m / 5.28° | 8.23m / 3.69° |
| AD-PoseNet | 6.40m / 3.09° | 33.82m / 6.77° | 20.11m / 4.93° | | | |
| MapNet | 9.29m / 3.34° | 44.61m / 10.38° | 26.95m / 6.86° | | | |
| PoseNet | 7.9m / 3.53° | 46.61m / 10.45° | 27.26m / 6.99° | | | |

Quantitative results are shown in Table 3.1 and trajectories are displayed in Figure 3.7. We observe that our proposal CoordiNet presents lower median and mean errors than previous Absolute Pose Regression methods on both Loop and Full scenes and for both translation and orientation. However, the absolute errors remain very high and not comparable to structure-based methods which present a sub-meter accuracy. Similar to related APR methods, we observe many pose predictions outside of the roads on the Full scene, which are due to failure cases when the network hesitates between 2 different areas of the map.

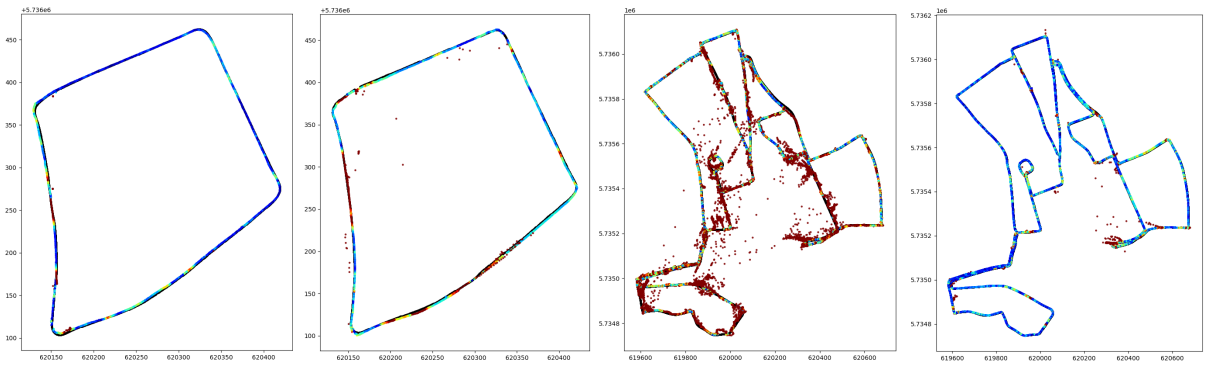

Figure 3.7: **Trajectories on Oxford Robotcar.** Left images are Loop results (2014-06-23-15-36-04, 2014-06-26-08-53-56) trained with 2 sequences, right images are Full results (2014-12-09-13-21-02). Middle right is trained with 2 sequences, right is trained with 15. Color map represents errors at a given location: blue is $\sim 1m$ error and red is $> 5m$ error.

#### 3.6.1.2 Cambridge Landmarks:

Then, we also report CoordiNet performances on Cambridge Landmarks. This dataset contains several small outdoor scenes with small training datasets. We train our model with both ResNet34 and EfficientNet-b3, and again compare CoordiNet to monocular and sequential methods. Images are computed at full resolution ($640 \times 350$). Results are reported in table 3.2. Compared to other monocular methods, CoordiNet reports best results on all scenes.

Table 3.2: **Absolute Pose Regression methods on Cambridge Landmarks dataset.**

| Method | Backbone | Old Hospital | Kings College | StMarysChurch | Shop Facade | Average |
|---|---|---|---|---|---|---|
| **CoordiNet** | EffNet b3 | 0.97m / 2.08° | 0.70m / 0.92° | 1.32m / 3.56° | 0.69m / 3.74° | 0.92m / 2.58° |
| **CoordiNet** | ResNet34 | 1.43m / 2.86° | 0.80m / 1.22° | 1.32m / 4.10° | 0.73m / 4.69° | 1.07m / 3.22° |
| LSTM-Pose | | 1.51m / 4.29° | 0.99m / 3.65° | 1.52m / 6.68° | 1.18m / 7.44° | 1.30m / 5.51° |
| PoseNet | ResNet34 | 2.17m / 2.94° | 0.99m / 1.06° | 1.49m / 3.43° | 1.05m / 3.97° | 1.43m / 2.85° |
| AD-PoseNet | ResNet34 | non reported | 1.3m / 1.67° | 2.28m / 4.80° | 1.22m / 4.64° | / |
| MapNet | ResNet34 | 1.94m / 3.91° | 1.07m / 1.89° | 2.00m / 4.53° | 1.63m / 4.22° | 1.66m / 3.64° |
| Bay. PoseNet | | 2.57m / 5.14° | 1.74m / 4.06° | 2.11m / 8.38° | 1.25m / 7.54° | 1.92m / 6.28° |

Still, the accuracy of APR methods is far behind approaches where the camera pose is computed by PnP [Sarlin et al., 2019, Brachmann and Rother, 2021]. Direct regression seems able to interpolate between observations but unable to extrapolate to non-observed camera poses with geometric reasoning, as outlined in [Sattler et al., 2019]. This property limits the accuracy on Cambridge Landmarks, where test images are captured at different locations compared to training trajectories.

### 3.6.2 Evaluation on larger scale datasets

Our interest is to use CoordiNet for localization in large environments for practical applications. Publicly available benchmarks are limited to 2-3 videos used as reference/training images. In scenarios related to development of localization functions for self-driving cars, one could rely on much larger scope of data available. In this section, we explore how well CoordiNet scales given amount of data provided for training is an order of magnitude higher compared to public benchmarks.

To do so, we use Oxford RobotCar dataset with a larger experiment : 15 sequences in training data resulting in 890k images in training set, 2 sequences used as validation and 7 test sequences, including the benchmark test sequence.

We also collected videos in Paris and Shanghai areas using dashcam cameras. We recovered ground truth poses from videos thanks to a large scale 3D reconstruction pipeline based on COLMAP SfM software [Schönberger and Frahm, 2016]. We provide examples from these datasets in figure 3.8 and we briefly introduce them:

- **Shanghai dataset**: a road of 3.3km, that contains highway and smaller roads. Very busy traffic observed in most of sequences. Training set: 12 sequences with 123k images. Validation set: 6 sequences with 32k images. Test set: 9 sequences with 48k images.

- **Paris dataset**: A loop of 1.9km in urban area. One part along the Seine is challenging because of mirror reflection in buildings, the other goes through complex intersections with sometimes busy traffic. Training set: 6 sequences with 148k images. Validation set: 2 sequences with 30k images. Test set: 2 sequences with 13k images.

We train CoordiNet on these 2 datasets in addition to Oxford, but also use available implementation of RVL [Huang et al., 2019] to compare performances with related methods (AD-PoseNet) and report results in table 3.3.

Table 3.3: **Results on large scale datasets.**

| | | **CoordiNet (ours)** | **AD - PoseNet** |
|---|---|---|---|
| **Oxford** | median | **1.53m / 0.46°** | 7.91m / 1.13° |
| | mean | **7.11m / 2.93°** | 19.89m / 4.51° |
| **Shanghai** | median | **0.69m / 0.69°** | 9.24m / 0.47° |
| | mean | **0.90m / 0.87°** | 11.78m / 1.49° |
| **Paris** | median | **0.29m / 0.29°** | 3.75m / 1.03° |
| | mean | **0.51m / 0.44°** | 5.11m / 1.25° |

We see that CoordiNet outperforms previous SOTA pose regressor on large areas by an order of magnitude and observe that larger training sets allow to reach sub-meter accuracy on test data. Enlarged Oxford training set from 2 to 15 sequences enables to decrease mean error from 9.56m to 1.94m and median error from 3.55m to 1.25m on the same test sequence, as shown in Figure 3.7. Not surprisingly, using larger training datasets makes a huge difference in the accuracy of Absolute Pose Regression, especially for our model CoordiNet. We obtain a sub-meter accuracy on Shangai and Paris dataset which is a good localization accuracy for self-driving applications.

### 3.6.3 Evaluation of localization under uncertainty

Rather than the raw CNN results, we are interested in the performance of the method outlined in 3.4.3, where poses and uncertainties are fused temporally into an EKF. We use the implementation provided by the ROS **robot_localization** package with default parameters. We demonstrate superiority of our learned uncertainty over a fixed baseline using three experiments:

- EKF with fixed covariance values,
- EKF with non-calibrated CoordiNet covariance values,
- EKF with calibrated CoordiNet covariance values.

Figure 3.8: **Samples from Paris (top) and Shanghai (bottom) datasets.**

We report the results of these experiments on a full run of the Paris dataset in Figure 3.9. We evaluate a smoothness score $s$ of the trajectory (the lower the better) by computing the norm of the difference between two consecutive unitary directional vectors:

$$s = \frac{1}{N-2} \sum_{t=0}^{N-2} \left\| \frac{T_{t+2} - T_{t+1}}{\|T_{t+2} - T_{t+1}\|} - \frac{T_{t+1} - T_t}{\|T_{t+1} - T_t\|} \right\|$$

As expected, coupled with an EKF the final trajectory gets smoother and the maximum error on the run is reduced. By rejecting outliers, the EKF reduces most of the time the mean error compared to the raw poses. We also show that it is crucial to estimate good covariance values in order to obtain the optimal trade-off between accuracy and smoothness: Coordinet + EKF with calibrated covariance performs the best in this experiment compared to fixed covariance values and to the baseline version.

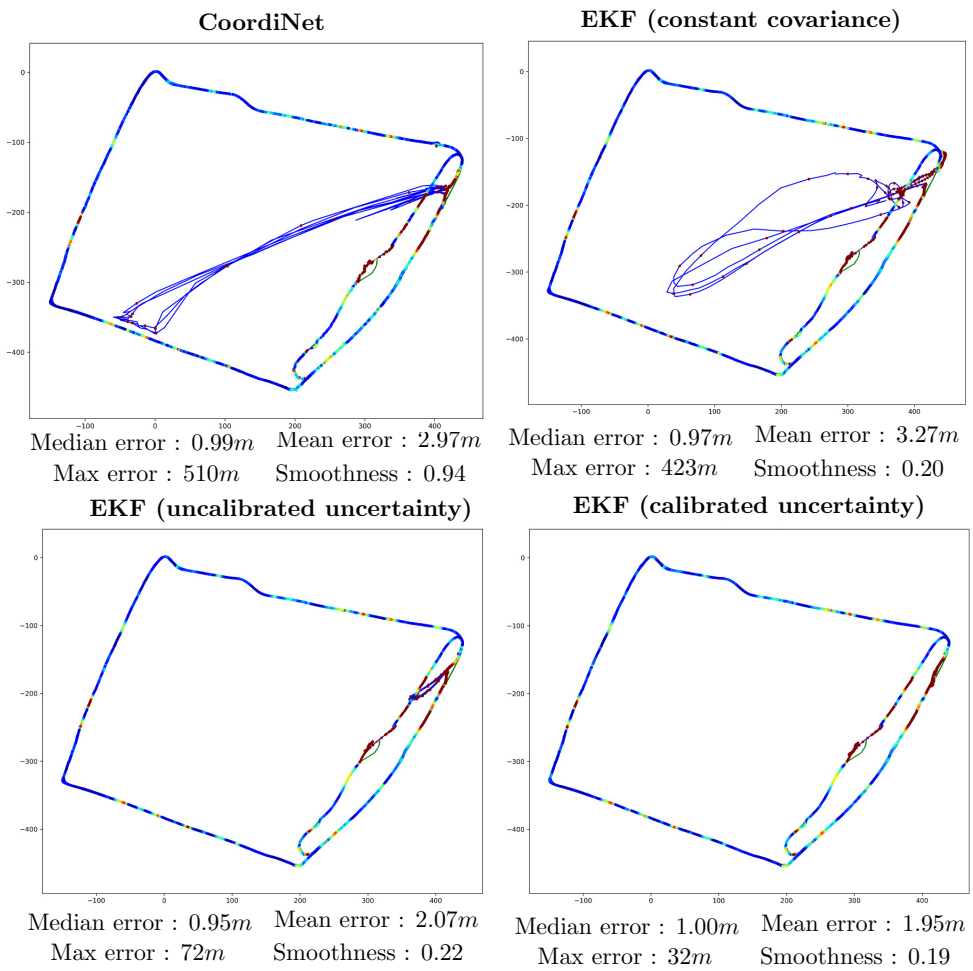


Figure 3.9: **Localization with uncertainty.** Top lane from left to right: CoordiNet predictions, EKF with CoordiNet poses and fixed covariance. Bottom lane from left to right: EKF with CoordiNet raw uncertainty, EKF with CoordiNet calibrated uncertainty. Colormap is the same as in figure 3.7.

We also report the result on Oxford Robotcar experiment in figure 3.10. Again, the EKF is smoothing the trajectory and reduces the mean and maximum error on the overall trajectory. For this experiment, we use uncertainty without additional data. A careful reader should notice that reported results of CoordiNet in this table are slightly different than in table 3.1. This difference comes from the integration

of Coordinet in our ROS framework. These results outperform methods with pose graph optimization reported by MapNet [Brahmbhatt et al., 2018] and RVL [Huang et al., 2019].

We show in figure 3.11 estimated covariance values as well as filtered trajectory in different parts of the Paris dataset map. Notice the shape of the covariance ellipsoid: for instance with the absence of lateral road markings, the covariance grows along the lateral direction.

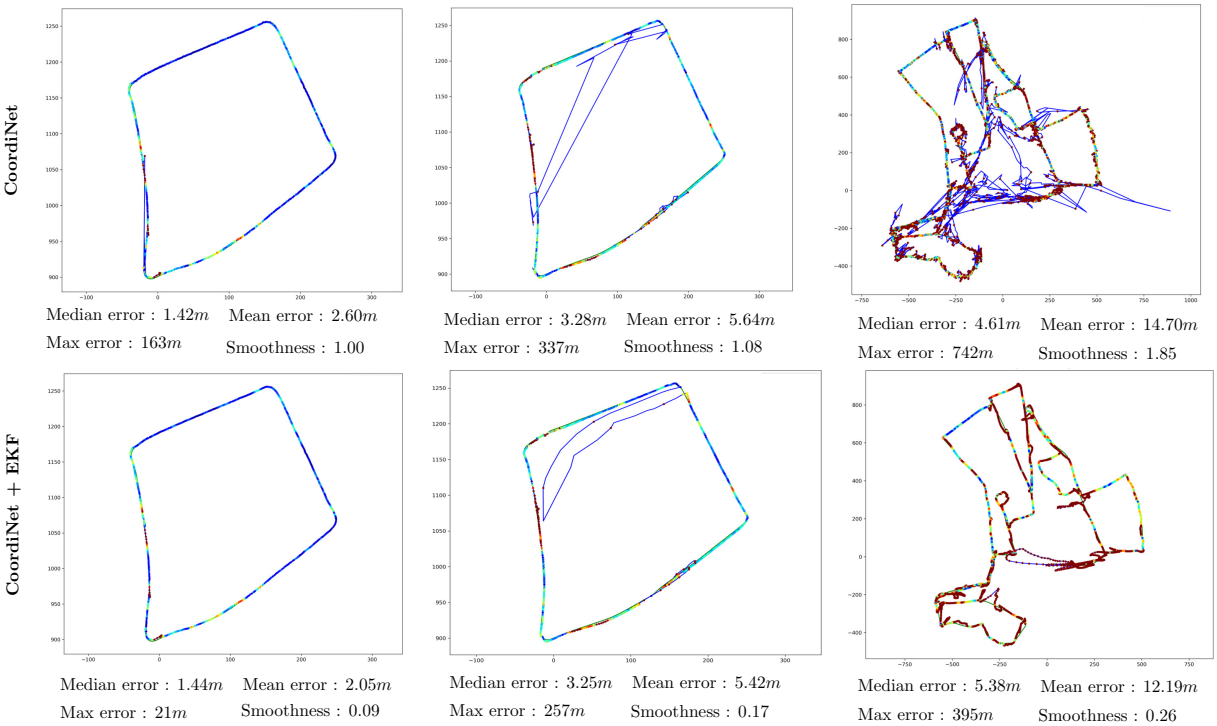


Figure 3.10: **EKF on Oxford experiment.** Top lane: CoordiNet trained with 2 runs. Bottom lane: EKF using CoordiNet poses and uncertainty. Colormap is the same as in figure 3.7.

### 3.6.4 Ablation studies

Finally, we evaluate individual components proposed in this chapter: coord convolution, loss function, pooling, geodesic rotation loss (**geo**) and the use of 3 translation losses instead of 1 (noted **split**). The proposed loss function with learned uncertainty is noted **heterosc.** for heteroscedatic uncertainty, we refer to usual "weighted pose loss" as **homosc**. Shanghai dataset (presented in 3.6.2) is used for this experiment. *EfficientNet b3* is used as image encoder.

Table 3.4: Ablation study on Shanghai dataset (errors in meters/degrees).

| Loss | Coord | CWAP | Split | Rot | Median err. | Mean err |
|---|---|---|---|---|---|---|
| heterosc. | X | X | X | geo. | 0.58 / 0.20 | 1.26 / 0.36 |
| heterosc. | | X | X | geo. | 0.69 / 0.69 | 0.90 / 0.87 |
| homosc. | | X | X | geo. | 0.93 / 0.76 | 1.24 / 1.03 |
| Lt + Lr | | X | X | geo. | 1.23 / 0.71 | 1.68 / 0.94 |
| heterosc. | | | X | geo. | 0.95 / 0.6 | 1.18 / 0.87 |
| heterosc. | | X | | geo. | 0.85 / 0.67 | 1.19 / 0.88 |
| heterosc. | | X | X | L1 | 0.74 / 0.75 | 0.94 / 1.25 |

On this dataset, training with heteroscedatic uncertainty improve results by 16%, coord convolutions by 16%, confidence-weighted average pooling by 27%, geodesic rotation loss by 30% on mean rotation

error and splitting the translation has a positive effect too.

Figure 3.11: **CoordiNet and EKF trajectories:** CoordiNet sequences of poses (red line) are shown with the uncertainty estimate of the current pose (purple ellipsoid). EKF trajectory (blue line) and ground truth (green line) are also displayed. Figure best viewed in color.

### 3.6.5 Localization with crowd-sourced images

This section evaluates the ability of CoordiNet to use crowd-sourced images for both train and test, using the solution proposed in section 3.3.3. More formally, we define the problem as Absolute Pose Regression using a dataset of reference images ($I_k$), each of which has been captured by a camera with known intrinsics parameters ($K_k$). At test time, localization is also performed on images captured by calibrated cameras not observed during training.

We use 2 scenes of the Photo Tourism dataset [Snavely et al., 2006]. It is composed of a collection of internet gathered photographs captured by tourists around famous buildings. We use Brandenburg Gate, illustrated in Figure 3.12, and Trevi Fountain scenes. Because these images present varying resolutions and camera intrinsics, we expect naive Absolute Pose Regression to present poor results because it can not take intrinsics information as input. Conversely, the camera-aware convolutions proposed in section 3.3.3 should be compatible with this heterogeneous dataset.

In this experiment, we compare :

- CoordiNet, presented in section 3.5, with as ResNet34 backbone and no modifications to the architecture described in Figure 3.6.
- A modified version of CoordiNet, where all convolutions (including ResNet layers and CoordConv) are replaced by the camera-aware convolutions.

Because camera-aware convolutions just include 2 additional input channels, they can be integrated in a backbone such as ResNet without breaking the compatibility with the pre-trained weights: a convo-

Figure 3.12: **Samples from the Brandenburg Gate scene in Photo Tourism dataset [Snavely et al., 2006].** Each image has been captured from varying conditions with a different device.

lutional layer with $N$ input and $M$ output channels will have $N+2$ input channels and then introduces only $2 \times M$ new kernels, independent from the $N \times M$ already trained. In this case, we implement them with a separate convolution with only the ray coordinates channels as input and sum the results of both convolution layers, which is mathematically equivalent.

Because all images have different sizes, we crop them to a size of $200 \times 200$, and adapt the intrinsics matrix accordingly. The crops are chosen randomly at each training iteration and we use the top-left corner at test time. The results of this comparison are shown in Table 3.5.

Table 3.5: **Results on Photo Tourism dataset, with and without camera-aware convolutions.**

| | | **CoordiNet** | **CoordiNet + cam conv** |
|---|---|---|---|
| **Brandenburg Gate** | median translation | 6.2cm | 5.0cm |
| | mean translation | 19.6cm | 16.6cm |
| | median rotation | 1.3° | 0.9° |
| **Trevi Fountain** | median translation | 23.3cm | 14.5cm |
| | mean translation | 42.6cm | 29.5cm |
| | median rotation | 1.6° | 0.7° |

We observe that the proposed model with camera-aware convolutions successfully uses the calibration information to localize crowd-sourced images more accurately than the baseline on both scenes.

## 3.7 Conclusions

In this chapter, we proposed multiple contributions about Absolute Pose Regression.

- With geometrical inductive biases, we improved the accuracy of APR networks.

- With heteroscedatic uncertainty quantification, we obtain a reliable uncertainty estimate in an efficient way. We have shown that it can be leveraged in a post-processing filtering method to improve the results.
- With camera-aware convolutions, an APR-based positioning system can operate with crowded-sourced data coming from heterogeneous cameras.
- We conducted much more experiments on APR than the previous literature, especially at large scale. We concluded that the accuracy, still limited compared to structure-based methods, highly depends on the quantity of training data available. We observed that in high-data regime, CoordiNet was able to provide a reliable localization information with a sufficient accuracy ($\approx$ 30cm median error in Paris). CoordiNet can be integrated in real-time vehicle localization systems for an accurate pose estimation in large and busy urban environments.

We also observed some limitations on the proposed model. Notably, despite the proposed architecture modifications, pose regression does not seem able to extrapolate to camera positions unseen during training. We will address this limitation by proposing a solution based on synthetic data in chapter 5. Absolute Pose Regression is also limited by the fact that features extraction and camera pose estimation are entangled in a single neural network, such that an expensive scene-specific training is required before deployment on a new map. In chapter 6, we propose a new direct learning-based approach that separate features extraction and map memorization to mitigate this problem.

Chapter 4

# Literature review: Neural networks as implicit scene representations

## Contents


**4.1 Résumé en Français** . . . . . **44**

**4.2 Implicit Neural Representations** . . . . . **45**

**4.3 Neural rendering with Radiance Fields** . . . . . **46**

4.3.1 Novel view synthesis . . . . . 46

4.3.2 Neural Radiance Fields . . . . . 47

**4.4 Improving Neural Radiance Fields** . . . . . **48**

4.4.1 Accelerate NeRF training and rendering . . . . . 48

4.4.2 Improving further rendering quality . . . . . 50

4.4.3 Neural rendering of dynamic outdoor scenes . . . . . 52

**4.5 Robotics applications of Implicit Neural Representations** . . . . . **53**

**4.6 Idea: Use implicit scene representations as the visual localization map** . . . . . **54**

## 4.1 Résumé en Français

Nous présentons une revue de littérature des représentations neuronales implicites, récemment introduites, qui représentent des données dans l'espace à travers des réseaux de neurones prenant des coordonnées en entrée. Nous examinons plus en détail les Neural Radiance Fields (NeRF) et leurs nombreuses extensions récentes. Nous présentons en particulier les améliorations qui réduisent les temps d'apprentissage et de rendu des NeRF, qui utilisent des formulations de scène plus avancées et traitent les scènes dynamiques. Enfin, nous présentons les applications robotiques des représentations implicites de la scène et introduisons l'idée directrice générale des chapitres suivants : utiliser ces représentations comme carte de localisation visuelle.

The common practice to address a machine learning problem is to consider the available data from one side, and the machine learning model from the other. A machine learning model is a function with learnable parameters, while data is usually represented as a collection of discretized signals (e.g. 2D pixel grids for images, 3D point cloud, voxels, etc.). While these signals are most of the time continuous in the real world, practitioners usually limit the data resolution to keep a tractable computational cost. Recently, the machine learning community has proposed to represent data points as functions stored in neural representations [Dupont et al., 2022], enabling better scaling properties.

This chapter presents Implicit Neural Representations, a novel way to represent signals in machine learning, and review the related literature, in particular neural rendering and robotics applications which have been closely involved in the next contributions of the thesis.

## 4.2 Implicit Neural Representations

Implicit Neural Representations, also referred to as Coordinate-Based Representations, are neural networks that represent data by mapping coordinates to quantities of interest. For example, an image can be parameterized by a function which takes 2D spatial coordinates as input (a location in the image), and outputs the RGB color at the corresponding position. Such intuitive examples are shown in Figure 4.1.

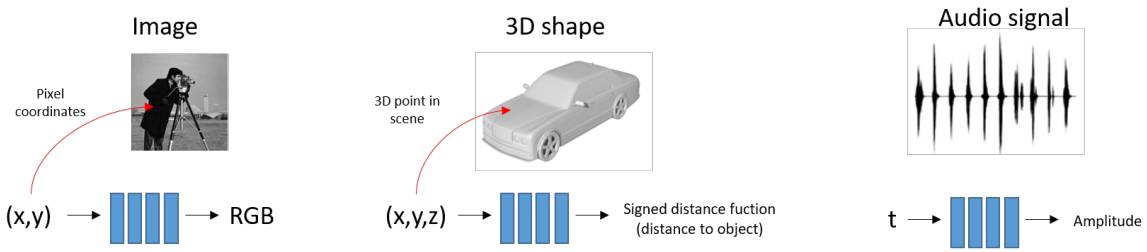


Figure 4.1: **Implicit Neural Representations:** Data such as images, 3D shapes or sound can be represented in neural network which takes coordinates as input and provides quantities of interest.

Successful examples of neural representations of data include images [Bemana et al., 2020, Sitzmann et al., 2020, Chen et al., 2021b], videos [Chen et al., 2022d, Mai and Liu, 2022], sound [Szatkowski et al., 2022], 3D shapes [Park et al., 2019, Atzmon and Lipman, 2020, Zakharov et al., 2022] and 3D scenes [Sitzmann et al., 2019, Jiang et al., 2020, Mildenhall et al., 2020].

These models are usually implemented with Multi Layer Perceptrons (MLP). SIREN [Sitzmann et al., 2020] proposes to use periodic activation functions in the architecture and shows its superiority on several types of data representation. CoordX [Liang et al., 2022] processes each input component in separate heads in order to make it more efficient. Vector Neurons [Deng et al., 2021] enables the learning of a rotation-equivariant shape space. MLPs have been observed to struggle to learn high frequency details of the signal when the input coordinates are low-dimensional [Tancik et al., 2020]. The proposed solution uses positional encodings with Fourier features to project input coordinates to a higher dimensional space, resulting in implicit representation with sharper details.

Implicit representations have several advantages over explicit data representations. First, their are not coupled to a given spatial resolution because they are continuous functions. As a result, their memory

footprint does not depend on resolution but only on the complexity of the underlying continuous signal. The information is embedded in neural networks weights which enable to store complex information compactly, usually with few megabytes. Finally, this continuous representation can be evaluated at any coordinate, enabling a potentially "infinite" resolution.

There are relevant applications of such representations in computer vision and graphics tasks such as data compression [Dupont et al., 2021, Strümpler et al., 2022], image and video super resolution [Chen et al., 2021b, Chen et al., 2022d], inverse graphics (learning 3D from 2D signals) [Niemeyer et al., 2020, Lin et al., 2020] and novel view synthesis [Mildenhall et al., 2020] described in section 4.3 or even robotics applications described in section 4.5.

## 4.3 Neural rendering with Radiance Fields

### 4.3.1 Novel view synthesis

Novel View Synthesis (NVS) consists in rendering an image of a scene/object from an arbitrary viewpoint, given a sparse set of observed views with known cameras poses. This task is illustrated in figure 4.2.

Figure 4.2: **Novel view synthesis.** Given a sparse set of observations of a scene (left) and their corresponding camera poses (middle), novel view synthesis algorithms are asked to render images of the same scene from unobserved camera poses (right). Figure from [Mildenhall et al., 2020].

Solving the NVS task requires to hallucinate the 2D projection of the scene in a new viewpoint, and thus require algorithms to have an accurate representation of the 3D scene geometry. However, only 2D images of the scene are available, such that NVS involves an "inverse graphics" problem, i.e. recovering the 3D scene geometry using epipolar geometry. Several classical approaches could be used to solve this problem. It relied either on light fields interpolation techniques [Gortler et al., 1996, Levoy and Hanrahan, 1996, Davis et al., 2012], or based on a textured mesh [Debevec et al., 1996, Wood et al., 2000, Waechter et al., 2014]. Multi-View Stereo [Furukawa et al., 2015, Yao et al., 2018, Yao et al., 2019] can be leveraged in order to obtain dense depth maps on reference images before re-projecting pixels on the novel view.

We want to highlight that novel view synthesis use the same datasets than visual localization (images depicting a given scene from different viewpoints, with corresponding camera poses) and solves the opposite problem: rendering the image from the camera pose instead of inferring the pose from the image, in a scene-specific model.

### 4.3.2 Neural Radiance Fields

Neural Radiance Fields [Mildenhall et al., 2020] (NeRF) address the novel view synthesis task with a 5D implicit neural representation optimized by minimizing the per-pixel rendering error on reference images. A scene is represented by a continuous function, whose inputs are a point in space $(x, y, z)$ and a viewing direction $(\theta, \phi)$ and outputs are a volume density $\sigma$ and the emitted color $c = (r, g, b)$ at the point observed from the given direction. This function is modelled by an 8 hidden layers MLP with ReLU activations and Fourier positional encodings [Tancik et al., 2020].

The rendering of a pixel is performed by the following operations: first, points are sampled along the corresponding camera ray (assuming a camera pinhole model with known intrinsics). Then, sampled points with the ray direction are processed by the NeRF implicit representation. Then, the obtained densities and colors along the ray are aggregated through volumetric rendering resulting in the final pixel color. This process is described in figure 4.3.

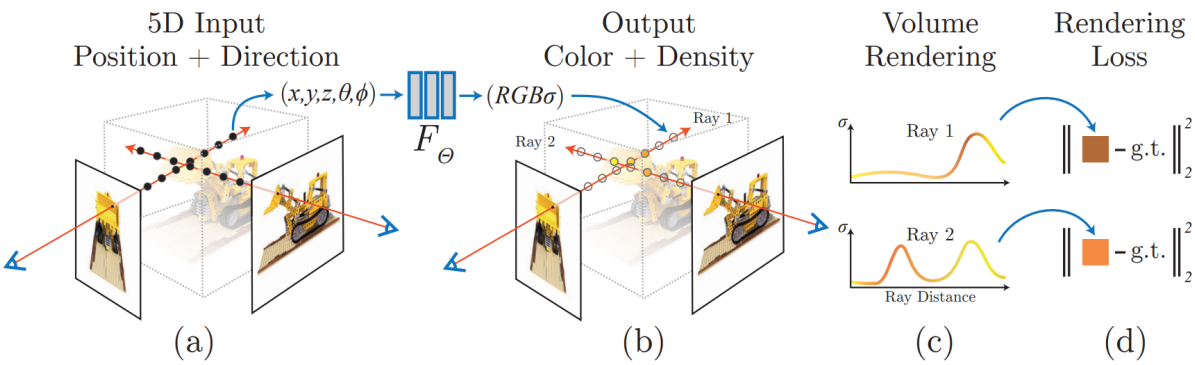


Figure 4.3: **Neural Radiance Fields:** NeRF are an implicit scene representation which takes a 3D point and a viewing direction as input, and returns the volume density at this point and a RGB color. Rendering a pixel is performed by ray marching in the 3D space, evaluation of NeRF on a discrete set of sampled points, and differentiable aggregation along the ray by volume rendering. Figure from [Mildenhall et al., 2020].

One crucial component of NeRF algorithm is the volumetric rendering formula, which aggregates densities $\sigma_i$, colors $c_i$ and distance to camera $t_i$ of points along a ray to obtain the pixel color $C$ :

$$C(r) = \sum_{i}^{n} T_i(1 - \exp(-\sigma_i \delta_i))\mathbf{c_i} \quad \text{where} \quad T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j) \quad \text{and} \quad \delta_i = t_{i+1} - t_i \tag{4.1}$$

It computes a simple volume rendering process by leveraging a probabilistic notion of visibility. Its main advantage is to be differentiable, such that gradients from the loss function computed on 2D pixels colors can be back-propagated to the densities and colors of each 3D point. More details are provided in [Max and Chen, 2005, Tagliasacchi and Mildenhall, 2022].

The viewing direction is used as input to take into account the specularities of non Lambertian objets. However, the volume density (which can be interpreted as occupancy) of a point does not depend on the viewing direction. This is achieved by feeding the viewing direction in the neural network at an intermediate layer where density has already been predicted (see Figure 4.8).

Points are sampled along the ray in a coarse-to-fine approach: first "coarse" points are uniformly

distributed from the camera center to a (arbitrary) maximum distance. The resulting density values provided at these points are then interpreted as a discrete probability distribution function and "fine" points are sampled from this distribution and expected to lie on the actual surface. The original NeRF model trains two separate model entities for "coarse" and "fine" resolutions.

With this formulation, NeRF presents state of the art results for novel view synthesis with photorealistic synthetic views. However, many limitations remain to be addressed:

- We make the assumption that the scene is static: lightning conditions and objects locations should remain constant between the different captured views. This is true in indoor environments and controlled settings. However, in the case of outdoor scenes studied in this thesis, this assumption does not hold: datasets are collected at different seasons and times of the day and many dynamic objects such as vehicles and pedestrians are observable. Scalability to large scenes is also highly limited.

- Training and rendering processes are very slow and compute intensive: the rendering of each pixel is computed independently and requires the evaluation of hundreds of points through the MLP. As a result, training a NeRF on a single scene can take days to obtain optimal results, and several seconds are required to render a single image on a powerful GPU.

- The rendering quality remains limited. First, finest high frequency details might not be captured accurately by the neural network which tends to smooth the signal. Then, when the number of observed views is limited, Neural Radiance Fields struggle to capture the correct 3D of the scene, resulting in poor quality of extrapolated novel views.

## 4.4 Improving Neural Radiance Fields

Since the release of Neural Radiance Fields in 2020, extensive research has been conducted to address the aforementioned limitations. This section presents these Neural Rendering models which focus on faster algorithms 4.4.1, rendering of dynamic outdoor scenes 4.4.3, and higher rendering quality models 4.4.2. An extensive overview of existing Neural Fields models is provided in [Xie et al., 2021b].

### 4.4.1 Accelerate NeRF training and rendering

The major bottleneck of Neural Radiance Fields for many downstream applications is the time of the rendering process. This is because $H \times W \times N$ MLP evaluations are required to render an image of resolution $H \times W$ with N points sampled per ray. Many solutions, presented below, have been pursued to enable real-time rendering with NeRF.

#### 4.4.1.1 Use smaller networks

The simplest idea is to reduce the size of the neural network (originally 8 hidden layers with 256 neurons). However tiny networks do not have enough capacity to memorize the details of an entire scene and then degrade image quality significantly.

KiloNeRF [Reiser et al., 2021] proposes a divide-and-conquer approach by training thousands of tiny MLPs each of which represents a small portion of the scene, enabling a 3 orders of magnitude speed-up

when combined with parallel computation. Training time is not improved because knowledge distillation between a regular NeRF and the tiny MLPs need to be performed.

A successful idea is to use hybrids explicit-implicit formulations (see figure 4.4). Because tiny networks can not model the radiance field depending on the spatial coordinates $(x, y, z)$ accurately, the solutions consists in injecting intermediate features as network input which contain more information than position. These learned features are spatially distributed in a discretized representation of the scene and continuity is ensured by linear interpolation. It can be done in a voxel grid [Liu et al., 2020b, Garbin et al., 2021, Hedman et al., 2021, Sun et al., 2022] or in octrees [Yu et al., 2021]. Notably, Instant-NGP [Müller et al., 2022] provides very fast training and inference combined with high quality rendering by interpolating multi-resolution voxel grids, implemented in hash tables. Plenoxels [Fridovich-Keil and Yu et al., 2022] and TensoRF [Chen et al., 2022a] even learn an explicit radiance field without neural networks.

Most of these solutions achieve faster rendering by trading compute against higher storage requirements. However, pruning methods can be used to mitigate the memory footprint of explicit voxel grids [Liu et al., 2020b, Deng and Tartaglione, 2023].

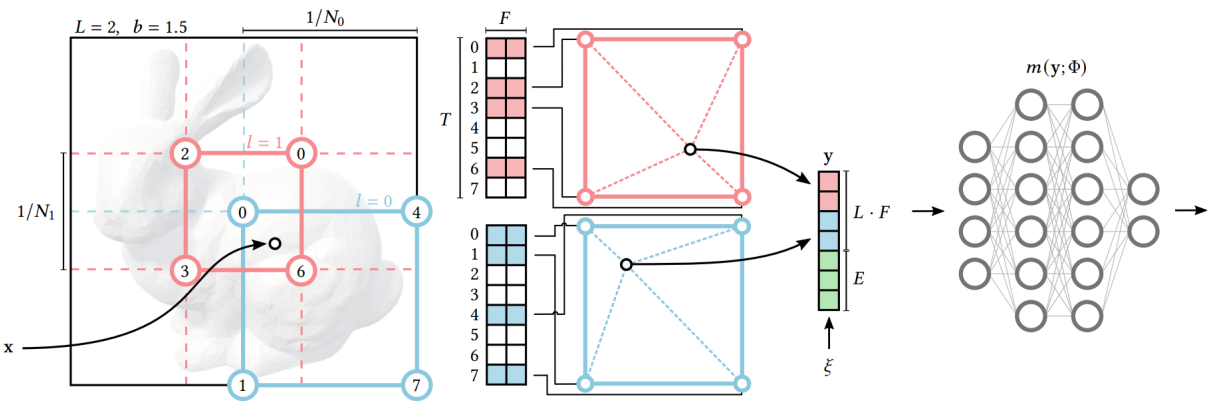


Figure 4.4: **Instant-NGP architecture.** Intermediate features at a given 3D point are extracted via trilinear interpolation of multi-resolution hash tables of features. Density and color are then decoded in a tiny MLP, enabling fast rendering. Figure from [Müller et al., 2022].

#### 4.4.1.2 Light Field formulations

An other research direction is to replace the ray marching and volumetric rendering approach (which involve hundreds MLP evaluations per pixel) by an implicit formulation which computes the color of a ray in a single evaluation. This camera ray representation is usually referred as light field or Lumigraph [Gortler et al., 1996]. The light field can be seen as a radiance field integrated along the ray. Several concurrent work have been developed in this direction:

One important design choice is the way to parameterize input rays. Neural Light Fields [Attal et al., 2022] and NeuLF [Li et al., 2022b] use a 4D formulation that considers the intersections of the ray with two planes. While simple, it is only applicable to forward-facing scenes. Light Field Networks [Sitzmann et al., 2021] use 6D Plücker coordinates enabling to model 360 degrees light fields. R2L [Wang et al.,

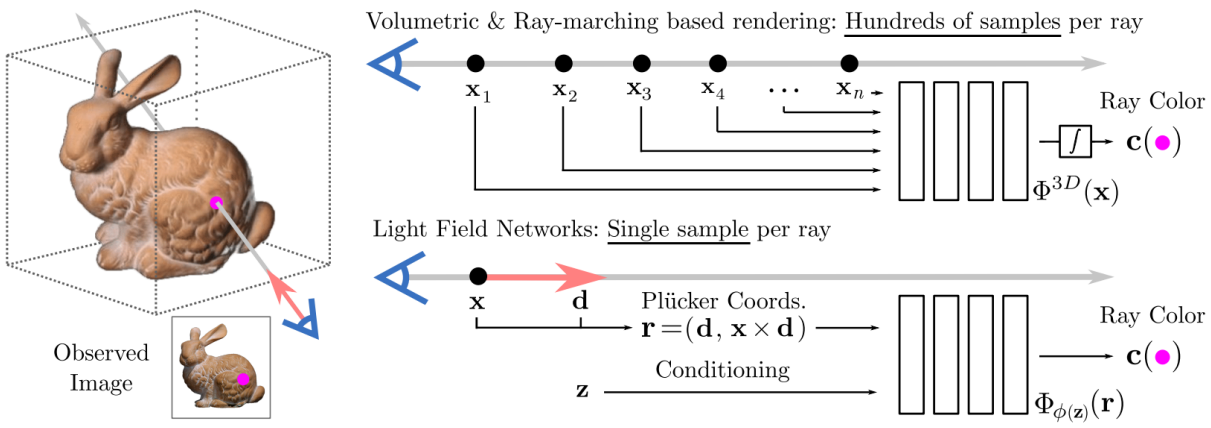


Figure 4.5: **Light Fields Network.** Ray marching and volume rendering are replaced by an implicit representation of the light fielf. Inputs are camera rays, parametrized by Plücker coordinates. Figure from [Sitzmann et al., 2021]

2022] concatenates the 3D coordinates of (randomly) sampled points along the ray, which the authors observe to perform better than Plücker coordinates. Neural Point Light Fields [Ost et al., 2022] uses only the ray direction but leverages cross attention on features learned from a sparse point cloud of the scene.

While radiance fields are relatively smooth functions which can be easily approximated by a neural network, this is not the case for light fields where rays with similar coordinates can have very different colors due to occlusions. As a result, implicit light fields are difficult to learn. They struggle to match the accuracy of NeRF in complex real world scenes [Sitzmann et al., 2021], or need to be supervised by an existing NeRF model [Wang et al., 2022].

### 4.4.2 Improving further rendering quality

While Neural Radiance Fields exhibits state-of-the-art image quality in the novel view synthesis task, they exhibit artefacts when the scene geometry is not learned correctly, when the objects are observed closer than during training, or when dealing with reflectances. This section presents models that mitigate these issues.

#### 4.4.2.1 Learning better 3D geometry

NeRf has the capability to recover the 3D geometry of the scene in the form of a volume density field (from which depth maps can be rendered), only from 2D RGB images supervision (and the camera pose from which the image has been captured). In this "inverse graphics" problem, each reference image used for training can be seen as an additional constraint which brings more information to solve the problem. As a result, when few training views are available, the problem is under-constrained in the sense that multiple 3D geometries can explain the 2D observations. In this scenario, described by [Zhang et al., 2020a] as shape-radiance ambiguity, the model is not guaranteed to converge to the actual geometry, resulting in wrong extrapolated views.

A first solution to this problem, proposed by DS-NeRF [Deng et al., 2022] is to provide direct depth supervision on the training views. It is feasible because the computation of the depth value is

differentiable w.r.t. the volume field. In addition to that, camera poses have to be computed before the training, which is classically done by performing a 3D reconstruction on the scene with structure-from-motion [Schönberger and Frahm, 2016]. This 3D reconstruction provides sparse correspondences between pixels and 3D points which can be used as depth labels during the NeRF training. Similarly, Urban Radiance Fields [Rematas et al., 2022] supervises the density field with lidar signal.

Otherwise, regularization losses have been proposed on the rendered depth maps. RegNerf [Niemeyer et al., 2022] minimizes the total variation on local patches, assuming that most objects are locally smooth. It avoids degenerated geometries and enhance the geometry of flat surface such as roads and walls, but degrades objects with more complex shapes such as trees. DiffNeRF [Ehret et al., 2022] proposes a more advanced regularization objective based on differential geometry. However, the regularization is done on the gradients of the depth values and thus requires 2 backward passes per training iteration.

#### 4.4.2.2 Using more advanced formulations

NeRF is a simplistic model of the image formation process from the 3D space to the camera center, which has the advantage of being differentiable from the radiance field to the resulting RGB image and computationally tractable. However, some of these simplifications involve poor image qualities in scenarios such as zooms or reflections.

**4.4.2.2.1 Anti-aliasing:** By representing the light captured by a camera pixel by a 1-dimensional ray, the model ignores the fact that an object very far from the camera projects a wider area in a given pixel than an object very close. In other words, the radiance emitted by a given point depends on the distance from the camera.

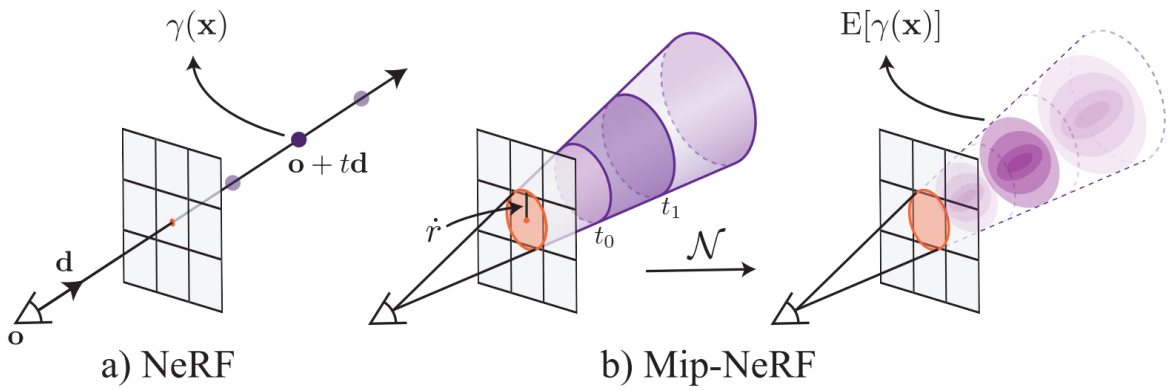


Figure 4.6: **MipNeRF.** NeRF samples points along the camera ray. In reality a pixel captures the light emitted from a conical frustum. Mip-NeRF uses integrated position encodings to capture the variations in resolution and remove aliasing effects. Figure from [Barron et al., 2021].

With the point-wise representation of NeRF model, an object which has always been observed from the same distance during training and synthesized from a closer view will not be rendered accurately because of an aliasing effect. This problem was not visible in the NeRF-synthetic dataset, widely used to evaluate NVS methods, because train and test views are distributed on a sphere around the object, but appears more frequently on real world datasets.

MipNeRF [Barron et al., 2021] proposes a new ray marching formulation for NeRF to address this problem. The MLP does not take coordinates of a single point as input but rather conical frustums (see Figure 4.6). Such frustums are represented by integrated positional encodings (IPE): the 3D volume of conical frustums is approximated by multivariate Gaussian and the closed-form integration of the Gaussian coordinates are used as input to the MLP. This way, the model takes into account the aforementioned aliasing problem when rendering each sample along the ray. MipNeRF performs consistently better than the original NeRF formulation in terms of image quality on real world benchmarks without extra computational cost.

### 4.4.3 Neural rendering of dynamic outdoor scenes

The impressive results presented in the original NeRF publication [Mildenhall et al., 2020] are performed on an object-centric synthetic dataset (referred as NeRF synthetic) and a real-world dataset with indoor forward-facing scenes (LFFF [Mildenhall et al., 2019]). However, many real world applications of such novel view synthesis models would require to be able to operate in dynamic environments, such as outdoor scenes where the observed background can be very far from the camera, some objects move over time, and the illumination is not the same between all observed views.

Such "in the wild" scenarios are modelled by NeRF-W [Martin-Brualla et al., 2021]. This framework learns to separate the static background of a scene from the transient occluders without supervision, as well as the specific lightning appearance of each training image. It is done by incorporating separate MLP heads for rendering static and transient scenes and 2 latent codes $\mathcal{L}_i^{(a)}$ (appearance embedding) and $\mathcal{L}_i^{(\tau)}$ (transient embedding) that provide control on the appearance and dynamic content of each rendered view. As a result, an accurate radiance field of the static scene can be learned and rendered with controllable lightning appearance (see figure 4.7). While transient objects are effectively removed from the rest of the scene, they can not be rendered accurately.

Rendering moving scenes can be tackled by making the density of each point dependent on the time [Li et al., 2021, Li et al., 2022a] or by modelling deformations [Pumarola et al., 2020, Tretschk et al., 2021, Park et al., 2021]. Another option is to construct a graph with independent instances of implicit representations for each object [Ost et al., 2021, Niemeyer and Geiger, 2021, Yang et al., 2021, Kundu et al., 2022]. It enables to animate the scene with control on the movement, instead of just "replaying" a video from different viewpoints.

Outdoor scenarios involve rendering background pixels which depict landmarks far away from the camera (or even the sky at an infinite distance), which is not compatible with the ray marching approach of NeRF, which sample points inside of a pre-defined bounded area. Nerf++ [Zhang et al., 2020a] proposes to separate foreground and background by a unit sphere. The foreground is rendered with a classical NeRF, while the background is rendered by an additional neural field model with an inverted sphere parametrization designed for unbounded scenes. Urban Radiance Fields [Rematas et al., 2022] leverages lidar sweeps to learn the geometry of outdoor scenes and models the sky with a separate model.

(a) Photos (b) Renderings

Figure 4.7: **Rendering of dynamic scenes.** Nerf-W is able to render a scene with controllable appearance from various tourists photographs. More examples shown in Figure 3.12.

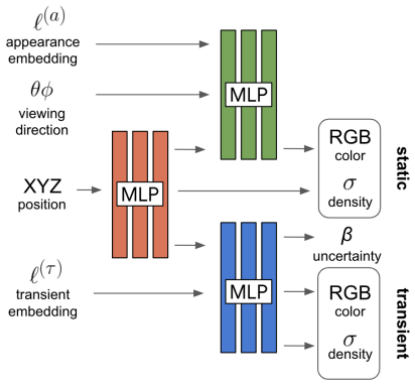


Figure 4.8: **NeRF-W.** The model renders the static scene in one branch and images with transient objects in the other. From [Martin-Brualla et al., 2021]

The scalability to very large scenes, i.e. city-scale, is addressed by Block-NeRF [Tancik et al., 2022]. The wide area is divided into smaller blocks and a NeRF instance is trained for each specific block, while maintaining appearance consistency between blocks. Arbitrarily large maps can be rendered by this technique, which is promising for autonomous driving applications, but requires extensive computational resources.

## 4.5 Robotics applications of Implicit Neural Representations

The main application of Neural Radiance Fields is novel view synthesis which is more related to computer graphics than the computer vision problems presented in this thesis. However, by their ability to represent a scene, Neural Fields and related implicit representations are recently being adopted by the robotic community for real-world downstream tasks. This section give an overview of the existing robotics applications.

First, some applications have been developed in robot grasping [Breyer et al., 2021, Ichnowski et al., 2020, Kerr et al., 2022, Zhou et al., 2023], for which NeRF is particularly useful to represent transparent objects. It is also used for trajectory planning [Adamkiewicz et al., 2022], object pose estimation [Zakharov et al., 2020, Irshad et al., 2022, Huang et al., 2022, Goli et al., 2022], scene reconstruction [Zakharov et al., 2021, Cao and de Charette, 2022, Ortiz et al., 2022], sensors extrinsics calibration [Herau et al., 2023] or training data generation [Wen et al., 2022] and auto-labelling [Zhi et al., 2023].

Closer to visual localization, RGB-D SLAM algorithms can leverage implicit scene representations to represent the map of the environment. iMAP [Sucar et al., 2021] builds a dense map represented by a MLP trained during the exploration of the environment. However, the scene geometry provided by the MLP lacks sharpness, and the method does not scale to large environments. NICE-SLAM [Zhu et al., 2022] mitigates these problems by introducing a hierarchical scene representation with geometrical priors. These works are inspiring in the context of visual localization. However, their implicit map representation is currently limited to indoor static environments (shown in Figure 4.9), rely on RGB-D sensors to build the map geometry and are currently less performing than structure-based methods. Camera relocalization methods inside of these neural maps with RGB-D camera is a promising direction [Bruns

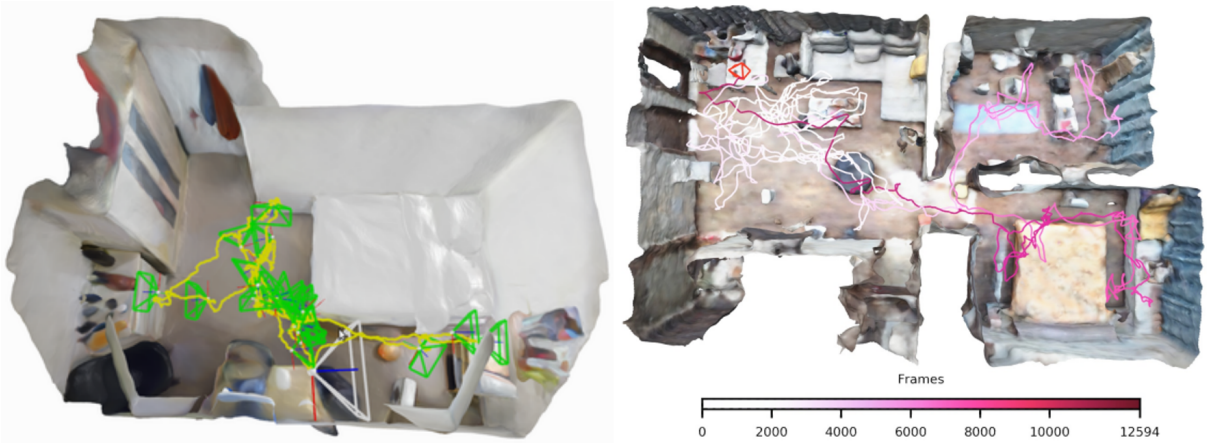


Figure 4.9: **Implicit scene reconstructions with RGB-D SLAM:** iMAP [Sucar et al., 2021] (left) and NICE-SLAM [Zhu et al., 2022] (right)

et al., 2022].

## 4.6 Idea: Use implicit scene representations as the visual localization map

As we discussed in chapter 2, most of state-of-the-art algorithms for visual localization do not use end-to-end neural networks but rather classical pipelines where features extracted by learning-based methods are matched against reference data. For example, Visual Place Recognition (see section 2.2) compares a global image descriptor to an image retrieval database and Structure-Based methods (see section 2.2) match keypoints against a 3D model.

Regardless of how this reference information is represented, it can be interpreted as the map the environment (i.e. a compressed representation of what we observed before with spatial information), and we notice that explicit representations are used. We wonder if this map information could be represented implicitly, and hypothesize potential benefits:

- In current map representations (retrieval database and 3D models), collecting more data result in a higher resolution at the cost of increased storage requirements and matching time. The storage requirement of implicit scenes representations does not depend on the quantity of collected data, but on the complexity of the underlying signal. As a result, in a high-data regime, we could learn very accurate dense map representations with compact storage requirements.
- NeRF and related methods have shown great potential in recovering the 3D information from 2D signals. Because visual localization use 2D images but needs to reason in the 3D world, using a NeRF-like approach would enable to build 3D-aware maps only from 2D data in an accurate way.

The next contributions of this thesis have explored this idea for different localization approaches.

- First, in chapter 5 we propose a simplistic approach to transfer the knowledge from a NeRF to an Absolute Pose Regressor: generate synthetic datasets of photorealistic novel views and use it to

train a localization network. This procedure shows great benefits because we observe real-world datasets to be limited and biased.

- In chapter 6, we replace the image retrieval database by an implicit map representation. With this formulation, we are not limited to one global descriptor per reference image but can compute the descriptor of any camera viewpoint in the map. We show how to train it, perform inference, and how it compares to pose regression and image retrieval competitors in kilometers-scale maps.
- Chapter 7 proposes to replace the sparse SfM 3D model commonly used in structure-based methods by a Neural Field that learn, store and render local features. Again, it enables to render images of local features from any viewpoint in the scene, and then to solve the localization problem by iterative refinement against the dense map representation.
- Finally, a visual localization algorithm for vehicle localization which combines these contributions in a hierarchical formulation is proposed, but not evaluated by lack of time, in chapter 8.

Chapter 5

# Visual Localization augmented by NeRF synthesis

## Contents


**5.1 Résumé en Français** . . . . . **58**

**5.2 Motivation** . . . . . **59**

**5.3 Related work** . . . . . **60**

**5.4 Synthetic dataset rendering with LENS** . . . . . **61**

5.4.1 NeRF-W training . . . . . 61

5.4.2 Density volume generation . . . . . 62

5.4.3 Virtual camera locations . . . . . 62

5.4.4 Novel view synthesis . . . . . 63

5.4.5 Camera pose regressor training . . . . . 63

**5.5 Experiments** . . . . . **63**

5.5.1 Datasets and implementation details . . . . . 64

5.5.2 Comparison with related localization methods . . . . . 65

5.5.3 Ablation studies . . . . . 65

5.5.4 Exploring camera pose regression on synthetic domain . . . . . 67

5.5.5 Limitations . . . . . 68

**5.6 Conclusion** . . . . . **68**

## 5.1 Résumé en Français

Nous présentons une technique pour améliorer les méthodes de Pose Regression avec des images synthétiques générées par NeRF. À partir d'un petit ensemble d'images de référence d'une scène donnée, nous entrainons un NeRF et l'utilisons pour synthétiser des images photoréalistes uniformément réparties dans la scène.

Ce grand jeu de données synthétiques est combiné aux images de réference pour entrainer un modèle de regression avec une précision améliorée. En utilisant notre méthode LENS, nous observons une amélioration importante de la précision sur des benchmarks publics pour lesquels les méthodes de Pose Regression sont généralement peu performantes. Les différentes expériences conduites montrent que l'amélioration ne provient pas seulement d'une plus grande quantité d'images, mais plutôt d'une meilleure distribution de l'ensemble de données d'entraînement.

## 5.2 Motivation

In chapter 3 we presented a vehicle localization approach based on Absolute Pose Regression. During our experiments, we noticed failure cases in some specific situations, such as the overtaking of a vehicle stopped on the road (see Figure 5.1). After investigation, it appeared that all the reference data in this street was captured on the right lane, and that the neural network was not able to extrapolate to a position outside of the training data range.

Figure 5.1: **Localization failure during an overtaking.** Absolute Pose Regression fails to recover the vehicle position in the left lane because this situation requires to extrapolate outside of the training dataset positions.

Our hypothesis is that camera pose regression formulated as a machine learning problem is currently limited because training datasets are highly biased and lead to a poor generalization. In contrast with many tasks solved with deep learning, camera pose regressors are overfitted to a single scene and we expect their output space to be the set of possible camera poses in this scene. But in practice, training datasets for camera pose regression are often built from a limited number of consecutive video frames. Consequently, they lack diversity compared to the set of well distributed camera positions and orientations in a given scene (see figure 5.2). [Sattler et al., 2019] have shown that these networks are not able to extrapolate to unseen camera positions, and then struggle to perform better than an image retrieval baseline. We suppose that a training dataset which is well distributed on the entire scene should help to overcome this limitation.

Our proposal is to augment training datasets for pose regression using Neural Radiance Fields (NeRF) [Mildenhall et al., 2020]. Compared to standard generative models [Zhang et al., 2020b] used for view synthesis, NeRF rendering is tailored for our localization problem as it produces geometry-consistent images for any pose query in the scene thanks to the ray tracing approach. To optimize the pose regressor localization performances and mitigate the cost of images rendering, we propose an algorithm, named LENS, which generates virtual camera locations distributed on a regular grid across the scene. In order to avoid usage of degenerate views due to occlusions or wrong orientations, LENS leverages the internal scene geometry learned by the NeRF model. It allows to discard poses close to occluders like buildings, and provides only meaningful images to the pose regressor without any scene-specific parameter tuning. We apply our method on both indoor and outdoor datasets to show the benefit of a spatially balanced dataset for training a pose regressor.

In the next section, we review previous work related to novel view synthesis for visual localization. In

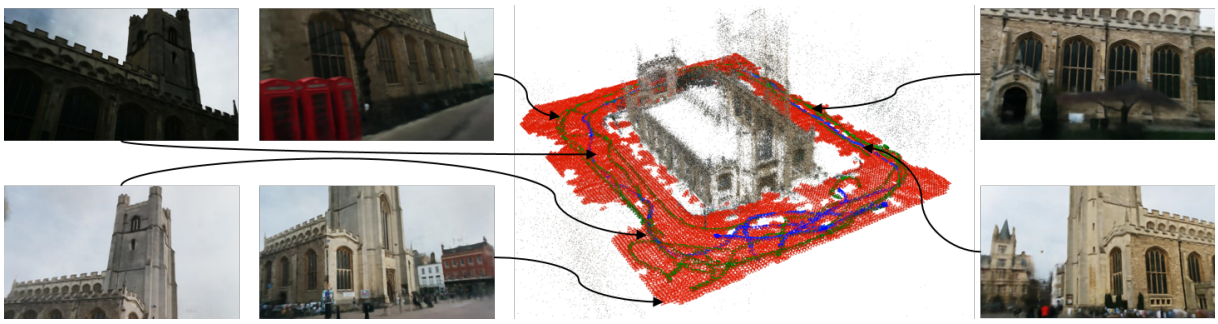

Figure 5.2: **Novel view synthesis with LENS:** given a 3D scene defined by a set of images labeled with corresponding poses (green cameras), we train a NeRF and render novel views (depicted images) on pose queries distributed across the scene (red cameras). Synthetic and real images are gathered to train a camera pose regression model, which performs twice better when evaluated on test set images (blue cameras) compared to the same model trained only on real training samples.

section 5.4 we describe our method, specifically the choice of NeRF and the synthetic poses generation algorithm. Section 5.5 is dedicated to experiments conducted on Cambridge Landmarks and 7scenes datasets. Section 5.6 concludes the chapter.

## 5.3 Related work

Novel view synthesis can be used at several steps of a visual localization method. [Zhang et al., 2020b] refine reference poses iteratively by comparing rendered view of the pose estimate with the original image. This idea can also be exploited in the relocalization step of a structure-based method: InLoc [Taira et al., 2018] verifies the predicted pose by comparing local patch descriptors between original and rendered image. iNeRF [Yen-Chen et al., 2021] performs gradient-based optimization to recover a pose estimate thanks to NeRF differentiability. Direct-PoseNet [Chen et al., 2021a] adapts this idea to camera pose regression training by using an additional photometric loss between query image and NeRF synthesis on the predicted pose. PoseGan [Liu et al., 2020a] learns jointly pose regression and view synthesis resulting in an improved localization accuracy.

Another direction is to use synthetic images to enlarge the reference database with more densely sampled views. Localization algorithms perform better when an image with a similar viewpoint is available for matching (structure based methods) or training (learning based). This idea can also be applied to other vision tasks, such as 3D semantic segmentation [Kundu et al., 2020]. These methods yield two important design choices: how to chose where virtual cameras are located, and how to perform novel view synthesis.

Virtual camera locations are usually sampled using ad-hoc methods: [Torii et al., 2015b] expand a place recognition database by sampling on a regular grid, removing locations that lie in buildings thanks to coarse depth plans representing basic 3D structures or locations that are too far from the original trajectory. [Aubry et al., 2014] and [Irschara et al., 2009] use a similar approach and discard cameras that doesn't view any portion of the 3d model. Some alternatives propose to use a probabilistic sampling strategy based on detectability of keypoints at a specific pose [Purkait et al., 2018], or try to find an optimal set of locations with a genetic algorithm [Chen and Li, 2004]. However, ad-hoc methods have

been observed to be sufficient in practice [Irschara et al., 2009], so our method uses a regular grid combined with NeRF internal representation of the scene to ensure meaningful views.

One popular approach to render novel view is to use a 3D textured mesh to represent the scene and to create virtual cameras within this scene [Aubry et al., 2014, Sattler et al., 2019]. However, obtaining such scene representation is costly and not easy to compute from crowd-sourced images acquired in a dynamic environment. Meshs built from crowd sourced images usually fail to describe low textured or crowded part of the scene, such as sky and ground. Synthetic images can also be rendered thanks to Generative Adversarial Networks [Purkait et al., 2018, Toker et al., 2021]. Finally, recent methods that learn a continuous volumetric representation of the scene such as Neural Radiance Fields [Mildenhall et al., 2020] outperform prior work and exhibit photorealistic results. To the best of our knowledge, no study has been conducted to enlarge camera pose regression training dataset with NeRF rendered views.

## 5.4 Synthetic dataset rendering with LENS

We provide here a description of our method, named LENS, which can be seen as an offline data augmentation pipeline to train pose regressors. Our goal is to generate a large and uniformly distributed dataset of synthetic images, using a small set of images depicting the scene labeled with reference camera poses (usually provided by structure from motion methods such as COLMAP [Schönberger and Frahm, 2016]). This dataset is then used to train a camera pose regressor, resulting in an improved accuracy without additional computation during localization inference. Our framework is described in figure 5.3 and is detailed below.

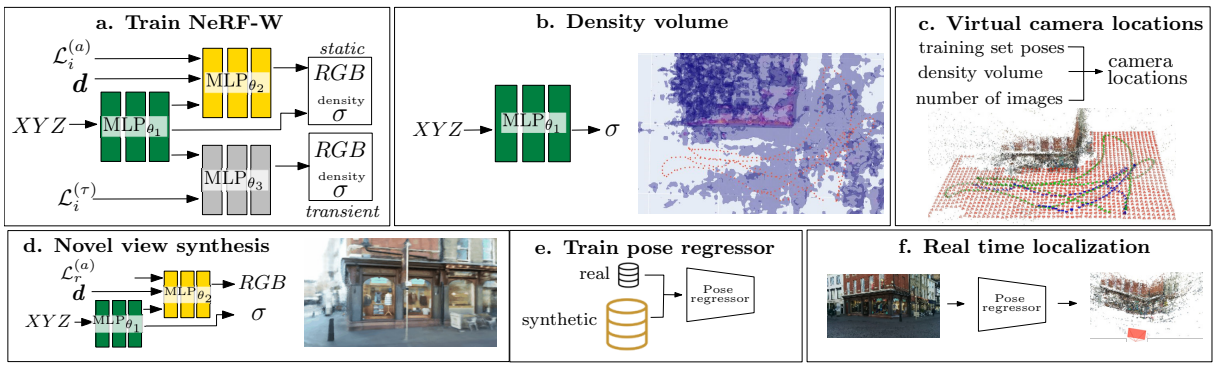


Figure 5.3: **LENS pipeline:** First, NeRF-W is trained with available real images (a.). Then, the trained model is used to detect high density points in the scene (b.). Poses from real images and high density locations are used to generate virtual camera locations (c.) on which novel view synthesis is performed (d.). Combined real and synthetic datasets are used to train a pose regressor (e.).

### 5.4.1 NeRF-W training

A neural radiance field learns a continuous 3D representation of a scene from a collections of images depicting this scene from known camera viewpoints. It uses 2 feed-forward neural networks: $\text{MLP}_{\theta_1}$ connects a 3D position to a density value $\sigma$ and $\text{MLP}_{\theta_2}$ predicts a RGB color from a viewing direction $\boldsymbol{d}$ and a latent vector $\text{MLP}_{\theta_1}(x, y, z)$. The final RGB value of a pixel is computed by approximating the

volume rendering integral using $N_c$ coarse samples and $N_f$ fine samples along a light ray with predicted colors $(\mathbf{c_i})_\mathbf{n}$ and densities $(\sigma_i)_n$.

With this formulation, NeRF models are able to render a static scene captured under controlled settings but fail in real world dynamic scenes with variable illuminations and dynamic objects. NeRF in the Wild [Martin-Brualla et al., 2021] (NeRF-W), overcomes this limitation by modelling these temporal modifications with 2 learned latent spaces $\mathcal{L}_i^{(a)}$ (appearance embedding) and $\mathcal{L}_i^{(\tau)}$ (transient embedding) that provide control on the appearance and dynamic content of each rendered view. An additional network $\text{MLP}_{\theta_3}$ is introduced to render scenes with transient objects during training thanks to $\mathcal{L}_i^{(\tau)}$, whereas $\text{MLP}_{\theta_2}$ only render the static part of the scene with $\mathcal{L}_i^{(a)}$ as an additional input, see figure 5.3-a. We train a NeRF-W model on each scene using a sparse set of registered images.

### 5.4.2 Density volume generation

In order to chose valid locations for the synthetic images, we first gather the volumetric representation of the scene learned by the NeRF-W model.

We query $\text{MLP}_{\theta_1}$ on a regular 3D grid that can be displayed as a density volume (see figure 5.3-b and figure 5.5). Lets consider $\mathcal{B}$ the smallest 3D bounding box that contains all the poses of the training images. The 3D grid extend is defined by another 3D bounding box $\mathcal{B}_+$ obtained by extending $\mathcal{B}$ with an extrapolation distance parameter $E_{max}$. We sample $m$ 3D points in $\mathcal{B}_+$ separated by a constant distance $\lambda_v$ that is obtained by dividing the smallest edge of $\mathcal{B}_+$ by a fixed resolution parameter $r_v$. Using this method we ensure that $m$ will be of the same order of magnitude for all scenes, avoiding the generation of intractable density volume.

To consider that a given location is unreachable in the scene, we set a threshold $t_\sigma$ and only take in account 3D points with density higher than $t_\sigma$.

### 5.4.3 Virtual camera locations

Next, our virtual camera location generation algorithm takes poses of real training images as input, but also the NeRF density volume and the desired number of virtual cameras $n$. Our method is two-step: defining the virtual camera positions and then determining their orientations. Our focus is to generate a training dataset for a pose regressor, therefore we want a set of locations to be uniformly distributed all over the area and viewpoints that could be visited by the agent to localize.

To generate our virtual camera positions candidates, we use a similar strategy as the one described in the previous section: we sample $n_i$ 3D points in $\mathcal{B}_+$ using a constant distance $\lambda_s$ between the 3D points obtain by dividing the smallest edge of $\mathcal{B}_+$ by a resolution parameter $r_i$, $i \in \mathbb{N}$. Then we proceed to a multiple criteria pruning step to remove irrelevant candidates: 3D points closer to $d_\sigma$ to a 3D point from the density volume are considered too close to a structure and are discarded, 3D points further than $d_{max}$ to a real camera view position are considered too far from the area of interest and are also discarded. We implement the nearest neighbour search with KDTrees for efficiency.

After candidate pruning, we obtain a final number of virtual camera positions $\hat{n_i}$. If $\hat{n_i}$ is smaller than desired number of cameras $n$, we update the resolution parameter $r_i$ and repeat the generation procedure: $r_{i+1} = r_i + \sigma_r$, with $\sigma_r$ the update step.

Finally, we need to define a camera orientation $(q_x, q_y, q_z, q_w)$ attached to the camera center $(x, y, z)$ for each virtual camera. In order to avoid degenerated views, we copy the orientation of the nearest pose in training set and add a small random perturbation on each axis drawn uniformly in $[-\frac{\theta}{2}, \frac{\theta}{2}]$, where $\theta$ is the maximum amplitude of the perturbation. As an output, we have a set of $n$ poses $[x, y, z, q_x, q_y, q_z, q_w]$ that are used as queries for novel view synthesis, see figure 5.3-c. More examples of generated poses can be find in figure 5.5.

### 5.4.4 Novel view synthesis

Novel views are synthesised on each virtual camera location using $\text{MLP}_{\theta_1}$ and $\text{MLP}_{\theta_2}$ (figure 5.3-d). The appearance embedding $\mathcal{L}_r^{(a)}$ is chosen by random interpolation of training set appearances defined by:

$$\mathcal{L}_r^{(a)} = \alpha \times \mathcal{L}_i^{(a)} + (1-\alpha) \times \mathcal{L}_j^{(a)}, \tag{5.1}$$

where i and j correspond to random indices of training images and $\alpha \sim \mathcal{U}_{[0,1]}$.

As a result, rendered images depict a scene from chosen viewpoints where transient occluders observed in training images are removed. Appearance interpolation acts as a data augmentation technique that increases robustness of the localization model under varying illuminations. Some example of resulting appearances are shown in figure 5.4.

### 5.4.5 Camera pose regressor training

The final step of our pipeline is to train the localization algorithm with our synthetic-augmented dataset. We simply combine real images with our synthesised images together; Then mini-batches are sampled randomly in this combined dataset for the training stage (figure 5.3-d). We use CoordiNet [Moreau et al., 2022a] as a camera pose regressor.

## 5.5 Experiments

In this section we evaluate LENS combined with CoordiNet on standard localization benchmarks, conduct an ablation study to confirm our design choices and investigate on training poses distribution for pose regressor training.

Figure 5.4: **Appearances interpolation with Nerf-W.** We display renderings from the same viewpoints with randomly interpolated appearance embeddings. We observe that resulting appearances look natural and exhibit a good diversity.

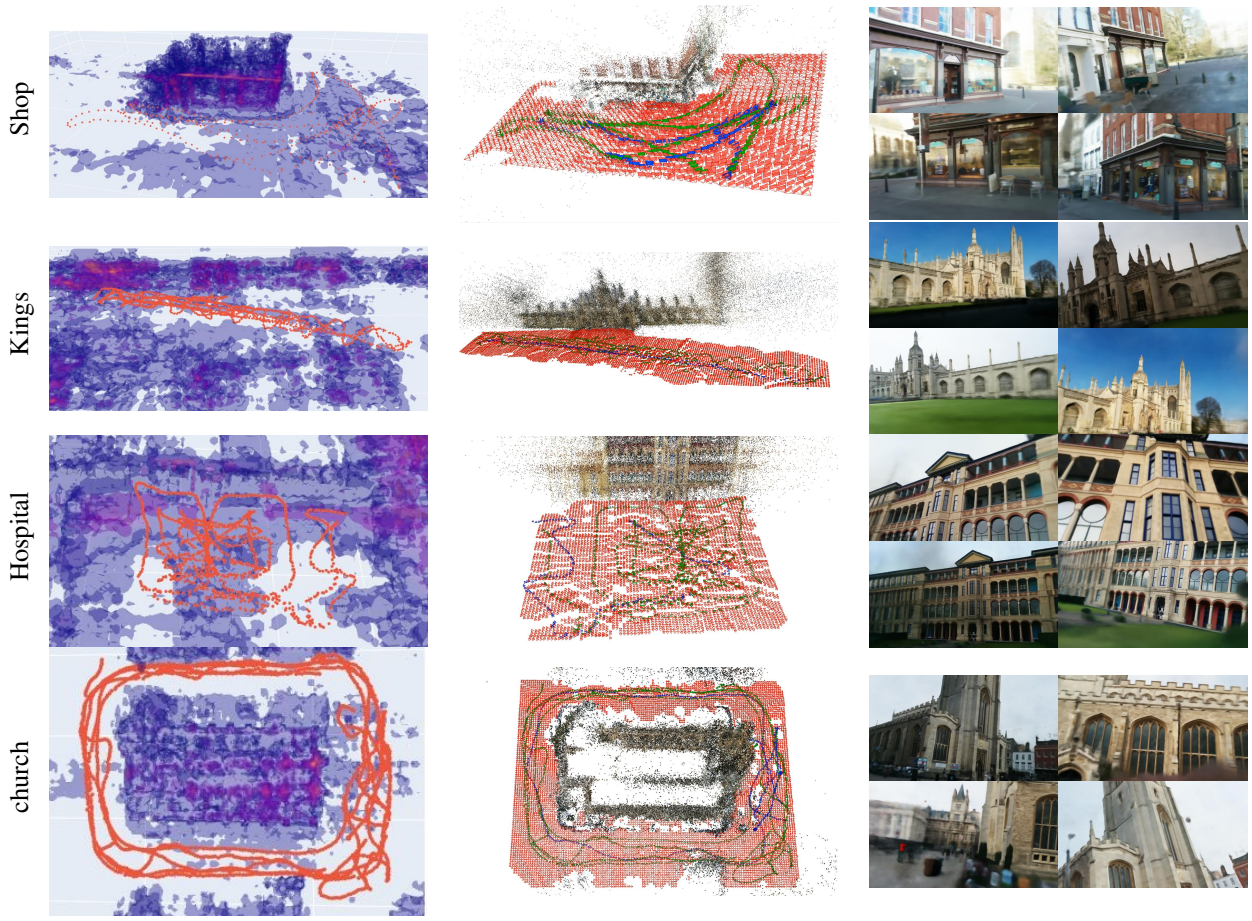


Figure 5.5: Visualisation of density volumes, virtual camera queries (training poses, test poses,virtual cameras) and example of rendered images on Cambridge Landmarks.

### 5.5.1 Datasets and implementation details

**5.5.1.0.1 Datasets.** We evaluate LENS on two standard visual localization benchmarks:

- **Cambridge Landmarks** [Kendall et al., 2015] contains 4 outdoor dynamic scenes captured by a smartphone. In each scene, a common building is visible from each image. Camera poses are recovered from SfM and training sets contains between 200 and 2000 images extracted from videos. We downscale input images to 640×360 pixels.

- **7scenes** [Shotton et al., 2013] consists of 7 static indoor scenes, captured by a Kinect RGB-D sensor. This dataset is challenging for camera pose regression algorithms because test sequences follow different paths than the ones of train sequences. We downscale input images to 320×240 pixels.

**5.5.1.0.2 NeRF parameters.** We use the code of [Quei-An, 2020] to train one NeRF-W [Martin-Brualla et al., 2021] by scene, using the default parameters of the Pytorch implementation. We train each model for 20 epochs using the training images to generate the training rays. We took a maximum of 5 training sequences (5k images) for the 7 scenes dataset and we define one appearance embedding by image in the training set. Generating images of Cambridge scenes takes approximately 40s/GPU, with

256 coarse and 256 fine sampling by ray. It takes 7s/GPU for 7 scenes while sampling only 128 coarse and 128 fine values as the scenes are smaller and less complex than Cambridge scenes.

**5.5.1.0.3 Virtual camera locations generation.** For outdoor scenes, virtual camera locations are sampled on a 2D plane and LENS parameters are: $r_v = 128$, $t_\sigma = 20$, $d_{max} = 8m$, $d_\sigma = 1m$, $E_{max} = 1m$, $r_0 = 1$, $\sigma_r = 1$ and $\theta = 15°$. For indoor scenes, the regular grid is defined on 3 dimensions and we adapt the following LENS parameters: $d_{max} = 50cm$, $d_\sigma = 20cm$, $E_{max} = 20cm$ and $\theta = 20°$.

We set the desired number of virtual views $n$ to 500% for outdoor scene and 1000% for indoor scene of the total number of real training images. We found that amount of images a good trade-off between computational cost and localization accuracy. More details are provided in section 5.5.3.

**5.5.1.0.4 Pose regressor.** We use CoordiNet [Moreau et al., 2022a] as camera pose regressor with EfficientNet-b3 [Tan and Le, 2019] as backbone and optimize the network by maximizing heteroscedatic log-likelihood (see section 3.4.2). We train our models for 250 epochs with a fixed learning rate of $1e-4$ for both datasets. During training, we use a batch size of 10 for Cambridge Landmarks and 40 for 7 scenes.

### 5.5.2 Comparison with related localization methods

**5.5.2.0.1 Competitors.** We compare our method CoordiNet + LENS with CoordiNet only trained on real images. SPP-Net [Purkait et al., 2018] is a small CNN pose regressor that has been trained with and without additional synthetic data, providing a good baseline for LENS. We also report results from TransPoseNet[Shavit et al., 2021b], which is a state of the art transformer approach for camera pose regression. DirectPoseNet [Chen et al., 2021a] uses a NeRF model as a photometric supervisor during the training. Active search [Sattler et al., 2012] is a baseline for structure-based methods, where local image features are matched against a 3D point cloud obtained by SfM.

**5.5.2.0.2 Results.** Median translation and orientation errors are reported in table 7.1. CoordiNet + LENS achieves best reported results for camera pose regression methods on both Cambridge Landmarks and 7scenes with a large margin. Moreover, our method is more accurate than Active Search [Sattler et al., 2012] on 5 scenes out of 11, reducing the gap between camera pose regression and structure-based methods. Then, we observe that LENS provides better relative localization improvement (approximately +60%) than the synthetic datasets generated by [Purkait et al., 2018] (+45%).

### 5.5.3 Ablation studies

We evaluate several components of our method independently:

- we investigate the optimal quantity of training sample that should be generated in to improve accuracy while keeping a reasonable computation time during the novel view synthesis process.
- we study the benefit of using side information provided by the trained NeRF model: the density volume in the pose queries algorithm and the appearances embedding interpolation in order to produce visually heterogeneous training sample.

Table 5.1: **6DOF localization errors of visual localization methods.** We report median translation/orientation error (meters/degrees). TransPN and DirectPN stand for TransPoseNet and DirectPoseNet. Superscripts numbers refer to the relative improvement (green) or deterioration (red) in percentage brought by synthetic data.

| Dataset | Camera Pose Regression | | | CPR + view synthesis | | | 3D |
|---|---|---|---|---|---|---|---|
| Cambridge | SPPNet | CoordiNet | TransPN | DirectPN | SPPNet + synth | **CoordiNet + LENS** | Active search |
| K College | 1.91/2.4 | 0.70/0.9 | 0.60/2.4 | - | $0.74^{61}/1.0^{58}$ | $\mathbf{0.33}^{53}/\mathbf{0.5}^{44}$ | 0.42/0.6 |
| OldHosp | 2.51/3.7 | 0.97/2.1 | 1.45/3.1 | - | $2.18^{13}/3.9^{5}$ | $\mathbf{0.44}^{55}/\mathbf{0.9}^{57}$ | 0.44/1.0 |
| Shop | 1.31/7.8 | 0.69/3.7 | 0.55/3.5 | - | $0.59^{55}/2.5^{68}$ | $\mathbf{0.27}^{61}/\mathbf{1.6}^{57}$ | 0.12/0.4 |
| Church | 3.21/7.0 | 1.32/3.6 | 1.09/5.0 | - | $1.44^{55}/3.3^{53}$ | $\mathbf{0.53}^{60}/\mathbf{1.6}^{56}$ | 0.19/0.5 |
| Average | 2.24/5.2 | 0.92/2.6 | 0.91/3.5 | - | $1.24^{45}/2.7^{48}$ | $\mathbf{0.39}^{58}/\mathbf{1.2}^{54}$ | 0.29/0.6 |
| 7scenes | | | | | | | |
| Chess | 0.22/7.6 | 0.14/6.7 | 0.08/5.7 | 0.10/3.5 | $0.12^{45}/4.4^{42}$ | $\mathbf{0.03}^{79}/\mathbf{1.3}^{80}$ | 0.04/2.0 |
| Fire | 0.37/14.1 | 0.27/11.6 | 0.24/10.6 | 0.27/11.7 | $0.22^{41}/8.9^{37}$ | $\mathbf{0.10}^{63}/\mathbf{3.7}^{68}$ | 0.03/1.5 |
| Heads | 0.22/14.1 | 0.13/13.6 | 0.13/12.7 | 0.17/13.1 | $0.11^{50}/8.3^{41}$ | $\mathbf{0.07}^{63}/\mathbf{5.8}^{57}$ | 0.02/1.5 |
| Office | 0.32/10.0 | 0.21/8.6 | 0.17/6.3 | 0.16/6.0 | $0.16^{50}/5.0^{50}$ | $\mathbf{0.07}^{67}/\mathbf{1.9}^{78}$ | 0.09/3.6 |
| Pumpkin | 0.47/10.2 | 0.25/7.2 | 0.17/5.6 | 0.19/3.9 | $0.21^{55}/4.9^{52}$ | $\mathbf{0.08}^{68}/\mathbf{2.2}^{69}$ | 0.08/3.1 |
| Kitchen | 0.34/11.3 | 0.26/7.5 | 0.19/6.8 | 0.22/5.1 | $0.21^{38}/4.8^{58}$ | $\mathbf{0.09}^{65}/\mathbf{2.2}^{71}$ | 0.07/3.4 |
| Stairs | 0.40/13.2 | 0.28/12.9 | 0.30/7.0 | 0.32/10.6 | $0.22^{45}/7.2^{45}$ | $\mathbf{0.14}^{50}/\mathbf{3.6}^{72}$ | 0.03/2.2 |
| Average | 0.33/11.6 | 0.22/9.7 | 0.18/7.8 | 0.20/7.3 | $0.18^{45}/6.2^{47}$ | $\mathbf{0.08}^{64}/\mathbf{3.0}^{69}$ | 0.05/2.5 |

**Synthetic dataset size.** We change the amount of generated sample from 100% to 1000% (up to 5000% for fire scene) of the total number of images in the real training set on three different scenes and we compare the relative improvement in term of localization accuracy. Results are shown in table 5.2.

As expected, we observe that using a higher grid resolution containing more samples leads to a better localization. In addition to that, we can see in figure 5.6 that the relative improvement compared to the baseline (*i.e.* no synthetic images) is correlated to the ratio between synthetic and real images rather than the total number of images itself: a ratio of 10 leads to a 59% translation improvement for ShopFacade, 58% for Church and 63% for Fire. Curves of error translation in figure 5.6 do not seems to reach a plateau: an higher ratio would potentially bring better localization results. This suggests that optimal training strategy is to generate a maximum number of synthetic samples for a given computation time budget.

| Synth. im. / Real im. | **ShopFacade** (231 im.) | **Church** (1487 im.) | **Fire** (2000 im.) |
|---|---|---|---|
| **0%** | 0.61/4.2 | 1.06/3.1 | 0.27/11.6 |
| **100%** | 0.61/2.4 | 0.75/2.4 | 0.18/5.7 |
| **200%** | 0.41/2.0 | 0.58/1.9 | 0.16/5.7 |
| **500%** | 0.26/1.2 | 0.51/1.5 | 0.12/4.2 |
| **1000%** | 0.25/1.2 | 0.45/1.6 | 0.10/3.2 |
| **5000%** | - | - | 0.07/2.4 |

Table 5.2: median translation (m) and orientation (°) errors depending on synthetic dataset size

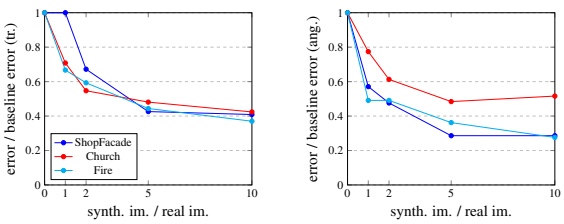


Figure 5.6: Translation (tr., left) and angular (ang., right) error relative decrease vs synthetic dataset size.

**Benefit of using volume and random appearances.** In table 5.3, we observe that occluded views located too close or inside a building disturb the training of the pose regressor, decreasing the localization accuracy. We can see in figure 5.7 that the use of the density volume provided by the trained NeRF correctly remove distracting samples. Providing a random appearances embedding during the NeRF

image generation also decrease slightly translation and rotation errors. Even if the improvement is minor and not very significant on this scene, we expect this appearance augmentation to be very useful on experiments with more diversity in illuminations (day and night images for example).

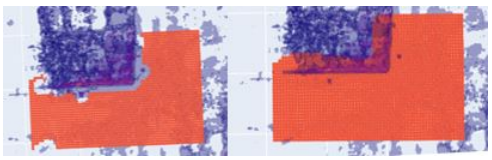

Figure 5.7: Virtual camera locations with (left) and without (right) NeRF volume pruning step.

| Errors | Volume | | Appearances emb. | |
|---|---|---|---|---|
| | with | without | random | constant |
| Translation | 0.27m | 0.36m | 0.25m | 0.26m |
| Rotation | 1.6° | 1.7° | 1.2° | 1.4° |

Table 5.3: Localization errors comparison for density volume and appearances embedding ablations on Shop Facade scene.

### 5.5.4 Exploring camera pose regression on synthetic domain

In order to investigate the impact of training camera poses distribution on the performance of pose regression without taking in account the domain gap between real and synthetic data, we perform the following experiment: we replace both training and testing real images by NeRF-rendered images at the exact same location, then we trained and test a pose regressor on this synthetic dataset. Results are reported for the indoor scene fire in the two first columns of table 5.4 (seq3-4 real vs seq3-4 synth.). We get similar localization performances compared to the same model trained on real data, which allow us to perform deeper analysis on the impact of distribution of camera poses used for the training.

Table 5.4: Median translation (m) and orientation (°) errors in Fire scene from 7scenes. seq3-4 refers to methods using images from sequences 3 & 4 as training data.

| **Method** | seq3-4 | seq3-4 | LENS | LENS | LENS | Act. Search |
|---|---|---|---|---|---|---|
| **Data type** | real | synth. | synth. | synth. | synth. | real |
| **Train set size** | 2k | 2k | 2k | 10k | 100k | - |
| **Error (m/°)** | 0.27/11.7 | 0.27/10.6 | 0.08/3.5 | 0.05/2.4 | 0.03/1.4 | 0.03/1.5 |

In a second experiment, we replace the original training camera poses by uniformly distributed poses generated by LENS. We observe a important decrease in median localization error from 27cm/11.7° to 8cm/3.5° whereas we use the same number of training data (table 5.4 columns 2 vs 3). Increasing the number of synthetic data, up to 5000% of the number of training data in the original training set, leads to even better localization results (table 5.4 columns 3 to 5). We also show that pose regression can reach structure-based method accuracy (evaluated on real data) if the model is trained with the largest amount of data. A similar experiment had been made by [Sattler et al., 2019] with opposite conclusions: even with a big synthetic dataset depicting the scene, pose regressors were not able to reach an accuracy comparable to structure-based methods, suggesting that camera pose regression algorithms are inherently limited by their approach without geometrical constraints.

From these experiments, we end up with a different conclusion: datasets using only video sequences create an unbalanced regression problem where training labels does not cover the entire set of possible poses and then lead to a poor localization accuracy. Deep learning approaches are known to perform well in high data regimes, but the main finding of this research is that training datasets distributed across the entire scene are crucial for camera pose regression performances. The different results we observe

compared to [Sattler et al., 2019] probably come from a higher quality of our synthesized views and a pose regression architecture that performs better.

### 5.5.5 Limitations

The first limitation of our pipeline is the offline computation time: Nerf-W needs to be trained several days with a GPU on a single scene in order to reach optimal rendering results. The slow rendering of novel views forces us to generate a dataset before training the localization algorithm instead of an online data generation pipeline. Faster synthesis would enable to learn from an infinite quantity of data instead of a limited number of images. However, faster NeRF training and rendering are active research fields (see section 4.4.1) that should enhance speed and performances of LENS in the future. Moreover, as reported in Table 5.5, our method offsets the costly offline computation by a fast and light online localization, enabling real-time embedded applications for robotics.

We also observed that training with only synthetic images leads to poor performances on real images. This is due to the domain gap between real and synthetic images: no dynamic objects, different textures, more blur and some artifacts are observable in the synthetic domain. Mixing real and synthetic images is the simpler way to mitigate this issue, however domain adaptation techniques [Cortes and Mohri, 2011] could be used to reduce domain discrepancy as well as higher quality rendered samples.

Table 5.5: Approximate time and memory requirements comparison between structure-based methods and ours. $^*$ denotes a scaling time and memory consumption according to the scene size. $^\dagger$ structure-based methods need to load the features vocabulary and access to the 3D points cloud during localization.

| | | **Offline (1 scene)** | | | | **Online (1 image)** | |
|---|---|---|---|---|---|---|---|
| **Structure based** | **Task** | **SfM** | | **Quantization** | | **SIFT** | **Loc.** |
| | Timing | Hours$^*$ | | Minutes$^*$ | | 50-500ms | ~500ms$^*$ |
| | Memory | NR | | NR | | NR | GBs$^{*\dagger}$ |
| **LENS + CoordiNet** | **Task** | **SfM** | **Train NeRF** | **NVS** | **Train PR** | **Coordinet inference** | |
| | Timing | Hours$^*$ | Hours$^*$ | 30s/im/gpu$^*$ | Hours$^*$ | ~50ms | |
| | Memory | NR | NR | NR | NR | <50MB | |

## 5.6 Conclusion

In this chapter, we address limitations of camera pose regressors at data-level: thanks to high quality novel view synthesis provided by NeRF-W, we propose to train a relocalization algorithm with synthetic images uniformly sampled on the entire scene. Our method enhance strongly localization performances, reducing the gap with structure based methods while keeping the advantages of pose regression: a fast inference with low memory footprint and the capability to scale to large environments. Our experiments show that camera pose regression can perform well when trained with large and diverse datasets. LENS coupled with CoordiNet improves camera pose regression state of the art and can be used for accurate real-time robot relocalization system.

Applying this idea for autonomous vehicle scenarios is feasible, for example by using Block-NeRF [Tancik et al., 2022] for the rendering part, but would require huge computational budget. We left these experiments as future work. However, the neural renderer proposed in chapter 7 is way more efficient than Nerf-W and could facilitate these experiments.

Chapter 6

# Large scale localization with an implicit map representation

## Contents


**6.1 Résumé en Français** . . . . . **70**
**6.2 Motivation** . . . . . **71**
**6.3 Visual Localization with ImPosing** . . . . . **72**
6.3.1 ImPosing localization process . . . . . 72
6.3.2 Training procedure . . . . . 74
**6.4 Experiments** . . . . . **75**
6.4.1 Single scene localization . . . . . 76
6.4.2 Multi-scene localization . . . . . 77
6.4.3 Continual learning in the real world . . . . . 78
6.4.4 Smaller scale benchmarks . . . . . 79
6.4.5 Efficiency comparison . . . . . 80
6.4.6 Ablation study . . . . . 81
6.4.7 Qualitative analysis . . . . . 82
**6.5 Discussion** . . . . . **83**
**6.6 Conclusion** . . . . . **84**

## 6.1 Résumé en Français

La solution habituelle pour résoudre le problème de localisation visuelle dans de très grandes cartes consiste à utiliser des algorithmes de reconnaissance de lieu, basés sur le principe d'Image Retrieval.

Ce chapitre étudie l'idée de remplacer la base de données de ces méthodes par un réseau de neurones qui représente implicitement les kilomètres de paysages observé par un véhicule. Dans cette formulation, le descripteur global de "n'importe quelle" position dans la carte peut être calculé. Par conséquent, la localisation n'est pas effectuée en comparant l'image de requête à une base de données discrète et finie, mais plutôt en échantillonnant et évaluant des positions dans l'espace continu de la carte.

Nous confirmons les avantages de notre méthode ImPosing sur des scénarios de conduite très divers ainsi que sa capacité à passer à l'échelle sur de larges jeux de données collectés au fil du temps.

## 6.2 Motivation

We aim to develop relocalization algorithms able to operate efficiently in embedded devices of autonomous vehicles in a deployment scenario where the target area is wide and collected datasets are large. This problem is challenging due to kilometer-scale maps and dynamic outdoor environments.

The well-established solution to address visual localization in large-scale maps is to use Image Retrieval (see section 2.2). While effective, this solution has a computational complexity that grows with the amount of collected data.

We also observed in chapter 3 that Absolute Pose Regression with CoordiNet was able to provide a sub-meter median accuracy in large maps. Because it requires scene-specific training that can be quite long, training from scratch independently for each map seems inefficient. We would like to have a separate entity which represent the map. During the research phase we first implemented discrete grids of learnable parameters that memorized the global descriptor of images observed at the same location. Problem: the accuracy is bounded by the resolution. Then we realized that this map information could be represented implicitly by a coordinate-based neural network.

This chapter proposes an algorithm in between Image Retrieval (IR) and Absolute Pose Regression (APR). IR uses one neural network for global feature extraction and a discrete set of observed images to represent the map. APR entangles feature extraction and map memorization in a single neural network. Our proposal uses 2 neural networks : one for features extraction and one for map memorization. It can either be seen as "inverting" the decoder of pose regression, or replacing the image retrieval database by an implicit representation of the map.

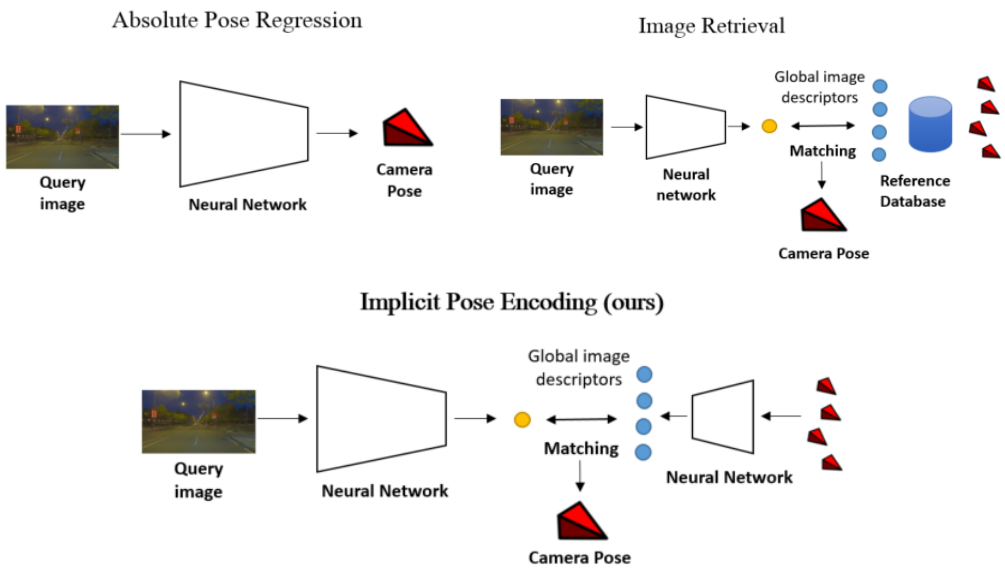


Figure 6.1: **High level comparison between Absolute Pose Regression (top left), Image Retrieval (top right) and the proposed algorithm (bottom).** APR entangles the entire visual localization task in a single neural network. IR extracts a global image descriptor from the query image and compares it to a database of descriptors. Our proposal models the reference database as a neural network.

The core idea is to connect image and camera pose representations, which are learned separately by two distinct neural networks, in a common latent space. We use an implicit neural representation to encode a specific viewpoint in the scene (i.e. a 6-DoF camera pose) into a higher dimensional vector.

With this formulation, the continuous representation of any camera pose in the scene (even a pose not observed in reference images) can be computed in a single network forward pass. We take advantage of this property to solve the localization task by searching the pose candidates which are the most similar to the learned image representation. To do so, we introduce a hierarchical sampling process able to retrieve the correct camera viewpoint using only a few batched queries on the pose encoder network. Our localization method, called Implicit Pose Encoding (ImPosing), provides real-time sub-metric localization performances that can be rapidly deployed on large areas.

We evaluate our system on a wide range of visual localization datasets, including several kilometers-scale road environments with challenging conditions (seasonal and appearance changes, limited training data). We observe that our method outperforms its regression-based competitors in terms of accuracy and training efficiency, especially in large-scale scenarios.

## 6.3 Visual Localization with ImPosing

Our method, ImPosing, estimates the 6-DoF camera pose $(t, q) \in SE(3)$ of a query image $I$, where $t$ is a translation vector and $q$ is a unit quaternion. We train our solution using a reference dataset of posed images $(I_k)$ collected in the target area and we do not make use of an additional 3D model of the scene.

The proposed algorithm, computes a vector that represents the image through the image encoder. Then, the camera pose is searched by evaluating initial pose candidates distributed across the map. Poses are processed by the pose encoder to produce a latent representation that can be matched against the image vector. Each pose candidate receives a score, based on distance to camera pose. High scores provide a coarse localization prior which is used to select new candidates. By repeating this process several times, our pool of candidates converges to the actual camera pose.

### 6.3.1 ImPosing localization process

This section describes the localization process step by step from the image to the final camera pose estimate, displayed in figure 6.2,

**6.3.1.0.1 1. Image encoder:** we compute a global image features vector $f_I(I) \in \mathbb{R}^d$ from the input query $I$ using our image encoder. The encoder architecture consists in a pretrained CNN backbone followed by a Global Average Pooling, and a fully-connected layer with $d$ output neurons. The feature vector is one order of magnitude smaller than global image descriptors commonly used in image retrieval (we use $d = 256$ whereas AP-GeM [Revaud et al., 2019] use $d = 2048$) in order to efficiently compare it to a large set of pose candidates at later steps.

**6.3.1.0.2 2. Initial pose candidates:** Our starting point is a set of $N$ camera poses $(p_n)_0$, sampled from the set of reference poses (= training poses). Through this initial selection, we introduce a prior for the localization process, similar to the anchors poses in [Saha et al., 2018] or regression methods that compute relative instead of absolute pose [Ding et al., 2019]. We observed that the algorithm is robust to this choice: a 2D grid on the map yield similar results.

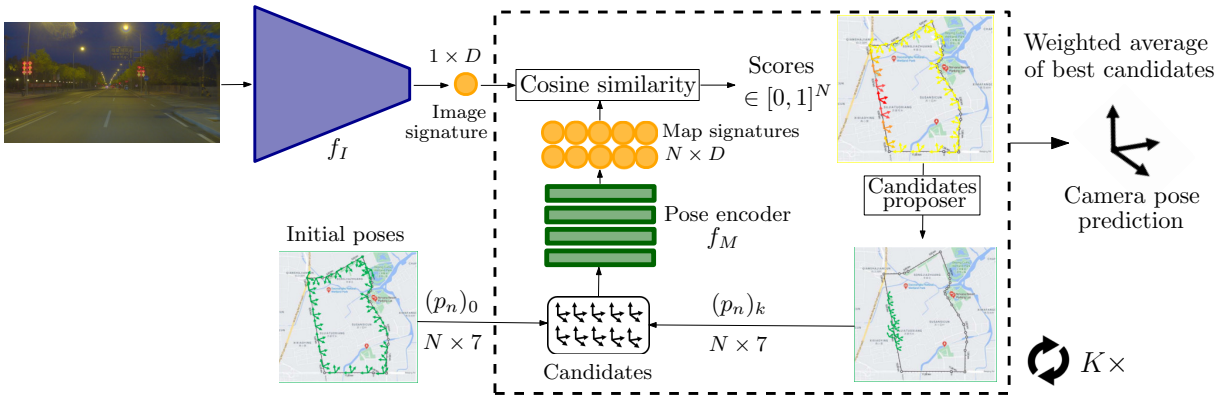


Figure 6.2: **Implicit pose encoding for hierarchical image localization.** A set of initial map signatures is compared to the image signature to determine the most probable localization of the camera. The similarity scores guides the selection of a new batch of pose candidates that are used to compute the new map signatures for the second refined localization step. This process is repeated multiple time to predict the final camera pose.

**6.3.1.0.3 3. Pose encoder:** Pose candidates are processed by a neural network which outputs latent vectors. This implicit representation learns the correspondence between camera viewpoints in a given scene and features vectors provided by the image encoder. First, following Tancik et al. [Tancik et al., 2020], each component of the camera pose $(tx, ty, tz, qx, qy, qz, qw)$ is projected to higher dimension using Fourier features : $x \rightarrow (x, sin(2kx), cos(2kx))_{0 \leq k \leq 10}$, as it helps networks with low dimensional input to fit high frequency functions. Then, we use a MLP $f_M$ with 4 layers of 256 neurons and ReLU activations on hidden layers. Each set of pose candidates is computed in a single batched forward pass.

**6.3.1.0.4 4. Similarity scores:** we obtain a similarity score $s$ by computing the cosine similarity between $f_I(I)$ and $f_M(p)$ for each image-pose pair $(I, p)$. We add a ReLU layer after the dot product, such that $s \in [0, 1]$. Intuitively, we aim to learn high scores for poses candidates close to the actual camera pose. With this formulation, we can evaluate hypotheses on the camera pose and search for pose candidates with high scores. Formally, our score is defined by:

$$s(I, p) = \frac{\langle f_I(I), f_M(p) \rangle}{\|f_I(I)\| \|f_M(p)\|} \mathbb{1}_{\langle f_I(I), f_M(p) \rangle > 0} \tag{6.1}$$

**6.3.1.0.5 5. Candidates proposer:** new poses $(p_n)_k$ are selected for the $k^{th}$ iteration based on scores obtained with poses $(p_n)_{k-1}$ at the previous iteration. First, we select the poses with top $B = 100$ higher scores $(h_i)_{0 \leq i < B} \subset (p_n)_{k-1}$. Then, new candidates are sampled from $(h_i)$ in a Gaussian Mixture Model with density:

$$P(x) = \sum_{i=1}^{100} \pi_i \mathcal{N}(x | h_i, v/k) \quad \text{where} \quad \pi_i = \frac{s(I, h_i)}{\sum_{l=1}^{100} s(I, h_l)}. \tag{6.2}$$

$v = [v_{tx}, v_{ty}, v_{tz}, v_{rx}, v_{ry}, v_{rz}]$ is the variance of the sampling process, a hyperparameter composed of a translation vector and Euler angles.

**6.3.1.0.6 6. Iterative pose refinement:** we repeat $K$ times the evaluation of pose candidates described in steps 3-4-5. After each iteration, the noise vector $v$ is divided by 2, such that new candidates are sampled closer to previous high scores. As a result, we can converge to a precise pose estimate in kilometers-scale maps while only evaluating a limited sparse set of poses. We evaluate each camera frame independently at each time step, however one could use localization priors from previous time steps to reduce the number of iterations in vehicles navigation scenarios. An example of selected poses at each iteration is shown in Fig. 6.3. By sampling $N$ candidates for initial poses, we preserve a constant memory peak.

**6.3.1.0.7 7. Pose averaging:** our final camera pose estimate is a weighted average of the 256 pose candidates with higher scores, which exhibits better interpolation properties than selecting the best score pose. We use scores as weighting coefficients and 3D rotation averaging is implemented following [Markley et al., 2007].

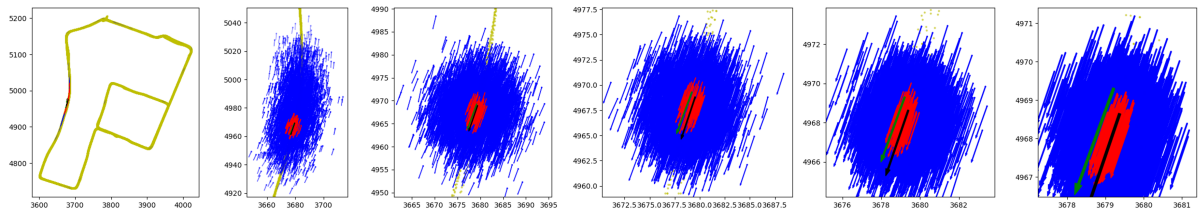

Figure 6.3: **Iterative candidates refinement.** At each $k$ step of the localization process, top scored poses are selected to sample the new candidate poses at step $k+1$. From left to right: top scored poses at $k=0$ to $k=5$, yellow points are positions of the training example, blue arrows are pose candidates and red arrows are the selected poses among the candidates.

The entire inference procedure requires 1 forward pass on the image encoder and $K$ passes on the pose encoder.

### 6.3.2 Training procedure

We do not train the system by minimizing the error on the final camera pose estimate. Instead, we apply our loss function directly on the predicted scores. As a result, one training iteration provides supervision on the $K \times N$ image-pose pairs that contains more information than the single localization error. We observed that this property results in superior training efficiency than regression approaches (see 6.4.6). We define target scores $s_t$ based on translation and rotation distances between the camera pose $p_I = (t_I, q_I)$ and the candidate pose $p = (t, q)$:

$$s_t(I,p) = ReLU\left(1 - \lambda_t \|t_i - t\|_2 - \lambda_r G(q_i, q)\right) \tag{6.3}$$

where $\lambda_t$ and $\lambda_r$ are weighting parameters set to 5 and 0.1 and $G$ is geodesic distance, defined as the minimal angle between 2 rotations:

$$G(q_1, q_2) = cos^{-1}\left(\frac{(tr(M_{q_1} M_{q_2}^{-1}) - 1)}{2}\right), \tag{6.4}$$

$M_q$ being the 3D rotation matrix associated with rotation $q$.

We train $f_I$ and $f_M$ by computing scores between reference images and pose candidates sampled at $K$ different resolutions as described in section 6.3.1. For training purpose, we add to initial poses an uniform noise sampled in $[-v, v]$ as we observed that it reduces overfitting. We also use poses associated with the top target scores in the candidates proposer, in addition with top predicted scores in order to guide training convergence in early iterations.

Finally, our optimization objective is:

$$L = \frac{1}{N} \sum_{k=0}^{K} \sum_{n=0}^{N-1} |s(I, p_{n,k}) - s_t(I, p_{n,k})| \tag{6.5}$$

An analogy can be made with content-based image retrieval [Arandjelović et al., 2016, Revaud et al., 2019]: global descriptors are usually trained using image triplets composed of a query image, a positive and a negative example. Positive samples are data close to the query, in metric or semantic domain depending on the final application, and negative samples are images with unrelated content to the query. Global descriptors can be trained by minimizing a triplet margin loss [Arandjelović et al., 2016]. In our case, positive examples are the poses with a non-zero score whereas negative examples are candidates farther from the camera pose than an arbitrary threshold. Instead of binary classification (positive or negative example), we rank the relative importance of the positive samples according to their distance to the ground truth label.

## 6.4 Experiments

We compare our approach against recent methods on several datasets covering a wide range of autonomous driving scenarios in large scale outdoor maps. This task is highly challenging due to the dynamic part of outdoor environments (moving objects, illumination, occlusions, etc.). We verify that our formulation enables accurate localization in 9 different large outdoor scenes. Then we show that our method can be naturally extended to multi-map scenarios and we report results using this setup. We also compare the computational efficiency of our method with competitors and finally present an ablation study on hyperparameters of ImPosing.

**6.4.0.0.1 Implementation details:** ImPosing is implemented in PyTorch. Images are computed at a small resolution $135 \times 240$. The image encoder uses a ResNet34 backbone pretrained on ImageNet. $N = 4096$ pose candidates are evaluated at each of the $K = 6$ refinement steps. For candidates sampling, the noise vector is set to $v = [8.0m, 0.2m, 8.0m, 1deg, 5deg, 1deg]$ where y is the altitude axis, and we use 100 GMM components. We train the image encoder and pose encoder for 250 epochs with Adam optimizer at a constant learning rate of $1e^{-4}$. We did not tune these parameters specifically for each scene, suggesting that they should work for any autonomous driving scene.

**6.4.0.0.2 Baselines:** Our first aim is to compare ImPosing to its direct learning-based methods competitors. We use CoordiNet [Moreau et al., 2022a] that report state-of-the-art results for absolute pose regression on Oxford Dataset as a baseline. We report previously published results on this dataset, and our own implementation for other datasets. We replace the EfficientNet backbone by ResNet34 for a

fair comparison with ImPosing. We share similarities with image retrieval by matching a global descriptor against the map. To compare ImPosing to retrieval, we use NetVLAD [Arandjelović et al., 2016] (VGG16 backbone) and AP-GeM. [Revaud et al., 2019] (GeM pooling, Resnet101 backbone) publicly available implementations[1]. Full sized images are used to compute global image descriptors followed by cosine similarity for features comparison, then we perform pose averaging on poses of top 20 database images as in [Sattler et al., 2019]. Scene coordinate regression [Brachmann and Rother, 2019, Brachmann and Rother, 2021] can not scale to large environments thus is not considered for evaluation. We did not conduct experiments with structure-based methods [Sarlin et al., 2019, Sarlin et al., 2021, Zhou et al., 2020]. These methods are more accurate than ours thanks to geometric reasoning with a 3D model, but also operate at a different computation scale than ours (see figure 6.6) making embedded deployment difficult. In scenarios where it can be afforded, ImPosing can be considered as a coarse localization step, followed by refinement with a 3D model, similar to HLoc[Sarlin et al., 2019] architecture.

### 6.4.1 Single scene localization

Table 6.1: Localization error on Oxford RobotCar and Daoxiang Lake datasets.

| | | Pose regression | | Image retrieval | | ImPosing |
|---|---|---|---|---|---|---|
| Dataset | | CoordiNet | AtLoc | NetVLAD | GeM | |
| Oxford Full | Median | 3.55m/1.1° | 11.1m/5.3° | 1.42m/1.4° | **1.36m/1.3°** | 1.90m/1.3° |
| | Mean | 14.96m/5.7° | 29.6m/12.4° | 4.47m/2.4° | **3.49m/2.3°** | 4.25m/4.3° |
| Oxford Loop | Median | 2.27m/0.9° | 5.36m/2.1° | 2.16m/1.1° | 2.39m/1.0° | **1.93m/1.0°** |
| | Mean | 4.15m/1.4° | 8.73m/4.6° | 4.16m/1.9° | 6.92m/3.1° | **3.03m/1.8°** |
| Average | Median | 2.91m/1.0° | 8.23m/3.7° | **1.79m**/1.2° | 1.88m/**1.1°** | 1.92m/**1.1°** |
| | Mean | 9.56m/3.4° | 19.17m/8.5° | 4.32m/**2.1°** | 5.20m/2.7° | **3.64m**/3.0° |
| Daoxiang Lake | Median | 6.82m/0.4° | – | 8.92m/0.8° | 27.13m/1.1° | **1.62m/0.3°** |
| | Mean | 25.18m/1.0° | – | 152.2m/15.5° | 328.8m/19.5° | **8.40m/0.5°** |

**6.4.1.0.1 Oxford RobotCar [Maddern et al., 2017]** contains images recorded by a vehicle in Oxford over a year. We reproduce experiments commonly reported for learning-based methods [Moreau et al., 2022a, Wang et al., 2020a, Xue et al., 2020]: we evaluate on the *Loop* and *Full* scenes, using only 2 sequences for training. Results are reported in Table 6.1.

First we observe that image retrieval performs better than pose regression. Previous learning-based methods struggle due to the low-data regime [Moreau et al., 2022a, Moreau et al., 2022b] and the decrease of the regression accuracy in large maps. Oxford city is an environment with rich features similar to visual place recognition training datasets, that make NetVLAD [Arandjelović et al., 2016] and GeM [Revaud et al., 2019] strong baselines in this scenario. ImPosing exhibits state-of-the art accuracy on Oxford Loop scene, as well as the best mean error in average. These results are obtained by reducing a lot the number of large failure cases that occur with prior methods.

We also observe that despite newly provided RTK ground truth provided by the authors [Maddern et al., 2020], the reference poses are largely inaccurate in some areas. As a result, evaluation metrics

[1] https://github.com/Nanne/pytorch-NetVlad and https://github.com/naver/deep-image-retrieval

are not significant at a centimeter level and models training might be impacted by this erroneous pose labels. For this reason, we conduct a benchmark on two recently released datasets with more reliable ground-truth.

**6.4.1.0.2 Daoxiang Lake [Zhou et al., 2020]** has been collected in a 12km loop in Beijing during 4 months. 8 recordings are available, we use 7 for training and 1 for testing with images from the front camera only. This scene contains the largest map and training dataset of our experiments. Median and mean errors are shown in Table 6.1. Daoxiang Lake is a more challenging dataset than Oxford because of repetitive areas with few discriminative features and various environments (urban, peri-urban, highways, nature, etc.). As a result, image retrieval performs worse than pose regression. ImPosing is way more accurate and exhibits a median error 4 times smaller than competitors.

**6.4.1.0.3 4 seasons [Wenzel et al., 2020]** contains data recorded in Munich area in various scenes (city, residential neighborhoods, countrysides) with varying seasonal conditions. We selected 6 scenes where at least 3 different recordings are provided: we use 1 for testing and others as training images. This benchmark is highly challenging due to extreme appearance changes between sequences, small data regime for some scenes, featureless environments (shown in Figure6.4) and kilometers-scale maps. Results are reported in table 6.2.

Table 6.2: Median localization error on 4Seasons dataset.

| | Dataset details | | | Image retrieval | | CoordiNet | ImPosing | |
|---|---|---|---|---|---|---|---|---|
| | Road length | Runs | Images | NetVLAD | GeM | | Single sc. | Multi sc. |
| Neighborhood | 2000 | 6 | 16520 | 0.72m/0.9° | 0.69m/0.9° | 0.74m/**0.6**° | **0.53m**/0.7° | 0.82m/1.0° |
| Office loop | 2600 | 5 | 20915 | 6.85m/3.0° | 6.39m/2.8° | 6.25m/1.5° | **0.99m/1.1**° | 1.58m/1.3° |
| Countryside | 6200 | 3 | 19804 | 32.24m/1.2° | 30.87m/1.3° | 47.33m/2.9° | **2.61m/0.9**° | 5.46m/1.1° |
| Bus. campus | 1000 | 2 | 6132 | 1.19m/1.3° | 1.96m/**1.2**° | 22.57m/6.0° | **1.16m**/1.3° | 1.70m/1.6° |
| City loop | 10000 | 2 | 17427 | 61.60m/3.5° | 317.4m/6.9° | 584.4m/14.4° | **5.32m/2.4**° | 10.53m/2.5° |
| Old Town | 4500 | 3 | 13959 | 3.45m/**1.2**° | 4.46m/1.6° | 50.83m/3.8° | **2.59m/1.2**° | 3.71m/1.3° |
| Average | - | - | - | 17.67m/1,8° | 60.30m/2,4° | 118.7m/4.9° | **2.2m/1.3**° | 3.97m/1.5° |

First, absolute localization accuracy is very heterogeneous between different scenes. We note that scenes with few training images are the most challenging. In particular, *Countryside* include navigation around fields and *City Loop* is a 10km map where the training dataset is composed of a winter sequence with snow and a rainy sequence with blur on camera lens. In these extreme cases, both pose regression and image retrieval fail to estimate reliable poses, whereas ImPosing is able to provide a coarse localization. With sufficiently large training datasets, our method still exhibits the more precise pose estimation.

### 6.4.2 Multi-scene localization

Learning-based methods for relocalization require scene specific training, inducing heavy computation for potential deployment in several areas at a large scale. Recent work [Blanton et al., 2020, Shavit et al., 2021a] has extended absolute pose regression to multi-scene scenarios. The core idea is to train a system with images from several maps while sharing image encoder parameters that could learn to extract features in a generic way. As our method separate image and map representation, ImPosing

Figure 6.4: **Featureless environments and varying weather conditions.** Test is performed on the image on the right, while the network has been trained with 3 recordings with different lightning conditions. Our method is able to provide a coarse localization in these scenarios, where as image retrieval and pose regression competitors fail.

naturally extends to multi scenes scenarios. To adapt ImPosing to a multi-map scenario, we perform the following modifications: the image encoder backbone is shared between all maps, whereas one specific pose encoder is learned for each scene. We also learn scene specific parameters for the final linear layer of the image encoder, to facilitate image features projection to the desired map representation. We train a multi-scene model on the 6 maps of 4 seasons [Wenzel et al., 2020]. Results are reported in Table 6.2. The model has been trained for 20 epochs only because of computational constraints, but still outperform all competitors except single scenes ImPosing models. While the convergence for a single scene is slower in the multimap formulation (but training a multiscene on $n$ maps is faster than performing $n$ different trainings on each map. The multi-scene formulation enables to localize in huge areas with minimal memory storage requirements (see section 6.4.5).

### 6.4.3 Continual learning in the real world

In chapter 1, we defined as objective the development of a "self-learning" localization system, i.e. the idea of collecting more data over time during deployment and, without human in the loop, use it to obtain a more accurate algorithm which improves with further training on new data. This section evaluates this property related to continual learning [Wang et al., 2021].

We did not conduct such experiments on public benchmarks but on an internal dataset collected in Shanghai. It consists in a loop in an urban area of 3.6 kms. After an initial data collection of several recordings, the area has been reconstructed by COLMAP [Schönberger and Frahm, 2016], providing reference camera poses for all images. At this point, 3 independent recordings are kept apart as the validation set. Then, 2 new recordings are captured everyday, registered in the SfM model in order to have reference poses, and added to the training dataset. Each day, the model is retrained from its current weights using all training data (newly captured and previously existing). While using only new data would be more efficient, using all data avoids the catastrophic forgetting problem [Robins, 1995]. After 14 days, 29 independent recordings were used for training, for a total of 511k images.

The evolution of the localization error on the validation dataset over time is shown in Figure 6.5. First, we observe a precise localization accuracy: 15cm median and 25cm mean errors at the end of the process. We observe a constant improvement of the positioning system over time. The first explanation is simple: the training dataset grows and we already observed in section 3.6.2 that larger datasets results in more accurate localization. The main insight of this experiment is the fact that, by collecting navigation data from the users we can register the images in the SfM map and use it to improve the system automatically through further training. This is a crucial property from the industrial point of view.

Another aspect is that this continual learning formulation have the potential to update the algorithm

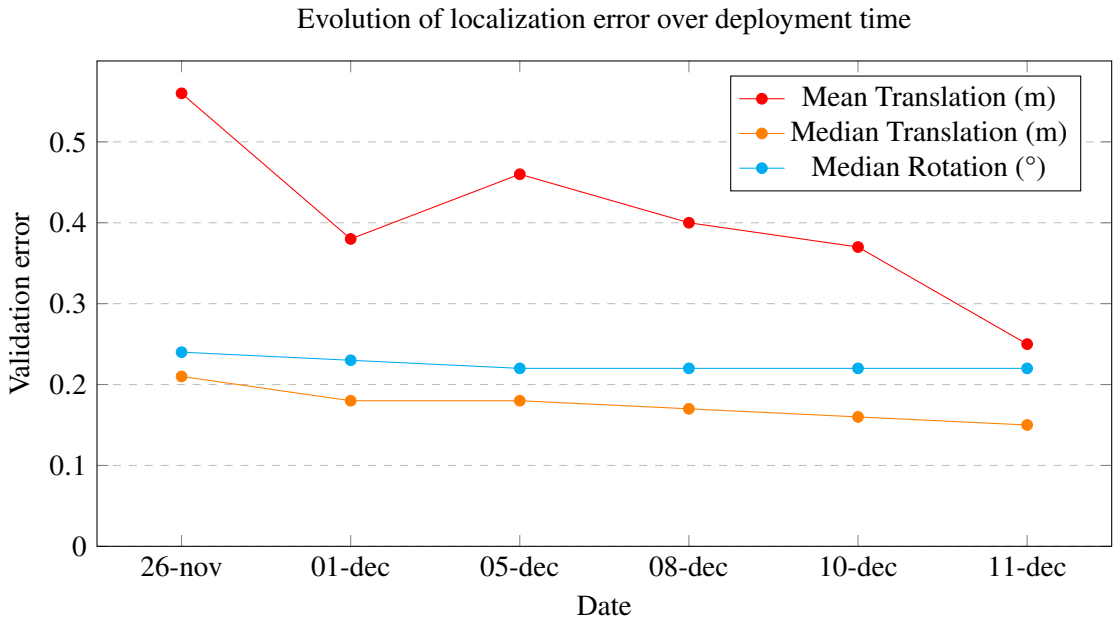


Figure 6.5: **Localization accuracy in continual learning setup.** The model is trained each day with new recordings automatically annotated and improves over time.

when the environment is modified. Examples such as roadworks or building modifications can cause important failures of the visual localization algorithm when the reference data is not up-to-date. Even though we did not evaluate this problem directly, continual learning is an obvious solution for direct learning-based methods.

### 6.4.4 Smaller scale benchmarks

For completeness, we evaluate ImPosing on Cambridge Landmarks [Kendall et al., 2015] and 7scenes [Shotton et al., 2013], which are widely used benchmarks for visual localization. These datasets evaluate visual localization algorithms in smaller environments where camera is carried by a human instead of equipped on a vehicle. Cambridge Landmarks depicts buildings with rich features observed from different viewpoints in outdoor dynamic conditions. 7scenes evaluates localization in small indoor static environments. In both cases, videos used for testing follow different paths than trajectories observed during training. Methods that use a 3D model usually perform well on these benchmarks thanks to geometric reasoning, whereas regression methods overfit on the small amount of training poses and exhibit a limited accuracy (see chapter 5). Results are reported in table 6.3.

On Cambridge Landmarks, ImPosing is less accurate than state-of-the-art pose regression and image retrieval competitors. However, in the multi scenes scenario, we observe that ImPosing is slightly better than single scene models, whereas accuracy of competitors decreases from single scene to multi scenes. On 7scenes, ImPosing outperforms Image Retrieval but present a higher error compared to pose regression, especially on rotation.

We believe that our candidates sampling method is not well suited for such smaller scenes where the set of camera orientations is more diverse than in large scale driving scenarios where our method compares favorably. These experiments confirm that ImPosing is compatible with multi scene scenarios,

Table 6.3: Small-scale datasets (median localization error in meters/degrees).

| | Dataset | Image retrieval | | Camera Pose Regression | | | | ImPosing | |
|---|---|---|---|---|---|---|---|---|---|
| | | NetVLAD | AP-GeM | PoseNet | | Transformer | | | |
| | | | | Single sc. | Multi sc. | Single sc. | Multi sc. | Single sc. | Multi sc. |
| Cambridge | K College | 2.94/6.23 | **0.70/0.9** | 0.99/1.1 | 1.73/3.6 | 0.60/2.4 | 0.83/1.5 | 1.71/3.6 | 1.58/3.5 |
| | OldHosp | 4.87/9.2 | **0.97/2.1** | 2.17/2.9 | 2.55/4.1 | 1.45/3.1 | 1.81/2.4 | 2.45/5.1 | 2.39/5.0 |
| | Shop | 1.32/7.8 | 0.69/3.7 | 1.05/4.0 | 2.02/7.5 | **0.55/3.5** | 0.86/3.1 | 0.97/7.0 | 0.93/6.9 |
| | Church | 3.71/11.1 | 1.32/3.6 | 1.49/**3.4** | 2.67/6.2 | **1.09**/5.0 | 1.62/4.0 | 1.51/6.6 | 1.60/7.0 |
| | Average | 3.21/8.6 | **0.92/2.6** | 1.43/2.9 | 2.24/5.4 | **0.92**/3.5 | 1.28/2.8 | 1.66/5.6 | 1.63/5.6 |
| 7scenes | Chess | 0.24/10.4 | 0.21/11.0 | 0.14/**4.5** | 0.09/4.8 | **0.08**/5.7 | 0.11/4.7 | 0.17/8.5 | 0.17/9.6 |
| | Fire | 0.33/14.0 | 0.33/15.1 | 0.27/11.8 | 0.29/10.5 | **0.24**/10.6 | **0.24/9.6** | 0.28/11.3 | 0.29/12.6 |
| | Heads | 0.18/16.4 | 0.17/16.7 | 0.18/**12.1** | 0.16/13.1 | 0.13/12.7 | 0.14/12.2 | **0.13**/13.0 | **0.13**/13.8 |
| | Office | 0.30/11.1 | 0.29/12.0 | 0.20/5.8 | **0.16**/6.8 | 0.17/6.3 | 0.17/**5.7** | 0.25/9.0 | 0.25/9.1 |
| | Pumpkin | 0.38/11.2 | 0.37/11.0 | 0.25/4.8 | 0.19/5.5 | 0.17/5.6 | 0.18/4.4 | 0.27/10.4 | 0.29/9.6 |
| | Kitchen | 0.34/12.3 | 0.36/12.2 | 0.24/**5.5** | 0.21/6.6 | 0.19/6.8 | **0.17/5.9** | 0.28/8.0 | 0.26/8.2 |
| | Stairs | 0.28/13.8 | 0.31/14.8 | 0.37/10.6 | 0.31/11.6 | 0.30/**7.0** | **0.26**/8.4 | 0.27/11.8 | **0.26**/11.3 |
| | Average | 0.29/12.7 | 0.29/13.3 | 0.24/7.9 | 0.20/8.4 | **0.18**/7.8 | **0.18/7.3** | 0.24/10.3 | 0.24/10.6 |

which is an important property for large scale deployment of localization systems.

### 6.4.5 Efficiency comparison

**Storage footprint.** Our method only needs to store neural networks weights and initial pose candidates in device. It represents 23MB for the image encoder, less than 1MB for the pose encoder and 1MB for the initial poses candidates. We also report in figure 6.6 the scaling law of memory footprint w.r.t. reference database size for different classes of visual localization methods. This is an important aspect in autonomous driving scenarios where large amounts of data are available. For a given map, learning-based methods have a constant memory requirement because the map information is embedded in the networks weights. To estimate storage requirement of retrieval methods, we consider the size of the database image descriptor (2048 for GeM and 4096 for NetVLAD) along with the size of the image encoder. Storage requirement of retrieval methods exceed 1 GB for large scale scene with more than 100k reference images. To estimate the memory requirement of structure-based methods we consider the numbers given in [Sarlin et al., 2019]: a 3D model built from 4328 images is composed of 685k 3D points. If we consider one local descriptor of size 128 by 3D points, we can derive a linear rule to

Table 6.4: **Qualitative comparison between methods.** We compare the properties of visual localization class of methods w.r.t. storage requirement, capability to operate in large maps (scalability), latency and accuracy. *IR* stands for Image Retrieval, *PR* for Pose Regression, *SCR* for Scene Coordinate Regression, *DB* for database and *NN* for neural networks weights. Storage of IR databases are detailed in [Song et al., 2022].

| **Algorithms** | **In device storage** | **Scalability** | **Latency** | **Accuracy** |
|---|---|---|---|---|
| IR+2D-3D matching | 3D model + IR DB + NN (5-100GBs) | High | Low | High |
| IR+Relative PR | IR DB with images + NN (5-100GBs) | High | Low | Medium |
| IR | IR DB + NN (2-50GBs) | High | Medium | Low |
| APR | NN ($\approx$ 25MB) | Medium | High | Low |
| SCR | NN ($\approx$ 25MB) | Low | High | High |
| **ImPosing (ours)** | NN (25MB) | High | High | Medium |

determine the 3D model size according to the number of reference images. This is a rough estimation but we can estimate that structure based method require at least 3 times more storage capacity than image retrieval methods. Compressing techniques exist to make these methods more tractable [Sattler et al., 2015, Camposeco et al., 2019], however compressed maps still represent gigabytes and are less accurate.

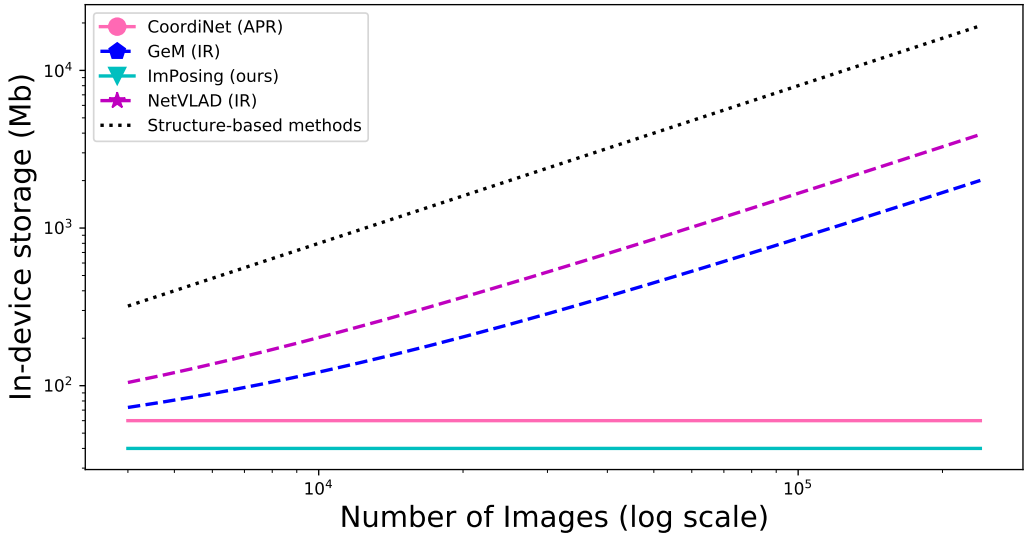


Figure 6.6: **In-device memory usage.** Structure-based methods (black) and image retrieval (blue and purple) use more memory when the reference dataset grows whereas pose regression methods and ImPosing (pink and cyan) storage requirement does not depend on dataset size.

**Computational complexity.** Our algorithm complexity depends on the image encoder backbone (3.6 billion FLOPs for ResNet34) and the hierarchical decoding process with the pose encoder. With the default hyperparameters, it involves 4.8 billion FLOPs. We measured a total inference time of 41ms for a single image using a NVIDIA RTX 2080 GPU. The complexity is linear w.r.t. the number of refinements $K$, the number of pose candidates $N$ and the number of layers in the MLP. It is quadratic w.r.t. the latent dimension $D$. It should be noted that parallel computations reduce the impact of $N$ and $D$ on the inference time. Considering these properties and the ablations provided in 6.4.6, one can choose the corresponding hyperparameters that match its computational requirements.

**Summary.** ImPosing exhibits very compact storage requirements and fast inference time coupled with state-of-the-art accuracy. Notably, neither memory footprint and computational complexity depends on the number of images in the reference database, which is a great advantage over image retrieval methods [Arandjelović et al., 2016, Revaud et al., 2019]. We also observe empirically that our method converges approximately 2 times faster than pose regression competitors [Moreau et al., 2022a] w.r.t. the number of training iterations (see figure 6.7).

### 6.4.6 Ablation study

We report the influence of several hyperparameters on the localization accuracy of ImPosing in figure 6.7. We evaluate the number of refinement steps $K$, the number of pose candidates $N$ and the number of best candidates used for pose averaging. We use the model trained on Daoxiang Lake and change the

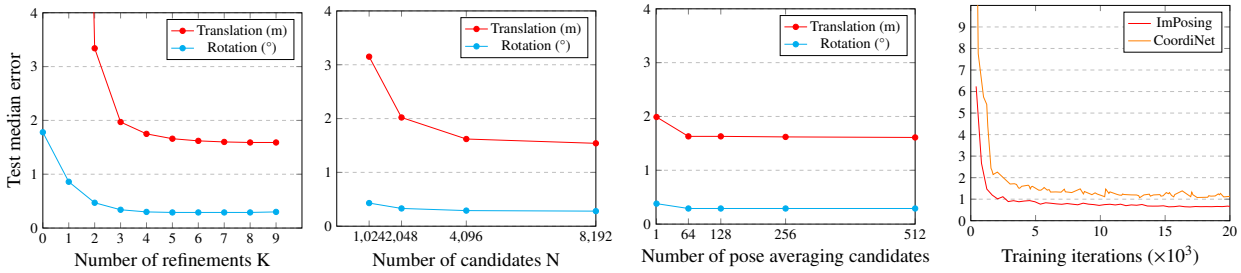


Figure 6.7: **From left to right:** median localization errors depending on number of refinements, pose candidates, and final number averaged poses. Training time comparison between pose regression [Moreau et al., 2022a] and ImPosing.

parameters at test time. Increasing the number of refinements and candidates improves localization accuracy, at the cost of a higher computational cost. We use a reasonable trade-off with $K = 6$ and $N = 4096$ as our default setup. We observe that pose averaging has a positive impact on accuracy, but the number of selected candidates is not critical.

### 6.4.7 Qualitative analysis

We propose a video showing various qualitative examples of our localization algorithm in different scenes. The video is accessible at the following address : `https://youtu.be/tHHQYdY1xDM`

The input image is displayed on the top left corner. The right part shows the current predicted trajectory in red, ground truth poses in green and training trajectories in gray. The bottom left corner displays the 256 best candidates selected for pose averaging in red, the predicted pose in black and the groundtruth pose in green. Finally, the last plot shows the score of all candidates in the entire map from transparent ($s = 0$) to red ($s = 1$).

Scenes are displayed in the following order : Daoxiang Lake (00:00 to 01:03), Neighborhood (01:04 to 02:09), Office Loop (02:10 to 02:40), Business campus (02:41 to 03:38), City Loop (03:39 to 03:59), Countryside (04:00 to 04:44), Old Town (04:45 to 05:00).

These video samples show clearly advantages and limitations of our method:

- Coarse localization is correct most of the time, even in large maps with repetitive and featureless environments (see figure 6.4).
- In ambiguous scenarios, our method provides a multimodal distribution of scores in first iterations and then solves the ambiguity in further steps (see figure 6.8).
- Sequences of predictions are not temporally smooth, because each frame is treated independently in this experiment. In practice, this can be solved by filtering with a motion model (see section 3.6.3).
- Precise pose estimation is sometimes inaccurate but sufficient to provide a lane level localization for navigation of autonomous vehicles.

It should also be noted that experiments on 4seasons dataset are extreme scenarios where the quantity of available data is small w.r.t. to the challenges introduced by weather conditions.

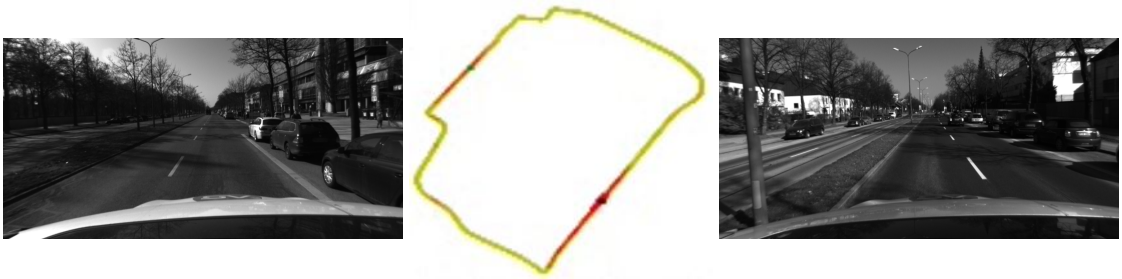

Figure 6.8: **Multimodal score distribution in ambiguous cases.** Many road areas present similar structure and appearance, introducing ambiguities in the localization task. In this scenario from the City Loop scene, the model outputs high scores for areas depicted in left and right images, which are very far one from each other. By refining the estimate in further steps, the model is able to solve this ambiguity in most cases.

## 6.5 Discussion

**What does the pose encoder learn?** In the pose regression approach, image and camera pose are connected by being the respective input and output of a single feedforward neural network. This formulation entangles features extraction, map memorization and camera pose prediction in a single model. While deep neural networks are known to perform well for the first, they have been observed to be inaccurate for pose prediction [Sattler et al., 2019]. Our solution circumvents this problem by "inverting" the decoder layers with the pose encoder. We don't try to predict the pose from features but to connect a given pose to its respective latent features. We let the network learn the optimal latent space to connect images and camera poses, with a single constraint: pose candidates close to the actual camera pose must have a vector relatively similar to the image representation. This property enables to search the best pose candidates in a coarse to fine manner, and interpret the resulting scores has a multimodal distribution of positions across the map.

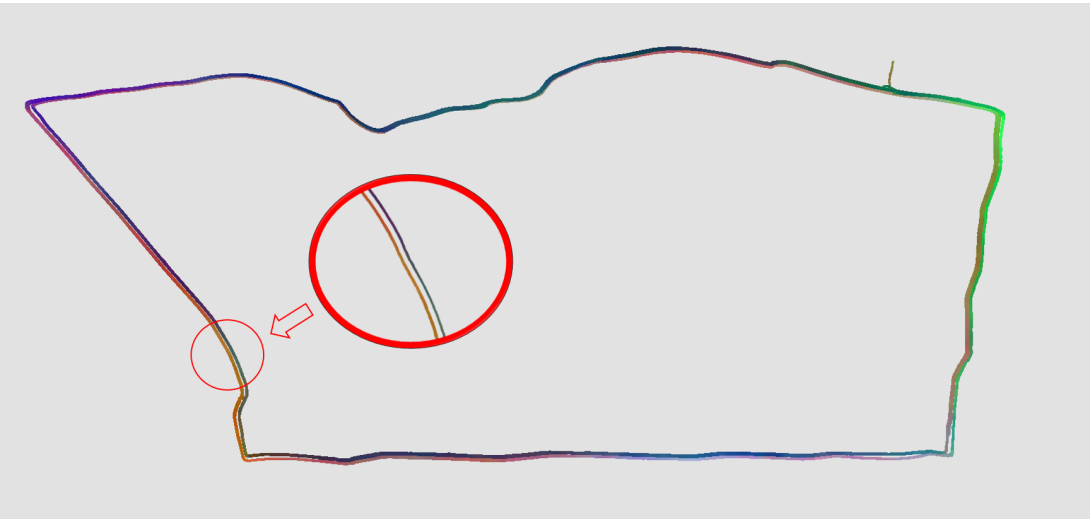

Figure 6.9: **Latent space visualization.** Training poses, colored by the 3 principal components of map descriptors. Poses with similar colors are close in the latent space. Opposite ways of the same road are represented by dissimilar representations. Best viewed in color

**Benefits, limitations and future work.** Our method keeps the main advantages of direct learning-

based methods: we obtain the pose efficiently with neural networks inference, we do not use a 3D model of the scene or a retrieval database, resulting in a very compact memory footprint. We observe that the accuracy our method highly depends on the quantity of training data available. Similar to regression, our method does not extrapolate to camera positions far from trainings examples. However, recent approaches has shown that these limitations can be overcome with synthetic datasets [Moreau et al., 2022b]. Moreover, in the driving scenario, a coarse localization estimate can be sufficient because horizontal localization (road lane) can be recovered thanks to perception [Qin et al., 2021]. The new paradigm we propose could be improved in many ways. It includes exploring better architectures for the pose encoder, inspired from recent work on coordinate-based representations [Zhu et al., 2021]. Another interesting direction is to extend the implicit map representation to local features instead of global image signatures, by finding a way to represent implicitly a 3D model. We explore this last direction in chapter 7.

## 6.6 Conclusion

We have proposed a new formulation for visual localization that perform state-of-the art accuracy for direct learning-based methods in large environments. By using an implicit representation of the map, we connect camera poses and image features in a latent high dimensional manifold well suited for localization. We have shown that with a simple pose candidates sampling procedure, we are able to estimate the absolute pose of an image. Our proposal can be directly applied in autonomous driving systems, by providing an efficient and accurate image-based localization algorithm that can operate at large scales in real-time.

We believe that, beyond our work, implicit scene representations, by their ability to model complex continuous signals in a fixed size neural network, are a promising research direction for camera pose estimation. Next chapter will extend this idea to another class of visual localization algorithms.

Chapter 7

# Self-supervised localization features in a radiance field

## Contents


**7.1 Résumé en Français** . . . . . **86**

**7.2 Motivation** . . . . . **87**

**7.3 Method** . . . . . **87**

7.3.1 Neural rendering of positional descriptors . . . . . 87

7.3.2 Self-supervised training of features in CROSSFIRE . . . . . 88

7.3.3 Visual Localization by iterative dense features matching . . . . . 91

**7.4 Experiments** . . . . . **92**

7.4.1 Comparison to related methods . . . . . 92

7.4.2 How good the pose priors need to be ? . . . . . 94

7.4.3 Ablation studies . . . . . 95

7.4.4 Efficiency . . . . . 97

**7.5 Limitations and Future Work** . . . . . **97**

**7.6 Conclusion** . . . . . **97**

## 7.1 Résumé en Français

Ce chapitre étend l'idée d'utiliser des répresentations de cartes implicites aux méthodes basées sur des correspondances locales, qui permettent une précision plus élevée. Classiquement ces méthodes fonctionnent en associant des pixels de l'image de requête avec des points 3D d'un nuage de points de la scène, ce qui permet d'estimer la position de la caméra avec des principes de géométrie projective.

Ici, nous remplacons le modèle 3D explicite par un Neural Radiance Field (NeRF), qui fournit des descripteurs locaux pour n'importe quel point observé dans la scène. Cette représentation de la scène présente plusieurs avantages tels que la compacité, la capacité d'effectuer d'établir des associations avec n'importe quel point et de raffiner l'estimation de la position de la caméra un nombre arbitraire de fois.

Nous décrivons notre système CROSSFIRE, comment l'entraîner de manière auto-supervisée et nous évaluons ses propriétés dans des environnements intérieurs et extérieurs.

## 7.2 Motivation

The algorithm presented in chapter 6 operates with global image descriptors. It enables efficient search in a large map but does not provide a very precise camera pose in smaller environments. To obtain a centimeter-level localization, the common solution consists in extracting local image features and match them against a 3D model (see section 2.3).

In this chapter, we extend the idea of implicit map representation to structure-based methods. The commonly used sparse 3D model is replaced by a neural field, able to render consistent local descriptors from any camera viewpoint.

We propose to introduce local descriptors in the NeRF implicit formulation and to use the resulting model as the scene representation of a 2D-3D features matching method. We train simultaneously a CNN feature extractor and a neural renderer to provide consistent scene-specific descriptors in a self-supervised way. During training, we leverage the 3D information learned by the radiance field in a metric learning optimization objective which does not require supervised pixel correspondences on image pairs or a pre-computed 3D model. The proposed "positional descriptors" represent not only the local 2D image content but also the 3D position of the observed point, which enables to solve ambiguities in areas with repetitive patterns. Our method can use any differentiable neural renderer and then can directly benefit from recent NeRF improvements. For instance, we make the model computationally tractable thanks to the multi-resolution hash encoding from Instant-NGP [Müller et al., 2022] and adapted to dynamic outdoor scenes thanks to appearance embeddings from Nerf-W [Martin-Brualla et al., 2021].

Finally, we show that these features can be used to solve the visual relocalization task with an iterative algorithm composed of a dense features matching step followed by standard Perspective-n-Points (PnP) camera pose computation. In our novel relocalization paradigm, named CROSSFIRE, the commonly used sparse 3D model obtained from Structure-from-Motion is replaced by a Neural Field. For a given camera pose candidate, we use the neural field to render dense descriptors and depth maps. Descriptors are used to establish 2D-2D matches which are upgraded to 2D-3D matches by the rendered depth. We can iteratively refine the estimated pose by repeating the aforementioned localization procedure.

We leverage the 3D information learned by the NeRF at multiple stages: we render depth maps used to obtain 2D-3D matches during localization, we do not need pairs of images with annotated correspondences for training because repeatability is naturally learned in the neural field, and our optimization objective depends on the 3D distance between pixels in order to learn a smooth descriptor field.

## 7.3 Method

Our algorithm estimates the 6-DoF camera pose of a query image in an already visited environment. We first train our modules in an offline step, using a set of reference images with corresponding poses, captured beforehand in the area of interest. A 3D model of the scene is not a pre-requisite because we learn the scene geometry during the training process.

### 7.3.1 Neural rendering of positional descriptors

Our neural renderer combines the original NeRF formulation with the multi-resolution hash encoding of Instant-NGP [Müller et al., 2022] and the appearance embeddings of Nerf-W [Martin-Brualla et al.,

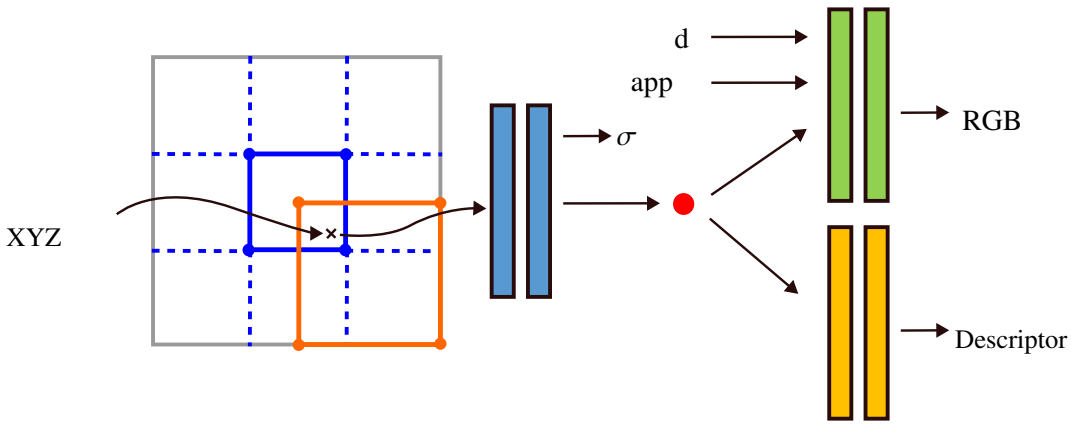


Figure 7.1: **Neural Positional Feature Fields architecture.** We build upon the architecture of Instant-NGP [Müller et al., 2022] enabling fast training and rendering. We use per-image appearance embeddings during training to handle varying illumination across training images. For accurate localization, we add an MLP that is invariant to viewing direction and appearance vector allowing to learn robust localization feature vectors.

2021] to efficiently render dynamic scenes. However, our main objective is not photorealistic rendering but, rather, features matching with new observations. While it is possible to align a query image with a NeRF model by minimizing the photometric error [Yen-Chen et al., 2021], such approach lacks robustness w.r.t. variations in illumination. Instead, we propose to add positional features, i.e. $D$-dimensional latent vectors which describe the visual content of a region of interest in the scene, as an additional output of the radiance field function. In contrast with the rendered color, we model these descriptors as invariant to viewing direction $d$ and appearance vector $\mathcal{L}_i^{(a)}$ (*i.e.* we do not provide $d$ and $\mathcal{L}_i^{(a)}$ to the MLP head responsible of generating the positional feature, see Figure 7.1). We verify through ablation study in section 7.4.3 that this descriptor property makes the matching process more robust. Similar to color, the 2D descriptor of a camera ray is aggregated by the usual volumetric rendering formula applied on descriptors of each point along the ray. The architecture of our proposed neural renderer is summarized in Figure 7.1 and implementations details are provided in section. The training pipeline of CROSSFIRE is explained in the next section.

### 7.3.2 Self-supervised training of features in CROSSFIRE

**Motivation:** In the previous section, we explained how our proposed neural renderer describes the map for relocalization purposes thanks to the introduced positional descriptors. Additionally, we also need to extract features from the query image. A simple solution, proposed by FQN [Germain et al., 2022], is to use an off-the-shelf pre-trained features extractor such as SuperPoint [DeTone et al., 2018] or D2-Net [Dusmanu et al., 2019], and train the neural renderer to memorize observed descriptors depending on the viewing direction. Optimizing scene-specific descriptors, however, allows to better differentiate repetitive patterns in the scene resulting, in a more robust localization and reducing failure cases. To this end, we propose to train jointly the feature extractor with the neural renderer by defining an optimization objective which leverages the scene geometry. We obtain descriptors specialized on the target scene which describe not only the visual content but also the 3D location of the observed point, with better discriminant property than generic descriptors.

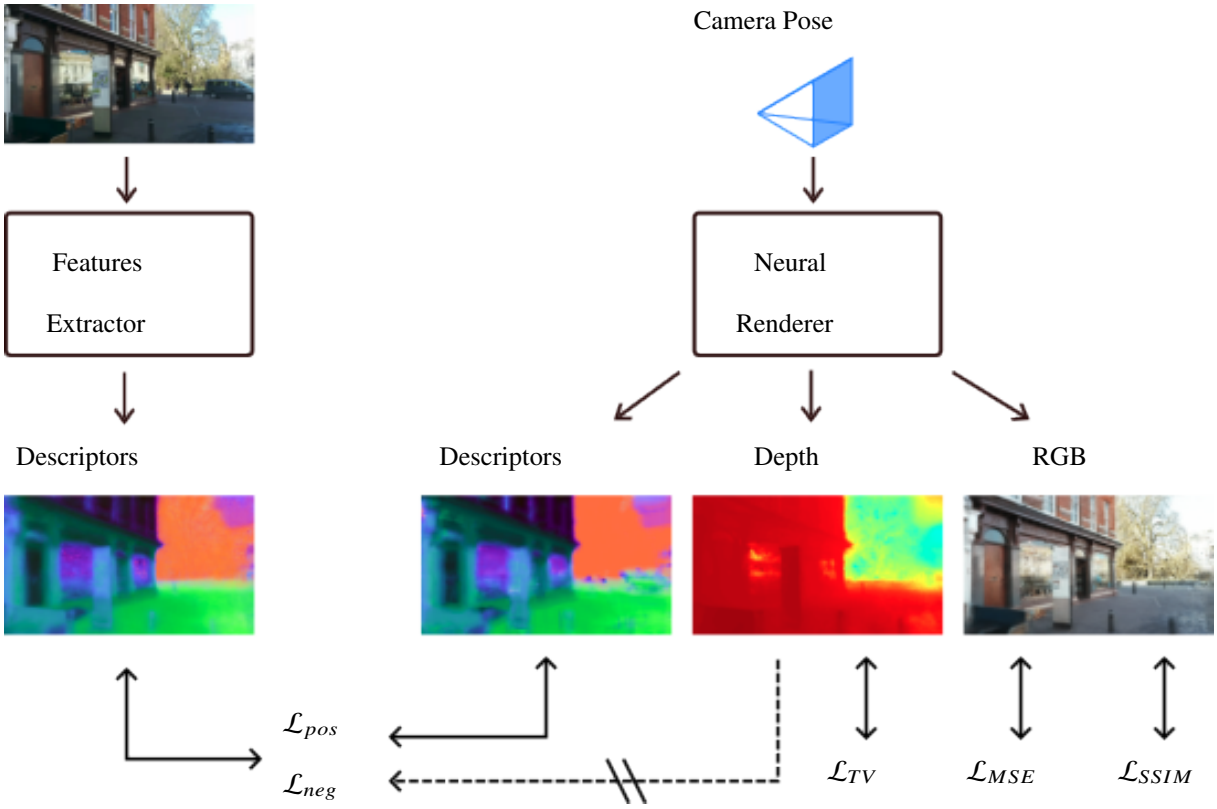


Figure 7.2: **Training pipeline of Neural Positional Features Fields.** We jointly optimize the neural renderer and the features extractor to obtain robust, scene-specific localization descriptors. We use regularization losses (i.e. TV and SSIM) to increase the consistency of the neural renderer. We propose a two-terms loss that maximizes the similarity between corresponding feature maps while penalizing pixel pairs that are geometrically distant from each other.

The training procedure of our system is described in Figure 7.2. One training sample corresponds to a reference image with its corresponding camera pose. From one side, the image is processed by the features extractor to obtain the descriptors map $F_I$. On the other side, we sample points along rays for each pixel using camera intrinsics, compute density, color and descriptor of each 3D point, and finally perform volumetric rendering to obtain a RGB view $C_R$, a descriptors map $F_R$ and a depth map $D_R$.

**Features Extraction:** Our features extractor is a simple fully convolutional neural network with 8 layers, ReLU activations and max poolings. The input is a RGB image $I$ of size $H \times W$ and produces a dense descriptors map $F_I \in \mathbb{R}^{H/4 \times W/4 \times d}$.

**Learning the Radiance Field:** Similar to NeRF [Mildenhall et al., 2020], we use the mean squared error loss $\mathcal{L}_{MSE}$ between $C_R$ and the real image to learn the radiance field. As we render entire, although downscaled, images in a single training step, we can leverage the local 2D image structure and minimise the structural dissimilarity (DSSIM) loss $\mathcal{L}_{SSIM}$ [Wang et al., 2004]. We observe that it produces sharper images and better results. Depth maps are used by the localization process to compute the camera pose, and then better depth results in more accurate poses. NeRF models trained with limited training views can yield incorrect depths, due to the shape-radiance ambiguity [Zhang et al., 2020a]. We add a regularization loss $\mathcal{L}_{TV}$ which minimizes depth total variation of randomly sampled 5x5 image patches to encourage smoothness and limit artefacts on the rendered depth maps [Niemeyer et al., 2022].

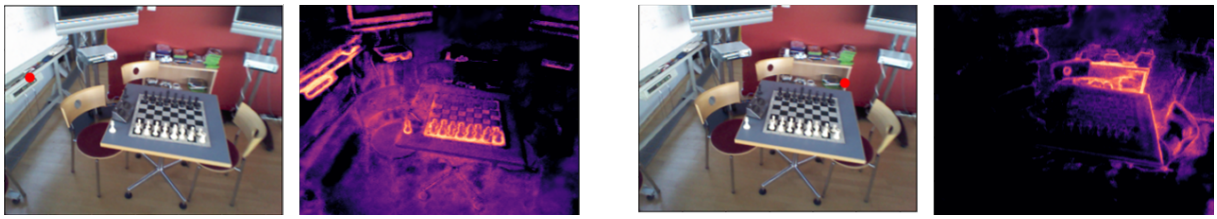

Figure 7.3: **Similarities of positional features.** We show the dense matching map between one descriptor from the query image (red dots in left images) and the reference descriptors from the neural renderer. Thanks to our training objective, descriptors close (in 3D) to the selected points have high similary whereas others do not match. This behaviour is enforced by our loss function.

**Learning the Descriptors Field**: Our main goal is to match the descriptors map from the CNN features extractor and the corresponding one from the neural renderer. The self-supervised optimization objective encourages both models to produce identical features for a given pixel while preventing high matching scores between points far from each other in the 3D scene. We define a loss function with two terms $\mathcal{L}_{pos}$ and $\mathcal{L}_{neg}$, applied on a pair of descriptors maps, each containing $n$ pixels. We use the cosine similarity, noted $\otimes$, to measure similarity between descriptors.

The first loss term $\mathcal{L}_{pos}$ maximizes the similarity between descriptors maps $F_I$ and $F_R$ from both models:

$$\mathcal{L}_{pos} = \frac{1}{n}\sum_{i=1}^{n} \max(0, 1 - F_I[i] \otimes F_R[i]) \tag{7.1}$$

The second loss term $\mathcal{L}_{neg}$ samples random pairs of pixels and ensures that pixel pairs with large 3D distances have dissimilar descriptors:

$$\mathcal{L}_{neg} = \frac{1}{mn}\sum_{k,i=1}^{m,n} \max(0, F_I[p_k(i)] \otimes F_R[i] - t_\lambda(p_k(i), i)) \tag{7.2}$$

where $t_\lambda(i,j) = max(0, 1 - \lambda\|xyz(i) - xyz(j)\|)$. $xyz(i)$ is the 3D coordinate of the point represented by the $i_{th}$ pixel in the descriptors map. We compute it from the camera parameters of the rendered view and predicted depth. It should be noted that we do not backpropagate the gradient of this loss to the depth map because the gradient of this loss does not provide meaningful signal to learn the scene geometry. $\lambda$ is an hyperparameter which controls the maximum similarity between descriptors at a given 3D distance. $(p_k)_m$ are random permutations of pixel indices from 1 to n.

The proposed self-supervised objective is close to a classical triplet loss [Arandjelović et al., 2016], but we show in fig 7.8 that scaling the loss by the 3D coordinates in the formulation is crucial to learn smooth and selective descriptors. A visualization of the similarity between descriptors enforced by the proposed loss is shown in Fig 7.3.

Finally, we optimize the following loss function at each training step:

$$\mathcal{L} = \mathcal{L}_{MSE} + \lambda_1 \mathcal{L}_{SSIM} + \lambda_2 \mathcal{L}_{TV} + \mathcal{L}_{pos} + \mathcal{L}_{neg} \tag{7.3}$$

where $\lambda_1 = 0.1$ and $\lambda_2 = 1e^{-3}$ are hyper-parameters introduced to balance SSIM and TV losses, respectively.

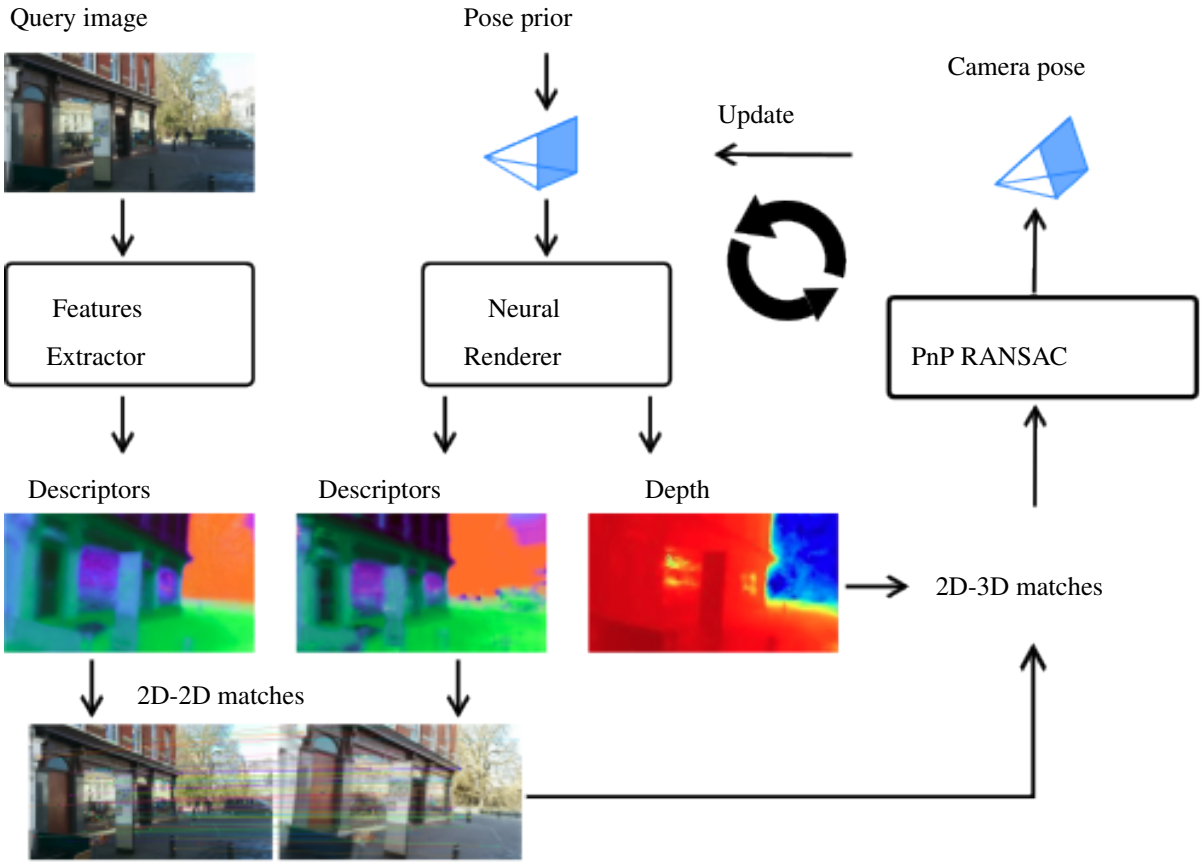


Figure 7.4: **Localization procedure.** Descriptors are extracted from the query image and matched against descriptors rendered from the localization prior. Depth information provides 2D-3D matches that enable to compute the pose with PnP + RANSAC. This process can be repeated iteratively, by rendering descriptors from the predicted pose.

### 7.3.3 Visual Localization by iterative dense features matching

This section describes the localization pipeline used to estimate the camera pose of a given query image using our learned renderer and features. An overview of this procedure is shown in Figure 7.4. The proposed solution combines simple and commonly used techniques and we do not claim any algorithmic novelty on this part. The goal is, rather, to demonstrate that the high quality and robustness of our learned features enables to reach precise localization while using basic features matching and pose estimation strategies.

**1. Localization prior** Similar to related features matching methods [Sattler et al., 2012, Sarlin et al., 2019, Germain et al., 2022], we assume to have access to a localization prior, i.e. a camera pose relatively close to the query pose. A view observed from the prior should have an overlapping visual content with the query image to make the matching process feasible. Such priors can be obtained by matching a global image descriptor against an image retrieval database [Arandjelović et al., 2016, Sarlin et al., 2019] or an implicit map [Moreau et al., 2023a].

**2. Features extraction** First, we extract dense descriptors from the query image through the CNN and descriptors and depth from the localization prior with the neural renderer.

**3. Dense Features Matching** Query and reference descriptors are matched with cosine similarity. We consider that 2 descriptors are a match if the similarity is higher than a threshold $\theta$ and if it represent the best candidate in the other map in both direction (mutual matching). We then compute the predicted 3D coordinate of rendered pixels which have been matched (thanks to camera parameters and depth) and obtain a set of 2D-3D matches.

**4. Camera Pose Estimation** We use the Perspective-N-Points algorithm combined with RANSAC [Fischler and Bolles, 1981], in order to get a robust estimate by discarding outliers matches.

**5. Iterative Pose Refinement** While classical 3D models only have access to a finite set of reference descriptors, our neural renderer can compute them from any camera pose. Similar to FQN [Germain et al., 2022] and ImPosing [Moreau et al., 2023a], we can then consider the camera pose estimate as a new localization prior and iterate the previously mentioned steps multiple times to refine the camera pose.

## 7.4 Experiments

This section evaluates the proposed model in several ways. In section 7.4.1, we compare CROSSFIRE to other NeRF-related methods in the literature. In section 7.4.2, we study the impact of the pose prior on the accuracy. Ablations studies on several components and parameters of the method are conducted in section 7.4.3. Finally, qualitative examples and analysis are provided in Figure 7.5 and Figure 7.7.

**Implementation:** Our system is implemented in PyTorch. The hash tables and MLPs of the neural renderer use tiny-cuda-nn [Müller, 2021]. We use the default PnP pose solver from PoseLib [Larsson, 2020]. In all the proposed experiments, we use descriptors of size 32. We train the models for 100k iterations. The initial learning rate is set to $1e^{-3}$ and reduced to $1e^{-4}$ after 2000 iterations. All trainings are performed on a RTX3090 GPU and take approximately 15 hours.

**Datasets:** We evaluate our method on 2 standard localization benchmarks. 7scenes [Shotton et al., 2013] consists in indoor static scenes captured using a hand-held camera. Cambridge Landmarks [Kendall et al., 2015] contains outdoor scenes representing buildings observed from different viewpoints and lighting conditions, with dynamic occluders such as pedestrians and cyclists in both train and test sets.

**Metrics:** We report commonly used metrics for visual localization, i.e. median translation and orientation errors. These metrics have the advantage to not be penalized by outliers.

### 7.4.1 Comparison to related methods

We evaluate our method on both datasets using a maximum of 3 iterations of the localization process. We use as localization prior the top 1 reference pose retrieved by DenseVLAD [Torii et al., 2015b]. In order to render reference frames efficiently, the matching step is done at a small resolution: 194x108 for Cambridge Landmarks and 161x120 for 7scenes.

We compare our algorithm to the most related and best performing recent learning-based visual relocalization methods, with a focus on methods using neural scenes representations in their pipeline.

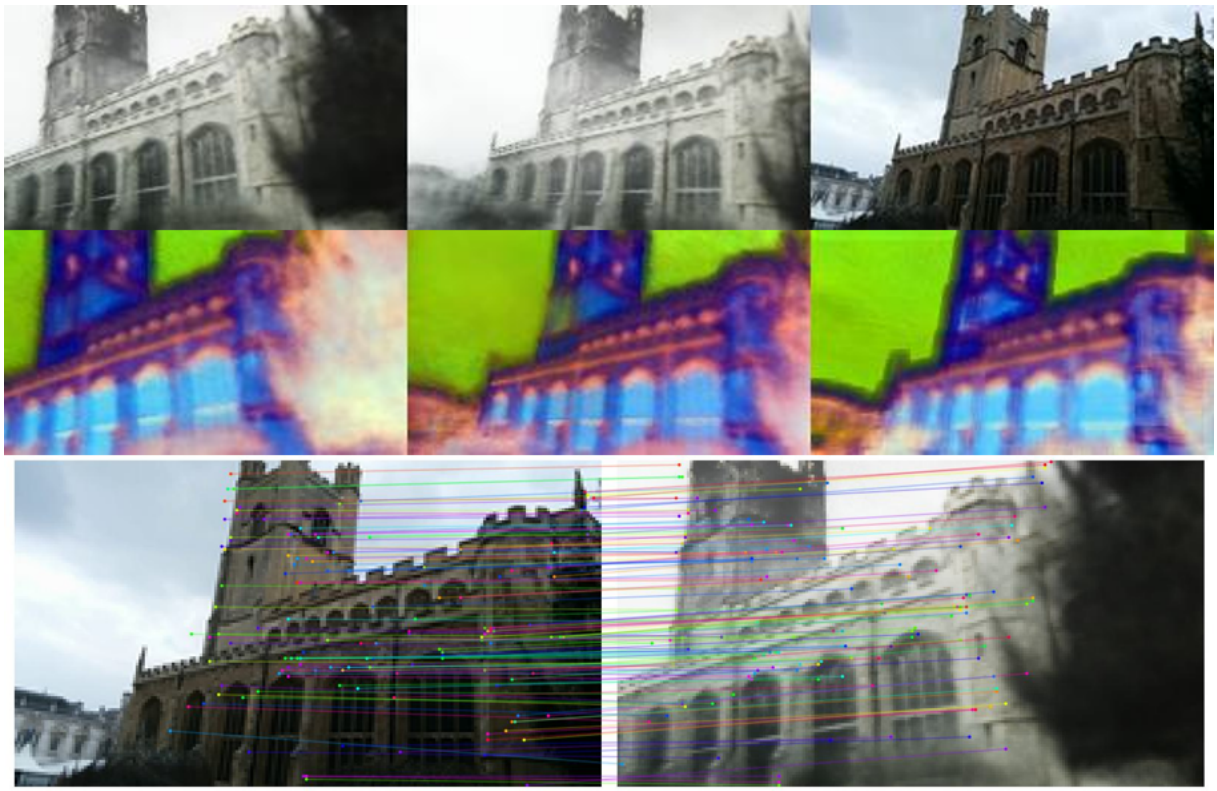

Figure 7.5: **Visualization of rendered views, descriptors and matches in StMarysChurch.** We show on the top row the query image (right), the RGB rendered view from the localization prior (left) and from the 1st estimated pose (middle). The second row represents a PCA visualization of the corresponding descriptors map from the neural renderer (left and middle) and the features extractor (right). The last row displays the inlier matches obtained by our pipeline.

- Direct-PoseNet [Chen et al., 2021a] trains an Absolute Pose Regressor with an additional photometric loss by rendering the estimated pose through NeRF.
- DFNet [Chen et al., 2022b] goes in the same direction but defines a features matching loss with the rendered view.
- LENS [Moreau et al., 2022b] trains an absolute pose regressor with NeRF rendered views uniformly distributed across the scene.
- FQN [Germain et al., 2022] regresses descriptors in an implicit representation of a sparse 3D model. This method is the closest to our work because it uses the same iterative localization process and store descriptors in a neural scene representation. The main differences are that descriptors are not trained specifically from the scene but memorized from a pretrained features extractors, and that the representation is sparse whereas ours is dense. Results are reported for D2-Net [Dusmanu et al., 2019] and MobileNetv2 [Sandler et al., 2018] descriptors.
- DSAC++ [Brachmann and Rother, 2018] regresses directly 2D-3D matches, learns through a differentiable RANSAC. Similar to our method, the CNN is scene-specific, but we replace predicted 3D coordinates by latent vectors matched against the NeRF. We report the results without a 3D model supervision because our method has the same input.

| Dataset / Methods | Absolute Pose Regression + NeRF | | | Implicit descriptors | | | SCR |
|---|---|---|---|---|---|---|---|
| Cambridge | DirectPN | DFNet | LENS | FQN-D2N | FQN-MN | **NPFF (Ours)** | DSAC++ |
| Kings College | - | 0.73m / 2.4° | 0.33m / 0.5° | 0.32m / 0.5° | 0.28m / 0.4° | 0.47m / 0.7° | **0.23m / 0.4°** |
| Old Hospital | - | 2.00m / 3.0° | 0.44m / 0.9° | 0.64m / 0.9° | 0.54m / 0.8° | 0.43m / 0.7° | **0.24m / 0.5°** |
| Shop Facade | - | 0.67m / 2.2° | 0.27m / 1.6° | 0.14m / 0.6° | 0.13m / 0.6° | 0.20m / 1.2° | **0.09m / 0.4°** |
| StMarys Church | - | 1.37m / 4.0° | 0.53m / 1.6° | 0.93m / 3.5° | 0.58m / 2.0° | 0.39m / 1.4° | **0.20m / 0.7°** |
| Average | - | 1.19m / 2.9° | 0.39m / 1.2° | 0.51m / 1.4° | 0.38m / 1.0° | 0.37m / 1.0° | **0.19m / 0.5°** |
| 7scenes | | | | | | | |
| Chess | 0.10m / 3.5° | 0.05m / 1.9° | 0.03m / 1.3° | 0.06m / 1.9° | 0.04m / 1.3° | **0.01m / 0.4°** | 0.02m / 0.7° |
| Fire | 0.27m / 11.7° | 0.17m / 6.5° | 0.10m / 3.7° | 0.14m / 4.1° | 0.10m / 3.0° | 0.05m / 1.9° | **0.03m / 1.1°** |
| Heads | 0.17m / 13.1° | 0.06m / 3.6° | 0.07m / 5.8° | 0.05m / 3.5° | 0.04m / 2.4° | **0.03m / 2.3°** | 0.12m / 6.7° |
| Office | 0.16m / 6.0° | 0.08m / 2.5° | 0.07m / 1.9° | 0.14m / 4.1° | 0.10m / 3.0° | 0.05m / 1.6° | **0.03m / 0.8°** |
| Pumpkin | 0.19m / 3.9° | 0.10m / 2.8° | 0.08m / 2.2° | 0.10m / 2.6° | 0.09m / 2.4° | **0.03m / 0.8°** | 0.05m / 1.1° |
| Kitchen | 0.22m / 5.1° | 0.22m / 5.5° | 0.09m / 2.2° | 0.18m / 4.8° | 0.16m / 4.4° | **0.02m / 0.8°** | 0.05m / 1.3° |
| Stairs | 0.32m / 10.6° | 0.16m / 3.3° | 0.14m / 3.6° | 1.41m / 53.0° | 1.40m / 34.7° | **0.12m / 1.9°** | 0.29m / 5.1° |
| Average | 0.20m / 7.3° | 0.12m / 3.7° | 0.08m / 3.0° | 0.30m / 10.6° | 0.28m / 7.3° | **0.04m / 1.1°** | 0.08m / 2.4° |

Table 7.1: **6-DoF median localization errors of learning-based visual localization methods.** NPFF is the best performing method on indoor scenes and second best in outdoor.

The results of the comparisons for both datasets are shown in Table 7.1. Our methods obtains the lowest error for indoor localization and the second best for outdoor scenes. In 7scenes dataset, the median localization error is 2 times better than the best competitor DSAC++. Results on the highly ambiguous Stairs scene are higher than in other scenes but still better than other methods for which the localization process sometimes totally fail.

Furthermore, we consistently perform better than NeRF-assisted APR methods and, more importantly, than pretrained implicit descriptors. Because the camera pose estimation process used in FQN is similar than in ours, these results indicate that our scene-specific features are beneficial compared to off-the-self features extractors.

We hypothesize that our method does not outperform scene coordinate regression in outdoor scenes for 2 main reasons. First, we lack a way to handle dynamic content such as pedestrians during the test step, which we observe to degrade the quality of our matches. Second, the quality of depth maps in these scenes is less accurate than in indoor scenarios, especially for background, due to observable image content very far from the camera. As we use depth to compute the 3D coordinates of matches, this introduces noise in the localization process.

### 7.4.2 How good the pose priors need to be ?

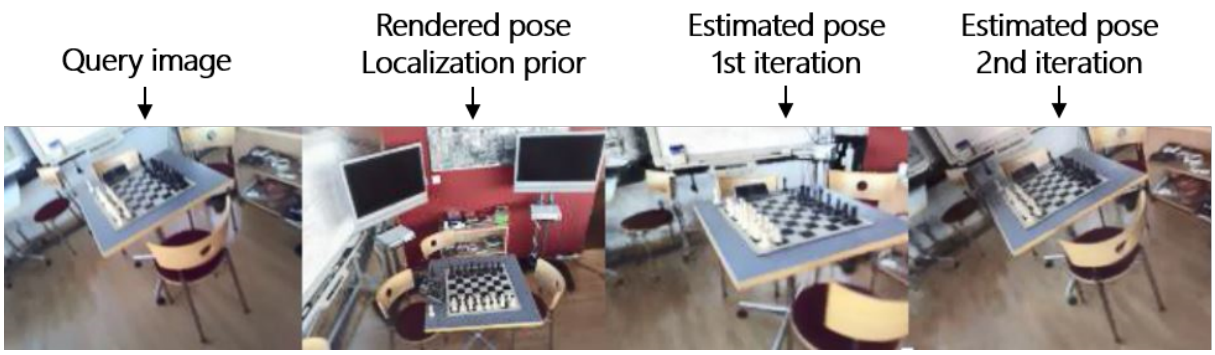


Figure 7.6: **Iterative visual localization from an imprecise prior.** Starting from a coarse localization prior, our algorithm estimates the pose of a query image iteratively by comparing image features to descriptors rendered from a neural scene representation.

To measure how bad initialization impacts results, we conducted an experiment on the Chess scene

Figure 7.7: **Success and failure cases:** Using dense features field for localization enables to establish accurate correspondences in texture-less areas (left). Failure cases are observed in the presence of dynamic objects (middle), for which the PnP converges on a wrong pool of matches, and ambiguous cases (right) where the CNN mixes up the symmetrical parts of the church due to lack of long-range reasoning.

with a constant prior for all test images (see Table 7.2). We observe that, thanks to our iterative refinement, imprecise priors do not affect the final localization accuracy but rather require more iterations to reach the correct camera pose.

Table 7.2: Median error w.r.t. prior strategy and No. of iterations.

| cm / ° | Prior | Iter 1 | Iter 2 | Iter 3 |
|---|---|---|---|---|
| Retrieval | 0.22 / 12.1 | 0.02 / 0.7 | 0.01 / 0.5 | 0.01 / 0.4 |
| Constant | 1.82 / 32.2 | 0.12 / 2.8 | 0.02 / 0.6 | 0.01 / 0.5 |

### 7.4.3 Ablation studies

#### 7.4.3.1 Descriptor loss

The self-supervised loss used to train descriptors is similar to the triplet loss commonly used for metric learning, except an additionnal term for negative pairs which depends on the 3D distance between points. We propose a qualitative comparison between the triplet loss and our proposal in figure 7.8. We observe that the representation learned by our system is smooth and more expressive than the triplet loss which only separate the scene into few clusters.

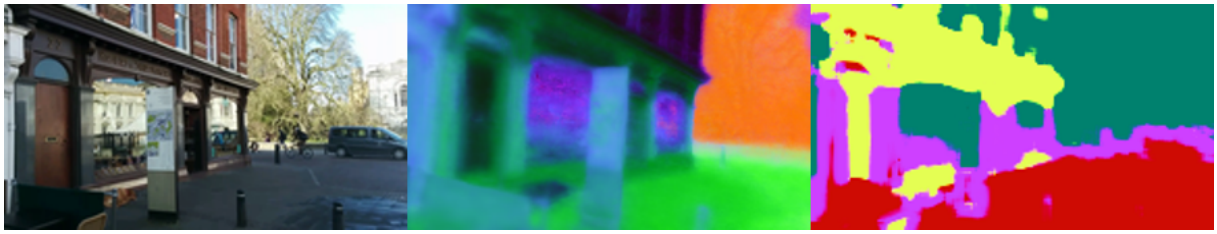

Figure 7.8: **Qualitative comparison of descriptors between the proposed loss and a classical triplet loss.** We visualize the PCA of descriptors from our loss (middle) and a triplet (right) for a given query image (left).

In addition to that, we propose an additional experiment where we investigate the impact of the parameter $\lambda$ on the proposed loss. We train 5 models on the "Chess" scene from the 7scenes dataset [Shotton et al., 2013] with different values for $\lambda$, while keeping other parameters fixed. The results are displayed in Fig 7.9.

We observe that when using extreme values such as 0.1 or 10 for this parameter, the resulting model is not able to produce reliable matches and the localization process fails. When $\lambda$ is small, the model learns similar descriptors for the entire scene and then is not able to effectively discriminate between

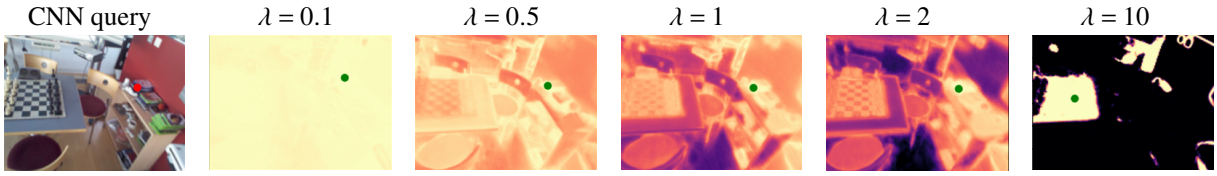


Figure 7.9: **Similarities scores between a CNN query pixel and rendered descriptors, depending on $\lambda$.** The left image shows a query pixel from a test sample while other images show a colormap of the similarity between rendered descriptors from the same viewpoint and the query pixel. Yellow color indicates a high similarity.

areas. On the other way, when $\lambda$ is too large, we ask the model to produce very different descriptors for very close points. While it could be seen as a good local property, we observe that this constraint can not be fully satisfied for a whole scene, given the compact 32-dimensional descriptors we use. A large $\lambda$ parameter creates ambiguities because the same descriptor is attributed to points far from each other in the scene, resulting in wrong matches.

All experiments reported in the main paper used $\lambda = 1$, and we observe that reasonable values ($\lambda = 0.5$ and $\lambda = 2$) for this parameter results in a similar localization accuracy, suggesting that a per-scene tuning of $\lambda$ is not necessary for deploying in a new environment.

Concerning localization accuracy on the test set, models trained with $\lambda = 0.1$ and $\lambda = 10$ completely fail to converge to an accurate pose, while both 3 models trained with $\lambda = 0.5, 1, 2$ have the same median error: 1cm and 0.4°.

#### 7.4.3.2 Implicit descriptors

We modeled the descriptors learned by the neural renderer as independent of the direction from which the point is observed. We verify that this choice is relevant by comparing it to the view-dependent case. Modeling the descriptors as dependent on the image appearance is not feasible because this parameter is unknown during the localization step.

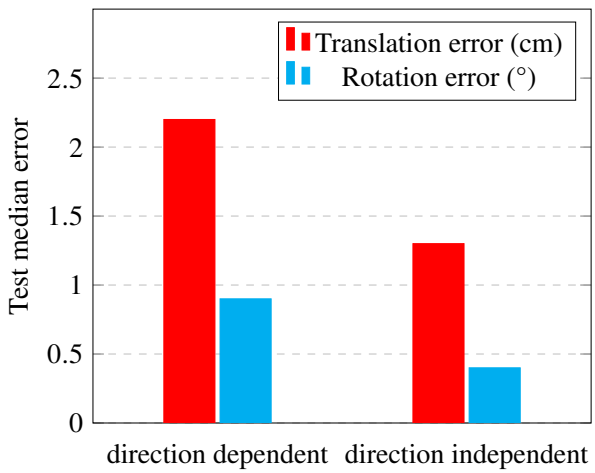


Figure 7.10: **Localization accuracy depending on descriptor head inputs.** We compare the final accuracy on the "Chess" scene with and without the viewing direction as descriptor input in the neural renderer.

#### 7.4.3.3 Reconstruction losses

We evaluated the benefits of the $\mathcal{L}_{SSIM}$ and $\mathcal{L}_{TV}$ terms of the loss function. On the Heads scene, the error is 3cm/2.3° with the proposed loss, 4cm/2.1° without $\mathcal{L}_{SSIM}$ and 6cm/4.0° without $\mathcal{L}_{TV}$. These terms actually improve the localization accuracy because they help to recover the correct scene geometry.

### 7.4.4 Efficiency

The storage requirement of our modules is 50MB (48MB for the hash tables and 2MB for the neural networks). In contrast with explicit maps, this number does not grow with the amount of reference data (see Figure 6.6).

Inference times are: 9ms for features extraction, 5ms for rendering, 5ms for dense matching and $\approx$ 60ms for PnP+RANSAC (because we have a lot of matches), resulting in $\approx$ 200ms for the total time with 3 iterations reported in the experiments (RTX 3090 GPU). Speedup can be achieved easily by less refinements, at the cost of minor accuracy drop.

## 7.5 Limitations and Future Work

**7.5.0.0.1 Scalability:** Similar to other Neural Scene Representations, our Neural Field struggles to represent large scale maps, such as the one used in autonomous driving, with a single radiance field instance. The current best solution, proposed by Block-NeRF [Tancik et al., 2022], is to split the environment into several smaller neural fields and enforce consistency at their boundaries. This solution is successful at a city-scale and could be implemented in our method for large scale localizatoin.

**7.5.0.0.2 Localization pipeline:** The proposed localization algorithm could be improved in many ways. Dense features matching could be performed by learning-based approaches [Truong et al., 2020, Berton et al., 2021] instead of simple heuristics. Resulting 2D-3D matchs could be improved by co-visibility filtering [Sattler et al., 2012, Panek et al., 2022]. Finally, the estimated camera pose could be optimized by direct features alignment, similar to GN-Net [Von Stumberg et al., 2020] and PixLoc [Sarlin et al., 2021]. The contribution of this paper lies in the learning of descriptors in a neural renderer, and this proposal can be used as a backbone for different and more advanced localization solutions.

**7.5.0.0.3 Computational cost:** Volumetric-based neural renderers enable to densely represent scenes in a compact way, at the cost of an expensive computation to render each view (mitigated by hash encoding and small resolution). Dense features matching between 2 views also generate many 2D-3D matches which can make the RANSAC process slow. This could be improved, as a future work, by computing keypoints detection scores in order to reduce the number of potential matches.

## 7.6 Conclusion

This chapter proposed Neural Positional Features Fields; a new way to learn and represent visual localization maps based on neural radiance fields.

The proposed formulation has the advantage of densely representing scenes with a small memory footprint. We demonstrate that the non-supervised learned local features, which are specialized for the target area, perform better than related visual localization methods, even when using basic localization techniques. The proposed pipeline can serve as a backbone to more advanced features matching pipelines and should be compatible with future improvements in the neural rendering field that could enable to scale these models to larger scenes and yield better localization accuracy by improving further the quality of the learned scene geometry.

Chapter 8

# Conclusion

## Contents


8.1 Résumé en Français . . . . . 100

8.2 Summary of contributions and discussions . . . . . 101

8.3 Future work: a hierarchical vehicle localization algorithm in implicit maps . . . . 102

8.4 Perspectives: an autonomous driving solution based on neural scenes representations . . . . . 103

## 8.1 Résumé en Français

Nous résumons nos contributions, qui peuvent être distinguées en deux parties. Tout d'abord nous avons évalué et amélioré les méthodes de machine learning utilisées dans des systèmes de localisation de véhicules en temps réel. Cela à conduit au développement et à la publication des algorithmes CoordiNet et ImPosing. Ensuite nous avons exploré l'idée de connecter la tâche de localisation visuelle et les méthodes NeRF. Dans ce cadre, nous avons proposé la méthodes d'augmentation de données LENS et l'algorithme CROSSFIRE.

Par la suite, nous expliquons comment les représentations implicites de cartes proposées pourraient être combinées dans un système de localisation visuelle à grande échelle. Nous concluons le manuscrit en discutant de la perspective de solutions de conduite autonome basées sur des représentations implicites.

## 8.2 Summary of contributions and discussions

Overall, the thesis has followed two main research directions:

- **Develop and evaluate direct learning-based methods for vehicle localization.** We have investigated the capability and the limits of direct-learning based methods to be used as vehicle localization systems. We proposed 2 algorithms for this task which improved the state-of-the-art in this field. We also proposed a method to quantify the uncertainty of the models, and used it in a Kalman filter to provide a smooth vehicle localization module. We investigated the trainings of these algorithms, observed how data hungry they are and that pose regression was not convenient to deploy on multiple maps. We proposed ideas to tackle these challenges with synthetic data and implicit representations. The technology we have developed has good scaling properties. The implicit maps we use are easy to transfer and very compact to store in a vehicle. Relocalization algorithms can be computed at a high frame rate. More interestingly, because the raw collected data can be automatically annotated by SfM, the system accuracy improves over time by collecting new data without human intervention.

- **Explore the connection with NeRF.** We started to study NeRF in 2021. We were excited by the cute rendering examples on the internet but also because it addressed in a new way a (classical) problem we were trying to solve: how to reason in 3D using 2D images? The connection with visual localization was obvious: both deals with images and camera poses as inputs/outputs. If NeRF is able to learn the correct 3D geometry of the scene with only images and camera poses, my visual localization algorithm should build on this. We started by the simplest way to transfer knowledge from a NeRF to a pose regressor: synthesize images uniformly distributed on the scene. Pose regression can not extrapolate ? Let's synthesize all the scene. It worked really well on the metrics! These datasets are usually difficult for these methods specifically because they require extrapolation capability on novel views, which is provided by NeRF in our solution. Then, we had this idea of storing learned features discretely distributed on the map. It looked like NeuMap [Tang et al., 2022] which has been released since then, but we finally had the idea to replace the discretization by an implicit formulation. Then we obtained an algorithm with global features and compared it to image retrieval. Finally, extending the idea to local features was following the same path, but led to long sleepless nights before conference deadlines. Implicit maps are promising. Our proposals are the first to use it for visual localization but there is huge room for improvement on this direction.

We conducted real-world experiments with our algorithms and confirmed their localization accuracy. ImPosing has even been tried as the localization module of an experimental autonomous driving pipeline, which navigated successfully on open roads for few kilometers in Shanghai, without any human intervention. While this achievement was really satisfying, a lot remain to be done before industrial deployment of the proposed solutions, such as more experiments on diverse scenarios to ensure the robustness of the solution or more work on detecting the potential failure modes.

We also advise against the use of our algorithms as the single localization system of an autonomous vehicle, for obvious safety reasons. One can not rely on a single sensor for such a purpose because system failures can cause dramatic consequences in autonomous driving. The safe solution consists in relying

on several sensors and several algorithms, through sensor fusion [Levinson et al., 2007] and redundant systems [Granados et al., 2020].

## 8.3 Future work: a hierarchical vehicle localization algorithm in implicit maps

The algorithms proposed in chapters 6 and 7 can be combined in a hierarchical structure-based method. Given an autonomous driving map, ImPosing can be used to provide a (sub)meter-level camera pose. Then we refine the estimate with dense features matching on a Neural Positional Features Field. Instead of training 2 separate CNNs for global and local features extraction, a single CNN with 2 output branches, similar to HLoc [Sarlin et al., 2019] can be used.

This coarse-to-fine approach relates to best performing structure-based pipelines for visual localization, such as MegLoc [Peng et al., 2021], Kaptur [Humenberger et al., 2020] or HLoc [Sarlin et al., 2019], which combine image retrieval and sparse features matching in a hierarchical manner (shown in Figure 2.4). The novelty we have compared to these methods is the map representation. Instead of the discrete and sparse explicit data structures commonly used, we learn implicit maps from which global and local features can be densely queried from any coordinates.

We show in Figure 8.1 an illustration of the model we just described. By lack of time, we did not conduct experiments with this algorithm, and let it as future work. One limitation is the current inability of neural fields to handle very large scenes, such that decomposing the area in individually trained neural fields is required [Tancik et al., 2022].

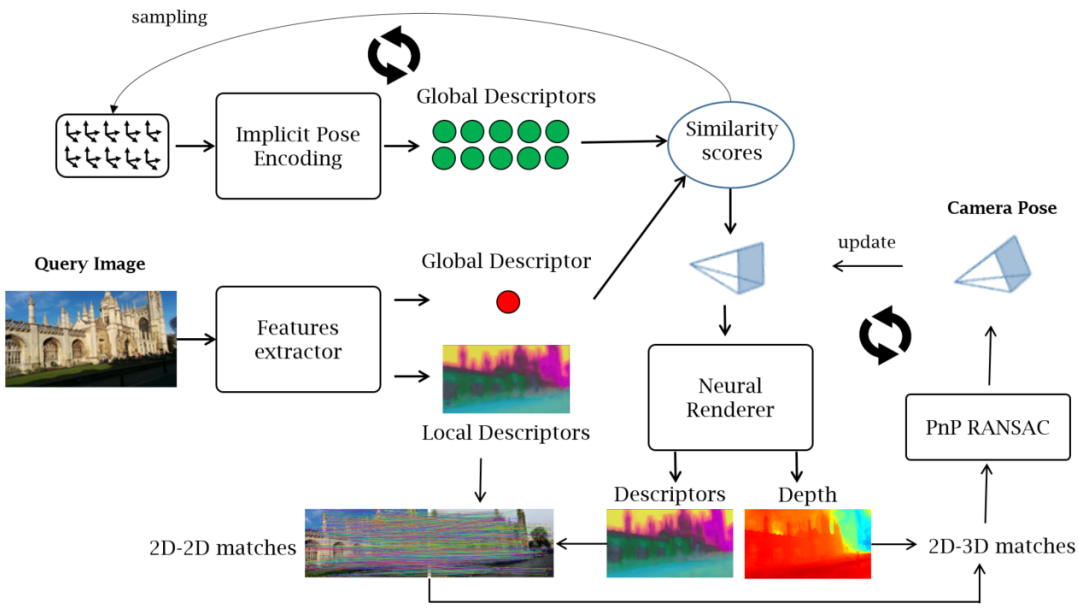


Figure 8.1: **Hierarchical visual localization with implicit maps.** A first camera pose is estimated by ImPosing and then refined by dense features matching in a Neural Positional Features Field.

The algorithm LENS, proposed in chapter 5, can naturally be used by this algorithm because the Neural Positional Features Field can render photorealistic RGB images. The rendering of the small scales images we use is fast, such that synthetic images from unseen viewpoints could be generated in

parallel of training. It would be interesting to measure the impact of LENS on ImPosing and NPFF. We suppose that this impact highly depends on the quality of the geometry learned by the neural field.

## 8.4 Perspectives: an autonomous driving solution based on neural scenes representations

Beyond our work on localization, neural scenes representations are a very promising direction for computer vision and autonomous driving. Being able to process a raw video into a sophisticated neural map with geometry, semantics, localization features and other modalities is very useful.

First, annotated synthetic data can be rendered and used to train better algorithms. We have experienced it for localization and expect the same for many perception tasks. If dynamic objects are correctly modelled and controllable, we can use the neural field as a (differentiable) photo-realistic simulator [Cleac'h et al., 2022] for reinforcement learning.

Then, we can also imagine an autonomous driving or mobile robots navigation based on neural maps, from perception and localization to planning. Embedding local scene properties in coordinate-based network and build interaction between these scene representations and other features can be beneficial at many steps of an autonomous pipeline.

Chapter 9

# Bibliography


[Achaji et al., 2022] Achaji, L., Barry, T., Fouqueray, T., Moreau, J., Aioun, F., and Charpillet, F. (2022). Pretr: spatio-temporal non-autoregressive trajectory prediction transformer. In 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), pages 2457–2464. IEEE.

[Adamkiewicz et al., 2022] Adamkiewicz, M., Chen, T., Caccavale, A., Gardner, R., Culbertson, P., Bohg, J., and Schwager, M. (2022). Vision-only robot navigation in a neural radiance world. IEEE Robotics and Automation Letters, 7(2):4606–4613.

[Arandjelović et al., 2016] Arandjelović, R., Gronat, P., Torii, A., Pajdla, T., and Sivic, J. (2016). NetVLAD: CNN architecture for weakly supervised place recognition. In IEEE Conference on Computer Vision and Pattern Recognition.

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

[Attal et al., 2022] Attal, B., Huang, J.-B., Zollhöfer, M., Kopf, J., and Kim, C. (2022). Learning neural light fields with ray-space embedding networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[Atzmon and Lipman, 2020] Atzmon, M. and Lipman, Y. (2020). Sal: Sign agnostic learning of shapes from raw data. In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[Aubry et al., 2014] Aubry, M., Russell, B. C., and Sivic, J. (2014). Painting-to-3D model alignment via discriminative visual elements. In ACM Transactions on Graphics (ToG), volume 33, pages 1–14. ACM New York, NY, USA.

[Baker and Matthews, 2004] Baker, S. and Matthews, I. (2004). Lucas-kanade 20 years on: A unifying framework. International journal of computer vision, 56(3):221–255.

[Balntas et al., 2018] Balntas, V., Li, S., and Prisacariu, V. (2018). Relocnet: Continuous metric learning relocalisation using neural nets. In The European Conference on Computer Vision (ECCV).

[Barron et al., 2021] Barron, J. T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., and Srinivasan, P. P. (2021). Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. ICCV.

[Bârsan et al., 2018] Bârsan, I. A., Wang, S., Pokrovsky, A., and Urtasun, R. (2018). Learning to localize using a lidar intensity map. In Conference on Robot Learning.

[Bay et al., 2006] Bay, H., Tuytelaars, T., and Gool, L. V. (2006). Surf: Speeded up robust features. In Proceedings of the European Conference on Computer Vision (ECCV), pages 404–417. Springer.

[Bemana et al., 2020] Bemana, M., Myszkowski, K., Seidel, H.-P., and Ritschel, T. (2020). X-fields: Implicit neural view-, light- and time-image interpolation. ACM Transactions on Graphics (Proc. SIGGRAPH Asia 2020), 39(6).

[Berton et al., 2021] Berton, G., Masone, C., Paolicelli, V., and Caputo, B. (2021). Viewpoint invariant dense matching for visual geolocalization. In Proceedings of the International Conference on Computer Vision (ICCV), pages 12169–12178.

[Betke et al., 2000] Betke, M., Haritaoglu, E., and Davis, L. S. (2000). Real-time multiple vehicle detection and tracking from a moving vehicle. Machine vision and applications, 12(2):69–83.

[Bhat et al., 2021] Bhat, S. F., Alhashim, I., and Wonka, P. (2021). Adabins: Depth estimation using adaptive bins. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4009–4018.

[Blanton et al., 2020] Blanton, H., Greenwell, C., Workman, S., and Jacobs, N. (2020). Extending absolute pose regression to multiple scenes. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pages 38–39.

[Brachmann et al., 2017] Brachmann, E., Krull, A., Nowozin, S., Shotton, J., Michel, F., Gumhold, S., and Rother, C. (2017). Dsac-differentiable ransac for camera localization. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 6684–6692.

[Brachmann and Rother, 2018] Brachmann, E. and Rother, C. (2018). Learning less is more - 6d camera localization via 3d surface regression. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[Brachmann and Rother, 2019] Brachmann, E. and Rother, C. (2019). Expert sample consensus applied to camera re-localization. In ICCV.

[Brachmann and Rother, 2021] Brachmann, E. and Rother, C. (2021). Visual camera re-localization from RGB and RGB-D images using dsac. In IEEE Transactions on Pattern Analysis and Machine Intelligence, pages 1–1.

[Brahmbhatt et al., 2018] Brahmbhatt, S., Gu, J., Kim, K., Hays, J., and Kautz, J. (2018). Geometry-aware learning of maps for camera localization. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[Breyer et al., 2021] Breyer, M., Chung, J. J., Ott, L., Siegwart, R. Y., and Nieto, J. I. (2021). Volumetric grasping network: Real-time 6 dof grasp detection in clutter. In Conference on Robot Learning.

[Bruns et al., 2022] Bruns, L., Zangeneh, F., and Jensfelt, P. (2022). Sdf-based rgb-d camera tracking in neural scene representations. arXiv preprint arXiv:2205.02079.

[Bujnak et al., 2008] Bujnak, M., Kukelova, Z., and Pajdla, T. (2008). A general solution to the p4p problem for camera with unknown focal length. In 2008 IEEE Conference on Computer Vision and Pattern Recognition, pages 1–8.

[Camposeco et al., 2019] Camposeco, F., Cohen, A., Pollefeys, M., and Sattler, T. (2019). Hybrid scene compression for visual localization. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), pages 7645–7654.

[Cao and de Charette, 2022] Cao, A.-Q. and de Charette, R. (2022). Scenerf: Self-supervised monocular 3d scene reconstruction with radiance fields. In arxiv.

[Chalvatzaras et al., 2022] Chalvatzaras, A., Pratikakis, I., and Amanatiadis, A. A. (2022). A survey on map-based localization techniques for autonomous vehicles. IEEE Transactions on Intelligent Vehicles, pages 1–23.

[Chen et al., 2022a] Chen, A., Xu, Z., Geiger, A., Yu, J., and Su, H. (2022a). Tensorf: Tensorial radiance fields. In European Conference on Computer Vision (ECCV).

[Chen et al., 2017] Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2017). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence, 40(4):834–848.

[Chen et al., 2022b] Chen, S., Li, X., Wang, Z., and Prisacariu, V. (2022b). Dfnet: Enhance absolute pose regression with direct feature matching. In Proceedings of the European Conference on Computer Vision (ECCV).

[Chen and Li, 2004] Chen, S. and Li, Y. (2004). Automatic sensor placement for model-based robot vision. In IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), volume 34, pages 393–408. IEEE.

[Chen et al., 2021a] Chen, S., Wang, Z., and Prisacariu, V. (2021a). Direct-posenet: Absolute pose regression with photometric consistency. In 2021 International Conference on 3D Vision (3DV), pages 1175–1185. IEEE.

[Chen et al., 2022c] Chen, W., Liu, Y., Wang, W., Bakker, E. M., Georgiou, T., Fieguth, P., Liu, L., and Lew, M. S. (2022c). Deep learning for instance retrieval: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence.

[Chen et al., 2021b] Chen, Y., Liu, S., and Wang, X. (2021b). Learning continuous image representation with local implicit image function. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 8628–8638.

[Chen et al., 2022d] Chen, Z., Chen, Y., Liu, J., Xu, X., Goel, V., Wang, Z., Shi, H., and Wang, X. (2022d). Videoinr: Learning video implicit neural representation for continuous space-time super-resolution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 2047–2057.

[Choy et al., 2016] Choy, C. B., Gwak, J., Savarese, S., and Chandraker, M. (2016). Universal correspondence network. In Proceedings of Advances in Neural Information Processing Systems (NeurIPS. Curran Associates, Inc.

[Clark et al., 2017] Clark, R., Wang, S., Markham, A., Trigoni, N., and Wen, H. (2017). Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6856–6864.

[Cleac'h et al., 2022] Cleac'h, S. L., Yu, H.-X., Guo, M., Howell, T. A., Gao, R., Wu, J., Manchester, Z., and Schwager, M. (2022). Differentiable physics simulation of dynamics-augmented neural objects. arXiv preprint arXiv:2210.09420.

[Cortes and Mohri, 2011] Cortes, C. and Mohri, M. (2011). Domain adaptation in regression. In International Conference on Algorithmic Learning Theory, pages 308–323. Springer.

[Davis et al., 2012] Davis, A., Levoy, M., and Durand, F. (2012). Unstructured light fields. In Computer Graphics Forum, volume 31, pages 305–314. Wiley Online Library.

[Debevec et al., 1996] Debevec, P. E., Taylor, C. J., and Malik, J. (1996). Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '96, page 11–20, New York, NY, USA. Association for Computing Machinery.

[Dellenbach et al., 2021] Dellenbach, P., Deschaud, J.-E., Jacquet, B., and Goulette, F. (2021). What's in my lidar odometry toolbox? In 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 4429–4436. IEEE.

[Deng et al., 2021] Deng, C., Litany, O., Duan, Y., Poulenard, A., Tagliasacchi, A., and Guibas, L. (2021). Vector neurons: A general framework for so (3)-equivariant networks. arXiv preprint arXiv:2104.12229.

[Deng and Tartaglione, 2023] Deng, C. L. and Tartaglione, E. (2023). Compressing explicit voxel grid representations: Fast nerfs become also small. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pages 1236–1245.

[Deng et al., 2009] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pages 248–255. Ieee.

[Deng et al., 2022] Deng, K., Liu, A., Zhu, J.-Y., and Ramanan, D. (2022). Depth-supervised NeRF: Fewer views and faster training for free. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).

[Deschaud, 2018] Deschaud, J.-E. (2018). Imls-slam: Scan-to-model matching based on 3d data. In 2018 IEEE International Conference on Robotics and Automation (ICRA), pages 2480–2485.

[DeTone et al., 2018] DeTone, D., Malisiewicz, T., and Rabinovich, A. (2018). SuperPoint: Self-supervised interest point detection and description. In CVPR Deep Learning for Visual SLAM Workshop.

[Ding et al., 2019] Ding, M., Wang, Z., Sun, J., Shi, J., and Luo, P. (2019). Camnet: Coarse-to-fine retrieval for camera re-localization. In 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pages 2871–2880.

[Dinh-Van et al., 2017] Dinh-Van, N., Nashashibi, F., Thanh-Huong, N., and Castelli, E. (2017). Indoor intelligent vehicle localization using wifi received signal strength indicator. In 2017 IEEE MTT-S international conference on microwaves for intelligent mobility (ICMIM), pages 33–36. IEEE.

[Dollar et al., 2011] Dollar, P., Wojek, C., Schiele, B., and Perona, P. (2011). Pedestrian detection: An evaluation of the state of the art. IEEE transactions on pattern analysis and machine intelligence, 34(4):743–761.

[Dupont et al., 2021] Dupont, E., Goliński, A., Alizadeh, M., Teh, Y. W., and Doucet, A. (2021). Coin: Compression with implicit neural representations. arXiv preprint arXiv:2103.03123.

[Dupont et al., 2022] Dupont, E., Kim, H., Eslami, S. M. A., Rezende, D. J., and Rosenbaum, D. (2022). From data to functa: Your data point is a function and you can treat it like one. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S., editors, Proceedings of the 39th International Conference on Machine Learning, volume 162 of Proceedings of Machine Learning Research, pages 5694–5725. PMLR.

[Dusmanu et al., 2019] Dusmanu, M., Rocco, I., Pajdla, T., Pollefeys, M., Sivic, J., Torii, A., and Sattler, T. (2019). D2-Net: A Trainable CNN for Joint Detection and Description of Local Features. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR).

[Ehret et al., 2022] Ehret, T., Marí, R., and Facciolo, G. (2022). Nerf, meet differential geometry! arXiv preprint arXiv:2206.14938.

[Engel et al., 2014] Engel, J., Schöps, T., and Cremers, D. (2014). Lsd-slam: Large-scale direct monocular slam. In Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part II 13, pages 834–849. Springer.

[Facil et al., 2019] Facil, J. M., Ummenhofer, B., Zhou, H., Montesano, L., Brox, T., and Civera, J. (2019). Cam-convs: Camera-aware multi-scale convolutions for single-view depth. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 11826–11835.

[Fayyad et al., 2020] Fayyad, J., Jaradat, M. A., Gruyer, D., and Najjaran, H. (2020). Deep learning sensor fusion for autonomous vehicle perception and localization: A review. Sensors, 20(15):4220.

[Fischler and Bolles, 1981] Fischler, M. and Bolles, R. (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.

[Fridovich-Keil and Yu et al., 2022] Fridovich-Keil and Yu, Tancik, M., Chen, Q., Recht, B., and Kanazawa, A. (2022). Plenoxels: Radiance fields without neural networks. In CVPR.

[Furukawa et al., 2015] Furukawa, Y., Hernández, C., et al. (2015). Multi-view stereo: A tutorial. Foundations and Trends® in Computer Graphics and Vision, 9(1-2):1–148.

[Gal and Ghahramani, 2016] Gal, Y. and Ghahramani, Z. (2016). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In international conference on machine learning, pages 1050–1059. PMLR.

[Garbin et al., 2021] Garbin, S. J., Kowalski, M., Johnson, M., Shotton, J., and Valentin, J. (2021). Fastnerf: High-fidelity neural rendering at 200fps. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 14346–14355.

[Garg et al., 2021] Garg, S., Fischer, T., and Milford, M. (2021). Where is your place, visual place recognition? In Zhou, Z.-H., editor, Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21, pages 4416–4425. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

[Germain et al., 2022] Germain, H., DeTone, D., Pascoe, G., Schmidt, T., Novotny, D., Newcombe, R., Sweeney, C., Szeliski, R., and Balntas, V. (2022). Feature query networks: Neural surface description for camera pose refinement. In Proceedings of the Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pages 5067–5077.

[Gesnouin et al., 2021] Gesnouin, J., Pechberti, S., Stanciulcscu, B., and Moutarde, F. (2021). Trouspinet: Spatio-temporal attention on parallel atrous convolutions and u-grus for skeletal pedestrian crossing prediction. In 2021 16th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2021), pages 01–07. IEEE.

[Gilles et al., 2021] Gilles, T., Sabatini, S., Tsishkou, D., Stanciulescu, B., and Moutarde, F. (2021). Home: Heatmap output for future motion estimation. In 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), pages 500–507. IEEE.

[Godard et al., 2019] Godard, C., Mac Aodha, O., Firman, M., and Brostow, G. J. (2019). Digging into self-supervised monocular depth estimation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 3828–3838.

[Goli et al., 2022] Goli, L., Rebain, D., Sabour, S., Garg, A., and Tagliasacchi, A. (2022). nerf2nerf: Pairwise registration of neural radiance fields. arxiv.

[Gordo et al., 2017] Gordo, A., Almazan, J., Revaud, J., and Larlus, D. (2017). End-to-end learning of deep visual representations for image retrieval. IJCV.

[Gortler et al., 1996] Gortler, S. J., Grzeszczuk, R., Szeliski, R., and Cohen, M. F. (1996). The lumigraph. In Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, pages 43–54.

[Granados et al., 2020] Granados, J. Á. F., Batalla, J. M., and Togay, C. (2020). Redundant localization system for automatic vehicles. Mechanical Systems and Signal Processing, 136:106433.

[Haralick et al., 1991] Haralick, R., Lee, D., Ottenburg, K., and Nolle, M. (1991). Analysis and solutions of the three point perspective pose estimation problem. In Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 592–598.

[Harris and Stephens, 1988] Harris, C. G. and Stephens, M. J. (1988). A combined corner and edge detector. In Alvey Vision Conference.

[Hartley and Zisserman, 2003] Hartley, R. and Zisserman, A. (2003). Multiple View Geometry in Computer Vision. Cambridge University Press, New York, NY, USA, 2 edition.

[He et al., 2016] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778.

[Hedman et al., 2021] Hedman, P., Srinivasan, P. P., Mildenhall, B., Barron, J. T., and Debevec, P. (2021). Baking neural radiance fields for real-time view synthesis. ICCV.

[Herau et al., 2023] Herau, Q., Piasco, N., Bennehar, M., Roldão, L., Tsishkou, D., Migniot, C., Vasseur, P., and Demonceaux, C. (2023). Moisst: Multi-modal optimization of implicit scene for spatiotemporal calibration. arXiv preprint arXiv:2303.03056.

[Hernández et al., 2017] Hernández, N., Hussein, A., Cruzado, D., Parra, I., and Armingol, J. M. (2017). Applying low cost wifi-based localization to in-campus autonomous vehicles. In 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC), pages 1–6. IEEE.

[Hu et al., 2017] Hu, Y., Wang, B., and Lin, S. (2017). Fc 4: Fully convolutional color constancy with confidence-weighted pooling. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4085–4094.

[Huang et al., 2022] Huang, L., Hodan, T., Ma, L., Zhang, L., Tran, L., Twigg, C., Wu, P.-C., Yuan, J., Keskin, C., and Wang, R. (2022). Neural correspondence field for object pose estimation. In European Conference on Computer Vision (ECCV).

[Huang et al., 2019] Huang, Z., Xu, Y., Shi, J., Zhou, X., Bao, H., and Zhang, G. (2019). Prior guided dropout for robust visual localization in dynamic environments. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV).

[Humemberger et al., 2021] Humemberger, M., Csurka, G., Guerin, N., and Chidlovskii, B. (2021). Methods for visual localization.

[Humenberger et al., 2020] Humenberger, M., Cabon, Y., Guerin, N., Morat, J., Revaud, J., Rerole, P., Pion, N., de Souza, C., Leroy, V., and Csurka, G. (2020). Robust image retrieval-based visual localization using kapture.

[Humenberger et al., 2022] Humenberger, M., Cabon, Y., Pion, N., Weinzaepfel, P., Lee, D., Guérin, N., Sattler, T., and Csurka, G. (2022). Investigating the role of image retrieval for visual localization. International Journal of Computer Vision, pages 1–26.

[Ichnowski et al., 2020] Ichnowski, J., Avigal, Y., Kerr, J., and Goldberg, K. (2020). Dex-NeRF: Using a neural radiance field to grasp transparent objects. In Conference on Robot Learning (CoRL).

[Irschara et al., 2009] Irschara, A., Zach, C., Frahm, J.-M., and Bischof, H. (2009). From structure-from-motion point clouds to fast location recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2599–2606.

[Irshad et al., 2022] Irshad, M. Z., Zakharov, S., Ambrus, R., Kollar, T., Kira, Z., and Gaidon, A. (2022). Shapo: Implicit representations for multi object shape appearance and pose optimization. In European Conference on Computer Vision (ECCV).

[Jahrer et al., 2008] Jahrer, M., Grabner, M., and Bischof, H. (2008). Learned local descriptors for recognition and matching. In Computer Vision Winter Workshop, volume 2, pages 103–118.

[Jégou et al., 2010] 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.

[Jiang et al., 2020] Jiang, C. M., Sud, A., Makadia, A., Huang, J., Nießner, M., and Funkhouser, T. (2020). Local implicit grid representations for 3d scenes. In Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).

[Kendall and Cipolla, 2016] Kendall, A. and Cipolla, R. (2016). Modelling uncertainty in deep learning for camera relocalization. In 2016 IEEE international conference on Robotics and Automation (ICRA), page 4762–4769. IEEE Press.

[Kendall and Cipolla, 2017] Kendall, A. and Cipolla, R. (2017). Geometric loss functions for camera pose regression with deep learning. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 6555–6564.

[Kendall and Gal, 2017] Kendall, A. and Gal, Y. (2017). What uncertainties do we need in bayesian deep learning for computer vision? In Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, page 5580–5590, Red Hook, NY, USA. Curran Associates Inc.

[Kendall et al., 2018] Kendall, A., Gal, Y., and Cipolla, R. (2018). Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 7482–7491.

[Kendall et al., 2015] Kendall, A., Grimes, M., and Cipolla, R. (2015). PoseNet: A convolutional network for real-time 6-DOF camera relocalization. In Proceedings of the IEEE International Conference on Computer Vision, pages 2938–2946.

[Kerr et al., 2022] Kerr, J., Fu, L., Huang, H., Ichnowski, J., Tancik, M., Avigal, Y., Kanazawa, A., and Goldberg, K. (2022). Evo-nerf: Evolving nerf for sequential robot grasping. In Proc. 6th Annu. Conf. Robot Learn., pages 1–15.

[Kukelova et al., 2011] Kukelova, Z., Bujnak, M., and Pajdla, T. (2011). Closed-form solutions to minimal absolute pose problems with known vertical direction. In Kimmel, R., Klette, R., and Sugimoto, A., editors, Computer Vision – ACCV 2010, pages 216–229, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Kundu et al., 2022] Kundu, A., Genova, K., Yin, X., Fathi, A., Pantofaru, C., Guibas, L., Tagliasacchi, A., Dellaert, F., and Funkhouser, T. (2022). Panoptic neural fields: A semantic object-aware neural scene representation. In CVPR.

[Kundu et al., 2020] Kundu, A., Yin, X., Fathi, A., Ross, D., Brewington, B., Funkhouser, T., and Pantofaru, C. (2020). Virtual multi-view fusion for 3D semantic segmentation. In European Conference on Computer Vision, pages 518–535. Springer.

[Kuutti et al., 2018] Kuutti, S., Fallah, S., Katsaros, K., Dianati, M., Mccullough, F., and Mouzakitis, A. (2018). A survey of the state-of-the-art localization techniques and their potentials for autonomous vehicle applications. IEEE Internet of Things Journal, 5(2):829–846.

[Lakshminarayanan et al., 2017] Lakshminarayanan, B., Pritzel, A., and Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. Advances in Neural Information Processing Systems, 30.

[Larsson, 2020] Larsson, V. (2020). PoseLib - Minimal Solvers for Camera Pose Estimation.

[Larsson et al., 2017a] Larsson, V., Astrom, K., and Oskarsson, M. (2017a). Efficient solvers for minimal problems by syzygy-based reduction. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[Larsson et al., 2017b] Larsson, V., Kukelova, Z., and Zheng, Y. (2017b). Making minimal solvers for absolute pose estimation compact and robust. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 2335–2343.

[Laskar et al., 2017] Laskar, Z., Melekhov, I., Kalia, S., and Kannala, J. (2017). Camera relocalization by computing pairwise relative poses using convolutional neural network. In The IEEE International Conference on Computer Vision (ICCV).

[Lee et al., 2021] Lee, S. J., Kim, D., Hwang, S. S., and Lee, D. (2021). Local to global: Efficient visual localization for a monocular camera. In 2021 IEEE Winter Conference on Applications of Computer Vision (WACV), pages 2230–2239.

[Lefèvre et al., 2014] Lefèvre, S., Vasquez, D., and Laugier, C. (2014). A survey on motion prediction and risk assessment for intelligent vehicles. ROBOMECH journal, 1(1):1–14.

[Lepetit et al., 2009] Lepetit, V., Moreno-Noguer, F., and Fua, P. (2009). Epnp: An accurate o(n) solution to the pnp problem. International Journal Of Computer Vision, 81:155–166.

[Levinson et al., 2007] Levinson, J., Montemerlo, M., and Thrun, S. (2007). Map-based precision vehicle localization in urban environments. In Robotics: science and systems, volume 4, page 1. Atlanta, GA, USA.

[Levoy and Hanrahan, 1996] Levoy, M. and Hanrahan, P. (1996). Light field rendering. In Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, pages 31–42.

[Li et al., 2020] Li, D., Shi, X., Long, Q., Liu, S., Yang, W., Wang, F., Wei, Q., and Qiao, F. (2020). DXSLAM: A robust and efficient visual SLAM system with deep features. arXiv preprint arXiv:2008.05416.

[Li et al., 2022a] Li, T., Slavcheva, M., Zollhoefer, M., Green, S., Lassner, C., Kim, C., Schmidt, T., Lovegrove, S., Goesele, M., Newcombe, R., et al. (2022a). Neural 3d video synthesis from multi-view video. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5521–5531.

[Li et al., 2021] Li, Z., Niklaus, S., Snavely, N., and Wang, O. (2021). Neural scene flow fields for space-time view synthesis of dynamic scenes. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6498–6508.

[Li et al., 2022b] Li, Z., Song, L., Liu, C., Yuan, J., and Xu, Y. (2022b). Neulf: Efficient novel view synthesis with neural 4d light field. In Eurographics Symposium on Rendering.

[Liang et al., 2022] Liang, R., Sun, H., and Vijaykumar, N. (2022). Coordx: Accelerating implicit neural representation with a split mlp architecture. arXiv preprint arXiv:2201.12425.

[Lin et al., 2020] Lin, C.-H., Wang, C., and Lucey, S. (2020). Sdf-srn: Learning signed distance 3d object reconstruction from static images. In Advances in Neural Information Processing Systems (NeurIPS).

[Liu et al., 2020a] Liu, K., Li, Q., and Qiu, G. (2020a). PoseGAN: A pose-to-image translation framework for camera localization. In ISPRS Journal of Photogrammetry and Remote Sensing, volume 166, pages 308–315. Elsevier.

[Liu et al., 2020b] Liu, L., Gu, J., Lin, K. Z., Chua, T.-S., and Theobalt, C. (2020b). Neural sparse voxel fields. NeurIPS.

[Liu et al., 2018] Liu, R., Lehman, J., Molino, P., Such, F. P., Frank, E., Sergeev, A., and Yosinski, J. (2018). An intriguing failing of convolutional neural networks and the coordconv solution. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS'18, page 9628–9639, Red Hook, NY, USA. Curran Associates Inc.

[Liu et al., 2019] Liu, W., Liao, S., Ren, W., Hu, W., and Yu, Y. (2019). High-level semantic feature detection: A new perspective for pedestrian detection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 5187–5196.

[López et al., 2010] López, A., Serrat, J., Canero, C., Lumbreras, F., and Graf, T. (2010). Robust lane markings detection and road geometry computation. International Journal of Automotive Technology, 11(3):395–407.

[Lowe, 1999] Lowe, D. G. (1999). Object recognition from local scale-invariant features. In Proceedings of the seventh IEEE international conference on computer vision, volume 2, pages 1150–1157. Ieee.

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

[Lowry et al., 2015] 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.

[Lucas and Kanade, 1981] Lucas, B. D. and Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In International Joint Conference on Artificial Intelligence.

[Maddern et al., 2020] Maddern, W., Pascoe, G., Gadd, M., Barnes, D., Yeomans, B., and Newman, P. (2020). Real-time kinematic ground truth for the oxford robotcar dataset. In arXiv preprint arXiv: 2002.10152.

[Maddern et al., 2017] Maddern, W., Pascoe, G., Linegar, C., and Newman, P. (2017). 1 Year, 1000km: The Oxford RobotCar Dataset. The International Journal of Robotics Research (IJRR), 36:3–15.

[Mahmoud et al., 2023] Mahmoud, A., Hu, J. S., and Waslander, S. L. (2023). Dense voxel fusion for 3d object detection. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pages 663–672.

[Mai and Liu, 2022] Mai, L. and Liu, F. (2022). Motion-Adjustable Neural Implicit Video Representation. In CVPR.

[Markley et al., 2007] Markley, F. L., Cheng, Y., Crassidis, J. L., and Oshman, Y. (2007). Averaging quaternions. Journal of Guidance, Control, and Dynamics, 30(4):1193–1197.

[Martin-Brualla et al., 2021] Martin-Brualla, R., Radwan, N., Sajjadi, M. S. M., Barron, J. T., Dosovitskiy, A., and Duckworth, D. (2021). NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR).

[Massiceti et al., 2017] Massiceti, D., Krull, A., Brachmann, E., Rother, C., and Torr, P. H. (2017). Random forests versus neural networks — what's best for camera localization? In 2017 IEEE International Conference on Robotics and Automation (ICRA), pages 5118–5125.

[Max and Chen, 2005] Max, N. and Chen, M. (2005). Local and global illumination in the volume rendering integral. In Scientific Visualization: Advanced Concepts.

[Melekhov et al., 2017] Melekhov, I., Ylioinas, J., Kannala, J., and Rahtu, E. (2017). Image-based localization using hourglass networks. In 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), pages 870–877.

[Merriaux et al., 2014] Merriaux, P., Dupuis, Y., Vasseur, P., and Savatier, X. (2014). Wheel odometry-based car localization and tracking on vectorial map. In 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), pages 1890–1891.

[Mildenhall et al., 2019] Mildenhall, B., Srinivasan, P. P., Ortiz-Cayon, R., Kalantari, N. K., Ramamoorthi, R., Ng, R., and Kar, A. (2019). Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. ACM Transactions on Graphics (TOG).

[Mildenhall et al., 2020] Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. (2020). Nerf: Representing scenes as neural radiance fields for view synthesis. In ECCV.

[Moreau et al., 2023a] Moreau, A., Gilles, T., Piasco, N., Tsishkou, D., Stanciulescu, B., and de La Fortelle, A. (2023a). Imposing: Implicit pose encoding for efficient visual localization. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pages 2892–2902.

[Moreau et al., 2023b] Moreau, A., Piasco, N., Bennehar, M., Tsishkou, D., Stanciulescu, B., and de La Fortelle, A. (2023b). Crossfire: Camera relocalization on self-supervised features from an implicit representation. arXiv preprint arXiv:2303.04869.

[Moreau et al., 2022a] Moreau, A., Piasco, N., Tsishkou, D., Stanciulescu, B., and de La Fortelle, A. (2022a). Coordinet: uncertainty-aware pose regressor for reliable vehicle localization. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pages 2229–2238.

[Moreau et al., 2022b] Moreau, A., Piasco, N., Tsishkou, D., Stanciulescu, B., and de La Fortelle, A. (2022b). Lens: Localization enhanced by nerf synthesis. In Proceedings of the 5th Conference on Robot Learning, volume 164 of Proceedings of Machine Learning Research, pages 1347–1356. PMLR.

[Müller, 2021] Müller, T. (2021). tiny-cuda-nn.

[Müller et al., 2022] Müller, T., Evans, A., Schied, C., and Keller, A. (2022). Instant neural graphics primitives with a multiresolution hash encoding. ACM Trans. Graph., 41(4):102:1–102:15.

[Mur-Artal et al., 2015] Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D. (2015). Orb-slam: a versatile and accurate monocular slam system. IEEE transactions on robotics, 31(5):1147–1163.

[Nguyen et al., 2016] Nguyen, D.-V., Recalde, M. E. V., and Nashashibi, F. (2016). Low speed vehicle localization using wifi fingerprinting. In 2016 14th international conference on control, automation, robotics and vision (ICARCV), pages 1–5. IEEE.

[Niemeyer et al., 2022] Niemeyer, M., Barron, J. T., Mildenhall, B., Sajjadi, M. S., Geiger, A., and Radwan, N. (2022). Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5480–5490.

[Niemeyer and Geiger, 2021] Niemeyer, M. and Geiger, A. (2021). Giraffe: Representing scenes as compositional generative neural feature fields. In Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).

[Niemeyer et al., 2020] Niemeyer, M., Mescheder, L., Oechsle, M., and Geiger, A. (2020). Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).

[Nister and Stewenius, 2006] Nister, D. and Stewenius, H. (2006). Scalable recognition with a vocabulary tree. In 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), volume 2, pages 2161–2168.

[Novotny et al., 2018] Novotny, D., Albanie, S., Larlus, D., and Vedaldi, A. (2018). Self-supervised learning of geometrically stable features through probabilistic introspection. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), pages 3637–3645.

[Ortiz et al., 2022] Ortiz, J., Clegg, A., Dong, J., Sucar, E., Novotny, D., Zollhoefer, M., and Mukadam, M. (2022). isdf: Real-time neural signed distance fields for robot perception. arXiv preprint arXiv:2204.02296.

[Ost et al., 2022] Ost, J., Laradji, I., Newell, A., Bahat, Y., and Heide, F. (2022). Neural point light fields. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 18419–18429.

[Ost et al., 2021] Ost, J., Mannan, F., Thuerey, N., Knodt, J., and Heide, F. (2021). Neural scene graphs for dynamic scenes. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 2856–2865.

[Panek et al., 2022] Panek, V., Kukelova, Z., and Sattler, T. (2022). MeshLoc: Mesh-Based Visual Localization. In Proceedings of the European Conference on Computer Vision (ECCV).

[Park et al., 2019] Park, J. J., Florence, P., Straub, J., Newcombe, R., and Lovegrove, S. (2019). Deepsdf: Learning continuous signed distance functions for shape representation. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[Park et al., 2021] Park, K., Sinha, U., Barron, J. T., Bouaziz, S., Goldman, D. B., Seitz, S. M., and Martin-Brualla, R. (2021). Nerfies: Deformable neural radiance fields. ICCV.

[Penatti et al., 2014] Penatti, O. A., Silva, F. B., Valle, E., Gouet-Brunet, V., and Torres, R. d. S. (2014). Visual word spatial arrangement for image retrieval and classification. Pattern Recognition, 47(2):705–720.

[Peng et al., 2021] Peng, S., He, Z., Zhang, H., Yan, R., Wang, C., Zhu, Q., and Liu, X. (2021). Megloc: A robust and accurate visual localization pipeline.

[Peretroukhin et al., 2019] Peretroukhin, V., Wagstaff, B., Giamou, M., and Kelly, J. (2019). Probabilistic regression of rotations using quaternion averaging and a deep multi-headed network. arXiv preprint arXiv:1904.03182.

[Phillips et al., 2021] Phillips, J., Martinez, J., Barsan, I. A., Casas, S., Sadat, A., and Urtasun, R. (2021). Deep multi-task learning for joint localization, perception, and prediction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 4679–4689.

[Piasco et al., 2018] Piasco, N., Sidibé, D., Demonceaux, C., and Gouet-Brunet, V. (2018). A survey on visual-based localization: On the benefit of heterogeneous data. Pattern Recognition, 74:90–109.

[Piasco et al., 2019a] Piasco, N., Sidibé, D., Demonceaux, C., and Gouet-Brunet, V. (2019a). Perspective-n-learned-point: Pose estimation from relative depth. In British machine vision conference (BMVC).

[Piasco et al., 2021] Piasco, N., Sidibé, D., Gouet-Brunet, V., and Demonceaux, C. (2021). Improving image description with auxiliary modality for visual localization in challenging conditions. International Journal of Computer Vision, 129:185–202.

[Piasco et al., 2019b] Piasco, N., Sidibé, D., Gouet-Brunet, V., and Demonceaux, C. (2019b). Learning scene geometry for visual localization in challenging conditions. In 2019 International Conference on Robotics and Automation (ICRA), pages 9094–9100.

[Pronobis et al., 2006] Pronobis, A., Caputo, B., Jensfelt, P., and Christensen, H. (2006). A discriminative approach to robust visual place recognition. In 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 3829–3836.

[Pumarola et al., 2020] Pumarola, A., Corona, E., Pons-Moll, G., and Moreno-Noguer, F. (2020). D-NeRF: Neural Radiance Fields for Dynamic Scenes. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.

[Purkait et al., 2018] Purkait, P., Zhao, C., and Zach, C. (2018). Synthetic view generation for absolute pose regression and image synthesis. In Proceedings of the British Machine Vision Conference, page 69.

[Qin et al., 2021] Qin, T., Zheng, Y., Chen, T., Chen, Y., and Su, Q. (2021). A light-weight semantic map for visual localization towards autonomous driving. In 2021 IEEE International Conference on Robotics and Automation (ICRA), pages 11248–11254.

[Quei-An, 2020] Quei-An, C. (2020). Nerf_pl: a pytorch-lightning implementation of nerf.

[Reiser et al., 2021] Reiser, C., Peng, S., Liao, Y., and Geiger, A. (2021). Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pages 14315–14325.

[Rematas et al., 2022] Rematas, K., Liu, A., Srinivasan, P. P., Barron, J. T., Tagliasacchi, A., Funkhouser, T., and Ferrari, V. (2022). Urban radiance fields. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 12932–12942.

[Revaud et al., 2019] Revaud, J., Almazan, J., Rezende, R., and de Souza, C. (2019). Learning with average precision: Training image retrieval with a listwise loss. In ICCV.

[Robins, 1995] Robins, A. (1995). Catastrophic forgetting, rehearsal and pseudorehearsal. Connection Science, 7(2):123–146.

[Roldao et al., 2022] Roldao, L., De Charette, R., and Verroust-Blondet, A. (2022). 3d semantic scene completion: A survey. International Journal of Computer Vision, 130(8):1978–2005.

[Rosten and Drummond, 2006] Rosten, E. and Drummond, T. (2006). Machine learning for high-speed corner detection. In European conference on computer vision, pages 430–443. Springer.

[Russell and Reale, 2021] Russell, R. L. and Reale, C. (2021). Multivariate uncertainty in deep learning. IEEE Transactions on Neural Networks and Learning Systems, 33(12):7937–7943.

[Saha et al., 2018] Saha, S., Varma, G., and Jawahar, C. (2018). Improved visual relocalization by discovering anchor points. arXiv preprint arXiv:1811.04370.

[Sandler et al., 2018] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4510–4520.

[Sarlin et al., 2019] Sarlin, P.-E., Cadena, C., Siegwart, R., and Dymczyk, M. (2019). From coarse to fine: Robust hierarchical localization at large scale. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR).

[Sarlin et al., 2020] Sarlin, P.-E., DeTone, D., Malisiewicz, T., and Rabinovich, A. (2020). SuperGlue: Learning feature matching with graph neural networks. In CVPR.

[Sarlin et al., 2021] Sarlin, P.-E., Unagar, A., Larsson, M., Germain, H., Toft, C., Larsson, V., Pollefeys, M., Lepetit, V., Hammarstrand, L., Kahl, F., and Sattler, T. (2021). Back to the Feature: Learning robust camera localization from pixels to pose. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR).

[Sattler et al., 2015] Sattler, T., Havlena, M., Radenovic, F., Schindler, K., and Pollefeys, M. (2015). Hyperpoints and fine vocabularies for large-scale location recognition. In 2015 IEEE International Conference on Computer Vision (ICCV), pages 2102–2110.

[Sattler et al., 2016] Sattler, T., Havlena, M., Schindler, K., and Pollefeys, M. (2016). Large-scale location recognition and the geometric burstiness problem. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1582–1590.

[Sattler et al., 2012] Sattler, T., Leibe, B., and Kobbelt, L. (2012). Improving image-based localization by active correspondence search. In Proceedings of the European Conference on Computer Vision (ECCV).

[Sattler et al., 2018] Sattler, T., Maddern, W., Toft, C., Torii, A., Hammarstrand, L., Stenborg, E., Safari, D., Okutomi, M., Pollefeys, M., Sivic, J., Kahl, F., and Pajdla, T. (2018). Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pages 8601–8610.

[Sattler et al., 2019] Sattler, T., Zhou, Q., Pollefeys, M., and Leal-taix, L. (2019). Understanding the Limitations of CNN-based Absolute Camera Pose Regression Chalmers University of Technology. In Cvpr, pages 3302–3312.

[Schmid et al., 2000] Schmid, C., Mohr, R., and Bauckhage, C. (2000). Evaluation of interest point detectors. International Journal of computer vision, 37(2):151–172.

[Schönberger and Frahm, 2016] Schönberger, J. L. and Frahm, J.-M. (2016). Structure-from-motion revisited. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR).

[Shavit et al., 2021a] Shavit, Y., Ferens, R., and Keller, Y. (2021a). Learning multi-scene absolute pose regression with transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 2733–2742.

[Shavit et al., 2021b] Shavit, Y., Ferens, R., and Keller, Y. (2021b). Paying attention to activation maps in camera pose regression. In arxiv preprint, arxiv:2103.11477.

[Shotton et al., 2013] Shotton, J., Glocker, B., Zach, C., Izadi, S., Criminisi, A., and Fitzgibbon, A. (2013). Scene coordinate regression forests for camera relocalization in rgb-d images. In Proc. Computer Vision and Pattern Recognition (CVPR). IEEE.

[Simo-Serra et al., 2015] Simo-Serra, E., Trulls, E., Ferraz, L., Kokkinos, I., Fua, P., and Moreno-Noguer, F. (2015). Discriminative learning of deep convolutional feature point descriptors. In Proceedings of the International Conference on Computer Vision (ICCV), pages 118–126.

[Sitzmann et al., 2020] Sitzmann, V., Martel, J. N., Bergman, A. W., Lindell, D. B., and Wetzstein, G. (2020). Implicit neural representations with periodic activation functions. In Proc. NeurIPS.

[Sitzmann et al., 2021] Sitzmann, V., Rezchikov, S., Freeman, W. T., Tenenbaum, J. B., and Durand, F. (2021). Light field networks: Neural scene representations with single-evaluation rendering. In Proc. NeurIPS.

[Sitzmann et al., 2019] Sitzmann, V., Zollhöfer, M., and Wetzstein, G. (2019). Scene representation networks: Continuous 3d-structure-aware neural scene representations. In Advances in Neural Information Processing Systems.

[Sivic and Zisserman, 2009] Sivic, J. and Zisserman, A. (2009). Efficient visual search of videos cast as text retrieval. IEEE Trans. Pattern Anal. Mach. Intell., 31(4):591–606.

[Snavely et al., 2006] Snavely, N., Seitz, S. M., and Szeliski, R. (2006). Photo tourism: Exploring photo collections in 3d. In SIGGRAPH Conference Proceedings, pages 835–846, New York, NY, USA. ACM Press.

[Song et al., 2022] Song, Y., Zhu, R., Yang, M., and He, D. (2022). Dalg: Deep attentive local and global modeling for image retrieval. arXiv preprint arXiv:2207.00287.

[Strümpler et al., 2022] Strümpler, Y., Postels, J., Yang, R., Gool, L. V., and Tombari, F. (2022). Implicit neural representations for image compression. In European Conference on Computer Vision, pages 74–91. Springer.

[Sucar et al., 2021] Sucar, E., Liu, S., Ortiz, J., and Davison, A. (2021). iMAP: Implicit mapping and positioning in real-time. In Proceedings of the International Conference on Computer Vision (ICCV).

[Sumikura et al., 2019] Sumikura, S., Shibuya, M., and Sakurada, K. (2019). OpenVSLAM: A Versatile Visual SLAM Framework. In Proceedings of the 27th ACM International Conference on Multimedia, MM '19, New York, NY, USA. ACM.

[Sun et al., 2022] Sun, C., Sun, M., and Chen, H. (2022). Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In CVPR.

[Szatkowski et al., 2022] Szatkowski, F., Piczak, K. J., Spurek, P., Tabor, J., and Trzciński, T. (2022). Hypersound: Generating implicit neural representations of audio signals with hypernetworks. arXiv preprint arXiv:2211.01839.

[Szegedy et al., 2015] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[Tabernik and Skočaj, 2019] Tabernik, D. and Skočaj, D. (2019). Deep learning for large-scale traffic-sign detection and recognition. IEEE transactions on intelligent transportation systems, 21(4):1427–1440.

[Tagliasacchi and Mildenhall, 2022] Tagliasacchi, A. and Mildenhall, B. (2022). Volume rendering digest (for nerf). arXiv preprint arXiv:2209.02417.

[Taira et al., 2018] Taira, H., Okutomi, M., Sattler, T., Cimpoi, M., Pollefeys, M., Sivic, J., Pajdla, T., and Torii, A. (2018). InLoc: Indoor visual localization with dense matching and view synthesis. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 7199–7209.

[Tan and Le, 2019] Tan, M. and Le, Q. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning, pages 6105–6114. PMLR.

[Tancik et al., 2022] Tancik, M., Casser, V., Yan, X., Pradhan, S., Mildenhall, B., Srinivasan, P. P., Barron, J. T., and Kretzschmar, H. (2022). Block-nerf: Scalable large scene neural view synthesis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 8248–8258.

[Tancik et al., 2020] Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., and Ng, R. (2020). Fourier features let networks learn high frequency functions in low dimensional domains. Advances in Neural Information Processing Systems, 33:7537–7547.

[Tang et al., 2018] Tang, J., Folkesson, J., and Jensfelt, P. (2018). Geometric correspondence network for camera motion estimation. IEEE Robotics and Automation Letters, 3(2):1010–1017.

[Tang et al., 2022] Tang, S., Tang, S., Tagliasacchi, A., Tan, P., and Furukawa, Y. (2022). Neumap: Neural coordinate mapping by auto-transdecoder for camera localization. arXiv preprint arXiv:2211.11177.

[Tateno et al., 2017] Tateno, K., Tombari, F., Laina, I., and Navab, N. (2017). Cnn-slam: Real-time dense monocular slam with learned depth prediction. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 6243–6252.

[Tian et al., 2019] Tian, Y., Yu, X., Fan, B., Wu, F., Heijnen, H., and Balntas, V. (2019). Sosnet: Second order similarity regularization for local descriptor learning. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), pages 11016–11025.

[Toker et al., 2021] Toker, A., Zhou, Q., Maximov, M., and Leal-Taixé, L. (2021). Coming down to earth: Satellite-to-street view synthesis for geo-localization. In arXiv preprint arXiv:2103.06818.

[Torii et al., 2015a] Torii, A., Arandjelović, R., Sivic, J., Okutomi, M., and Pajdla, T. (2015a). 24/7 place recognition by view synthesis. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), pages 1808–1817.

[Torii et al., 2015b] Torii, A., Arandjelović, R., Sivic, J., Okutomi, M., and Pajdla, T. (2015b). 24/7 place recognition by view synthesis. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1808–1817.

[Torii et al., 2011] Torii, A., Sivic, J., and Pajdla, T. (2011). Visual localization by linear combination of image descriptors. In 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops), pages 102–109. IEEE.

[Tretschk et al., 2021] Tretschk, E., Tewari, A., Golyanik, V., Zollhöfer, M., Lassner, C., and Theobalt, C. (2021). Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In IEEE International Conference on Computer Vision (ICCV). IEEE.

[Truong et al., 2020] Truong, P., Danelljan, M., Gool, L. V., and Timofte, R. (2020). GOCor: Bringing globally optimized correspondence volumes into your neural network. In Proceedings of Advances in Neural Information Processing Systems (NeurIPS.

[Verdie et al., 2015] Verdie, Y., Yi, K., Fua, P., and Lepetit, V. (2015). Tilde: A temporally invariant learned detector. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 5279–5288.

[Von Stumberg et al., 2020] Von Stumberg, L., Wenzel, P., Khan, Q., and Cremers, D. (2020). GN-Net: The gauss-newton loss for multi-weather relocalization. IEEE Robotics and Automation Letters, 5(2):890–897.

[Waechter et al., 2014] Waechter, M., Moehrle, N., and Goesele, M. (2014). Let there be color! large-scale texturing of 3d reconstructions. In European Conference on Computer Vision.

[Walch et al., 2016] Walch, F., Hazirbas, C., Leal-Taixé, L., Sattler, T., Hilsenbeck, S., and Cremers, D. (2016). Image-based localization using lstms for structured feature correlation. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 627–637.

[Wang et al., 2020a] Wang, B., Chen, C., Xiaoxuan Lu, C., Zhao, P., Trigoni, N., and Markham, A. (2020a). Atloc: Attention guided camera localization. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pages 10393–10401.

[Wang et al., 2022] Wang, H., Ren, J., Huang, Z., Olszewski, K., Chai, M., Fu, Y., and Tulyakov, S. (2022). R2l: Distilling neural radiance field to neural light field for efficient novel view synthesis. In European Conference on Computer Vision.

[Wang et al., 2020b] Wang, Q., Zhou, X., Hariharan, B., and Snavely, N. (2020b). Learning feature descriptors using camera pose supervision. In Proceedings of the European Conference on Computer Vision (ECCV).

[Wang et al., 2021] Wang, S., Laskar, Z., Melekhov, I., Li, X., and Kannala, J. (2021). Continual learning for image-based camera localization. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 3252–3262.

[Wang et al., 2004] Wang, Z., Bovik, A., Sheikh, H., and Simoncelli, E. (2004). Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing, 13(4):600–612.

[Ward and Folkesson, 2016] Ward, E. and Folkesson, J. (2016). Vehicle localization with low cost radar sensors. In 2016 IEEE Intelligent Vehicles Symposium (IV), pages 864–870. IEEE.

[Wen et al., 2022] Wen, Y., Li, X., Pan, H., Yang, L., Wang, Z., Komura, T., and Wang, W. (2022). Disp6d: Disentangled implicit shape and pose learning for scalable 6d pose estimation. In European Conference on Computer Vision (ECCV).

[Weng et al., 2021] Weng, L., Gouet-Brunet, V., and Soheilian, B. (2021). Semantic signatures for large-scale visual localization. Multimedia Tools and Applications, 80:22347–22372.

[Wenzel et al., 2020] Wenzel, P., Wang, R., Yang, N., Cheng, Q., Khan, Q., von Stumberg, L., Zeller, N., and Cremers, D. (2020). 4Seasons: A cross-season dataset for multi-weather SLAM in autonomous driving. In Proceedings of the German Conference on Pattern Recognition (GCPR).

[Wood et al., 2000] Wood, D. N., Azuma, D. I., Aldinger, K. R., Curless, B., Duchamp, T., Salesin, D., and Stuetzle, W. (2000). Surface light fields for 3d photography. Proceedings of the 27th annual conference on Computer graphics and interactive techniques.

[Wu, 2013] Wu, C. (2013). Towards linear-time incremental structure from motion. In 2013 International Conference on 3D Vision - 3DV 2013, pages 127–134.

[Xie et al., 2021a] Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., and Luo, P. (2021a). Segformer: Simple and efficient design for semantic segmentation with transformers. Advances in Neural Information Processing Systems, 34:12077–12090.

[Xie et al., 2021b] Xie, Y., Takikawa, T., Saito, S., Litany, O., Yan, S., Khan, N., Tombari, F., Tompkin, J., Sitzmann, V., and Sridhar, S. (2021b). Neural fields in visual computing and beyond. In Eurographics STAR.

[Xue et al., 2020] Xue, F., Wu, X., Cai, S., and Wang, J. (2020). Learning multi-view camera relocalization with graph neural networks. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 11372–11381.

[Yang et al., 2021] Yang, B., Zhang, Y., Xu, Y., Li, Y., Zhou, H., Bao, H., Zhang, G., and Cui, Z. (2021). Learning object-compositional neural radiance field for editable scene rendering. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pages 13779–13788.

[Yang et al., 2020] Yang, N., Stumberg, L. v., Wang, R., and Cremers, D. (2020). D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 1281–1292.

[Yao et al., 2018] Yao, Y., Luo, Z., Li, S., Fang, T., and Quan, L. (2018). Mvsnet: Depth inference for unstructured multi-view stereo. European Conference on Computer Vision (ECCV).

[Yao et al., 2019] Yao, Y., Luo, Z., Li, S., Shen, T., Fang, T., and Quan, L. (2019). Recurrent mvsnet for high-resolution multi-view stereo depth inference. Computer Vision and Pattern Recognition (CVPR).

[Yen-Chen et al., 2021] Yen-Chen, L., Florence, P., Barron, J. T., Rodriguez, A., Isola, P., and Lin, T.-Y. (2021). iNeRF: Inverting neural radiance fields for pose estimation. In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

[Yi et al., 2007] Yi, J., Zhang, J., Song, D., and Jayasuriya, S. (2007). Imu-based localization and slip estimation for skid-steered mobile robots. In 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 2845–2850. IEEE.

[Yousif et al., 2015] Yousif, K., Bab-Hadiashar, A., and Hoseinnezhad, R. (2015). An overview to visual odometry and visual slam: Applications to mobile robotics. Intelligent Industrial Systems, 1(4):289–311.

[Yu et al., 2021] Yu, A., Li, R., Tancik, M., Li, H., Ng, R., and Kanazawa, A. (2021). PlenOctrees for real-time rendering of neural radiance fields. In ICCV.

[Zagoruyko and Komodakis, 2015] Zagoruyko, S. and Komodakis, N. (2015). Learning to compare image patches via convolutional neural networks. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), pages 4353–4361.

[Zakharov et al., 2021] Zakharov, S., Ambrus, R., Guizilini, V. C., Park, D., Kehl, W., Durand, F., Tenenbaum, J. B., Sitzmann, V., Wu, J., and Gaidon, A. (2021). Single-shot scene reconstruction. In Conference on Robot Learning.

[Zakharov et al., 2022] Zakharov, S., Ambrus, R., Liu, K., and Gaidon, A. (2022). Road: Learning an implicit recursive octree auto-decoder to efficiently encode 3d shapes. In Conference on Robot Learning (CoRL).

[Zakharov et al., 2020] Zakharov, S., Kehl, W., Bhargava, A., and Gaidon, A. (2020). Autolabeling 3d objects with differentiable rendering of sdf shape priors. In IEEE Computer Vision and Pattern Recognition (CVPR).

[Zaklouta and Stanciulescu, 2014] Zaklouta, F. and Stanciulescu, B. (2014). Real-time traffic sign recognition in three stages. Robotics and autonomous systems, 62(1):16–24.

[Zangeneh et al., 2023] Zangeneh, F., Bruns, L., Dekel, A., Pieropan, A., and Jensfelt, P. (2023). A probabilistic framework for visual localization in ambiguous scenes. In Proceedings of the IEEE International Conference on Robotics and Automation.

[Zhang et al., 2020a] Zhang, K., Riegler, G., Snavely, N., and Koltun, V. (2020a). Nerf++: Analyzing and improving neural radiance fields. arXiv:2010.07492.

[Zhang et al., 2020b] Zhang, Z., Sattler, T., and Scaramuzza, D. (2020b). Reference pose generation for long-term visual localization via learned features and view synthesis. In International Journal of Computer Vision, volume 129, pages 821–844.

[Zhao et al., 2021] Zhao, H., Gao, J., Lan, T., Sun, C., Sapp, B., Varadarajan, B., Shen, Y., Shen, Y., Chai, Y., Schmid, C., et al. (2021). Tnt: Target-driven trajectory prediction. In Conference on Robot Learning, pages 895–904. PMLR.

[Zhi et al., 2023] Zhi, S., Sucar, E., Mouton, A., Haughton, I., Laidlow, T., and Davison, A. J. (2023). ilabel: Revealing objects in neural fields. IEEE Robotics and Automation Letters, 8(2):832–839.

[Zhou et al., 2023] Zhou, A., Kim, M. J., Wang, L., Florence, P., and Finn, C. (2023). Nerf in the palm of your hand: Corrective augmentation for robotics via novel-view synthesis. In arXiv:2301.08556.

[Zhou et al., 2019] Zhou, Y., Barnes, C., Jingwan, L., Jimei, Y., and Hao, L. (2019). On the continuity of rotation representations in neural networks. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[Zhou et al., 2020] Zhou, Y., Wan, G., Hou, S., Yu, L., Wang, G., Rui, X., and Song, S. (2020). Da4ad: End-to-end deep attention-based visual localization for autonomous driving. In Proceedings of the European Conference on Computer Vision (ECCV).

[Zhu et al., 2021] Zhu, Y., Gao, R., Huang, S., Zhu, S., and Wu, Y. (2021). Learning neural representation of camera pose with matrix representation of pose shift via view synthesis. In 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 9954–9963, Los Alamitos, CA, USA. IEEE Computer Society.

[Zhu et al., 2022] Zhu, Z., Peng, S., Larsson, V., Xu, W., Bao, H., Cui, Z., Oswald, M. R., and Pollefeys, M. (2022). Nice-slam: Neural implicit scalable encoding for slam. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), pages 12786–12796.

## RÉSUMÉ

La conduite autonome est appelée à révolutionner les transports de demain. Les systèmes de localisation sont un élément clé des véhicules autonomes afin d'assurer une navigation sûre, fluide et fiable. La position d'un véhicule dans son environnement peut être déterminée à l'aide de différents capteurs, utiliser l'image permet de se localiser plus précisement qu'un GPS et requiert uniquement une caméra. Cette thèse vise à résoudre le problème de la relocalisation à l'aide de méthodes d'apprentissage automatique. Nous nous appuyons sur les avancées récentes dans le domaine de l'apprentissage profond pour développer des algorithmes qui apprennent à localiser la position de la caméra à partir d'une grande base de données d'images recueillies dans la zone d'intérêt. Tout d'abord, nous étudions la capacité des réseaux de neurones convolutionnels à être utilisées comme système de localisation de véhicules dans des scénarios de conduite autonome. Dans un second temps, nous avons exploré le lien entre les représentations implicites de scènes et les algorithmes de localisation visuelle. Ces représentations implicites génerent des images synthétiques utilisées pour entraîner de meilleurs algorithmes, mais aussi pour représenter la carte de l'environnement. Nous montrons que les informations pertinentes capturées sur des routes de plusieurs kilomètres peuvent être encodées en quelques mégabytes afin de réaliser la localisation de véhicules en temps réel. De plus, nous remplaçons les modèles 3D traditionnels par un Neural Radiance Field (NeRF) dans les méthodes de "features matching". Globalement, ce travail réhabilite les méthodes de regression, qui sont considérées comme moins précises que les méthodes classiques basées sur les features. Au final, l'efficacité des méthodes d'apprentissages dépend des données et peut donc être bénéfique dans certaines situations, comme la conduite autonome.

## MOTS CLÉS

Localisation basée image, Apprentissage automatique, Vision par ordinateur

## ABSTRACT

Autonomous driving is expected to revolutionize tomorrow's transportation technologies. Positioning systems are a key component of self-driving vehicles in order to ensure a safe, smooth and reliable navigation. The precise ego position of a vehicle inside of its environment can be recovered by a wide range of sensors and algorithms, image-based localization is more acurate than GNSS and only needs camera sensors. This thesis aims to solve the map-based relocalization problem with machine learning methods. We build on recent advances on the deep learning area to develop algorithms which learn to relocalize from a large collection of images gathered in the area of interest. First, we study the capability of convolutional neural networks to be used as a vehicle localization system in autonomous driving scenarios. Then, we investigated in different ways the connection between implicit scene representations and visual localization algorithms. By their ability to represent continuously a complex scene in a neural network, these implicit representations can be used to generate photo-realistic synthetic data used to train better algorithms, but also to represent the map of the environment. We show that relevant information captured on roads of several kilometers can be encoded in few megabytes in order to achieve real-time vehicle localization, but also that local features can be learned, stored and rendered by a Neural Field to achieve centimeter-level camera pose estimation in a dense features matching pipeline. Overall, this work aims to rehabilitate direct learning-based formulations, which are considered to be less precise than classical features-based methods. In the end, the effectiveness of data-driven methods depends on the data and then can be beneficial in some situations, such as autonomous driving.

## KEYWORDS

Visual localization, Machine learning, Computer vision