3D geometry-based neural camera pose estimation

Hugo Germain

École des Ponts ParisTech

THÈSE DE DOCTORAT
de l'École des Ponts ParisTech

3D GEOMETRY-BASED NEURAL CAMERA POSE ESTIMATION

ED MSTIC n°532

Doctorat en Informatique

Thèse préparée au sein du laboratoire IMAGINE, LIGM

Thèse soutenue le 10/12/2022, par
Hugo GERMAIN

Composition du jury :

Josef, SIVIC Directeur de recherche, INRIA Paris	*Président*
Ondrej, CHUM Professeur associé, Czech Technical University	*Rapporteur*
Krystian, MIKOLAJCZYK Professeur, Imperial College London	*Rapporteur*
Gabriela, CSURKA Directrice de recherche, Naver Labs Europe	*Examinateur*
Torsten, SATTLER Directeur de recherche, Czech Technical University	*Examinateur*
Vincent, LEPETIT Directeur de recherche, École des Ponts ParisTech	*Directeur de thèse*
Guillaume, BOURMAUD Professeur associé, Université de Bordeaux	*Co-Directeur de thèse*

École Doctorale Paris-Est
Mathématiques & Sciences et Technologies
de l'Information et de la Communication

3D Geometry-based Neural Camera Pose Estimation

Hugo GERMAIN

Doctoral thesis in the domain of
signal and image processing
supervised by Vincent LEPETIT
and Guillaume BOURMAUD

Presented on 10/12/2021 to a committee consisting of:

Vincent LEPETIT	École des Ponts ParisTech	Supervisor
Guillaume BOURMAUD	Université de Bordeaux	Co-Supervisor
Krystian MIKOLAJCZYK	Imperial College London	Reviewer
Ondrej CHUM	Czech Technical University	Reviewer
Torsten SATTLER	Czech Technical University	Examiner
Gabriela CSURKA KHEDARI	Naver Labs Europe	Examiner
Josef SIVIC	INRIA Paris	Examiner

LIGM, IMAGINE,
École des Ponts ParisTech,
Université Gustave Eiffel,
ESIEE Paris, CNRS

Université Paris-Est Sup
École Doctorale Paris-Est MSTIC
Département Études Doctorales

6, Avenue Blaise Pascal - Cité Descartes
Champs-sur-Marne
77455 Marne-la-Vallée Cedex 2
France

Abstract

Vision-based absolute camera pose estimation, also known as visual localization, is an underpinning backbone to many computer vision applications, such as augmented or virtual reality, robotics and autonomous driving. When working with crowd-sourced images captured under challenging conditions, visual disturbances are frequently encountered. These perturbations make visual localization a very hard -and so far unsolved- problem. The goal of this thesis is to develop models that can improve the performance of absolute camera pose algorithms. The first part of this thesis focuses on the task of matching 2D keypoints against a 3D model, which is a commonly used building block to structure-based visual localization approaches. We propose a novel keypoint matching paradigm which explicitly models dense keypoint matching uncertainties in images, and finds it improves over state-of-the-art keypoint matching methods. Then, we introduce a novel reprojection error to merge feature learning and absolute camera pose estimation, which we call the Neural Reprojection Error. Our formulation reuses the previously introduced dense matching uncertainties to significantly improve the camera pose estimation accuracy, compared to standard approaches. This formulation is data-driven and thus helps us avoid cumbersome hyperparameter optimization. The last contribution of this thesis is to study the problem of visual correspondence hallucination. We train a deep learning model to regress matching distributions in non-covisible image areas (i.e. that are either occluded or fall outside of the image boundaries). We show our model is not only able to make such predictions, but that when coupled with the Neural Reprojection Error it significantly outperforms existing absolute camera pose estimation methods, when presented with very low-overlap image pairs.

Keywords: Visual localization, image matching, absolute camera pose estimation, optimization

Résumé

L'estimation de pose absolue de caméra basée sur la vision, également connue sous le nom de localisation visuelle, est l'épine dorsale de nombreuses applications de vision par ordinateur, telles que la réalité augmentée ou virtuelle, la robotique ou la conduite autonome. Lorsque l'on travaille avec des images naturelles capturées dans des conditions changeantes, on encontre fréquemment des perturbations visuelles. Ces perturbations font de la localisation visuelle un problème très difficile - et jusqu'à présent non résolu. L'objectif de cette thèse est de développer des modèles pouvant améliorer la performance des algorithmes d'estimation de pose absolue de caméra. La première partie de cette thèse se concentre sur la mise en correspondance de points d'intérêt 2D avec un modèle 3D, qui est un élément communément utilisé dans les approches de localisation visuelle basées sur la géométrie 3D. Nous présentons un nouveau paradigme d'appariement de points d'intérêt qui modélise explicitement les incertitudes de mise en correspondance de manière dense dans les images. Nos expériences montrent que cette approche permet d'améliorer l'état de l'art en estimation de pose absolue de caméra. Puis, nous introduisons une nouvelle erreur de reprojection pour fusionner l'apprentissage des caractéristiques d'une image et l'estimation de la pose absolue de la caméra, appelée "Neural Reprojection Error". Notre formulation réutilise les incertitudes d'appariement dense introduites précédemment pour améliorer la précision de l'estimation de la pose, en comparaison aux approches standard. Cette formulation a l'avantage d'être basée sur les données d'apprentissage uniquement, et nous permet d'éviter une optimisation fastidieuse des hyperparamètres. La dernière contribution de cette thèse consiste à étudier le problème de l'hallucination de correspondance visuelle. Nous entraînons un réseau de neurone profond pour prédire des distributions de correspondance dans des zones d'image non co-visibles (i.e. qui sont soit occultées, soit en dehors des limites de l'image). Nos expériences démontrent que notre modèle est non seulement capable de faire de telles prédictions, mais que lorsqu'il est couplé à la "Neural Reprojection Error", il surpasse de manière significative les méthodes existantes d'estimation de pose absolue de caméra sur des paires d'images à très faible recouvrement.

Mots-clés: Localisation visuelle, appariement de points d'intérêt, estimation absolue de pose de caméra, optimisation

Remerciements

Je souhaite tout d'abord remercier Vincent et Guillaume pour m'avoir fait confiance dès le premier jour de cette thèse. Votre expérience, bienveillance et vos précieux conseils m'ont permis d'explorer avec sérénité ce domaine de recherche. Merci pour votre grande disponibilité et merci de m'avoir transmis votre passion pour la recherche et la rigueur scientifique, ce fut un plaisir et un honneur de travailler avec vous. Toutes ces heures de discussions ont été formatrices et fructueuses et nous ont permis j'espère, d'élargir nos horizons vers des idées nouvelles et audacieuses.

Je voudrais ensuite remercier mes premiers compagnons de thèse Giorgia et Michaël, avec qui nous avons partagé joies, doutes, et frustrations, et finalement cheminé depuis Bordeaux vers Paris. Merci à la grande famille d'Imagine pour leur accueil à bras ouverts, et avec qui j'espère rester en contact pour les années à venir.

Un grand merci à Pierre pour avoir coordonné de nombreuses discussions dans l'ombre, et avoir fait en sorte que je n'ai eu besoin de me soucier de rien d'autre que mes travaux de recherche. Je tiens également à remercier Klaus Dieterich et la fondation de recherche Bosch Forschungsstiftung pour leur confiance dans ce projet de thèse.

Merci à Torsten de m'avoir patiemment aidé durant les premiers mois de ma thèse. Merci aussi à Vasileios pour sa collaboration fructueuse et sa bénévolence.

Je tiens à remercier ma famille, pour le support qu'ils m'ont apporté tout au long de cette thèse et plus généralement de mon parcours scolaire. Merci de m'avoir toujours encouragé à poursuivre mes passions.

Enfin, je remercie tout particulièrement Tiphaine d'avoir été à mes côté du début à la fin de ce voyage. Merci pour ta présence, ton écoute et ton soutien indéfectible.

Contents

Nomenclature

Camera Variables

$\mathbf{p}$	A 2D point coordinate, in $\mathbb{R}^2$
$\mathbf{u}$	A 3D point coordinate, in $\mathbb{R}^3$
$\mathbf{t}$	The camera translation vector, in $\mathbb{R}^3$
$\mathtt{R}$	The camera rotation matrix, in SO(3)
$\mathtt{K}$	The camera linear calibration matrix, in $\mathbb{R}^{3\times3}$
$\mathtt{I}$	An image, in $\mathbb{R}^{H\times W\times3}$

Ensembles

$\mathcal{M}$	A 3D model $\{\mathbf{u}_i\}_{i=1}^{M}$
$\mathcal{U}$	The set of 2D-to-3D correspondences (e.g. produced by keypoint matching)
$\mathcal{I}$	The set of inlier correspondences (i.e. after RANSAC), such that $\mathcal{I}\subset\mathcal{U}$

Tensors

$\mathbf{h}$	A sparse keypoint descriptor, in $\mathbb{R}^{1\times1\times D}$
$\mathtt{H}$	A dense feature map, in $\mathbb{R}^{H_\mathtt{H}\times W_\mathtt{H}\times D}$
$\breve{\mathtt{C}}$	A feature correlation map, in $\mathbb{R}^{H_\mathtt{H}\times W_\mathtt{H}}$
$\mathtt{C}$	A correspondence map, in $\mathbb{R}^{H_\mathtt{H}\times W_\mathtt{H}}$
$\tilde{\mathtt{C}}$	The negative-log of a correspondence map, in $\mathbb{R}^{H_\mathtt{H}\times W_\mathtt{H}}$
$\mathtt{L}$	A dense loss map, in $\mathbb{R}^{H_\mathtt{H}\times W_\mathtt{H}}$

Chapter 1

Introduction

1.1 Goals

The goal of this thesis is to develop deep learning-based methods that improve the performance of visual localization algorithms. As will be presented further down, we will focus more specifically on the problem of *structure-based* localization, that leverages an existing 3D representation of the world to estimate the camera pose of a newly captured image. In practice, this objective can be split into two sub-goals: (*i*) improve the performance of 2D-to-3D keypoint matching algorithms to better handle strong visual perturbations and (*ii*) rethink the existing absolute camera pose estimators to better account for keypoint matching uncertainties.

2D-to-3D matching is the task of finding local correspondences between a 2D image and a 3D model. This problem is often highly related to the task of finding matches between an image (referred to as a *query* image) and the 2D reprojection of the 3D model inside a nearby image for which the camera pose and parameters are known (referred to as *reference* images). 2D-to-3D matching can be thus be treated as a 2D-to-2D problem between query and reference images, in a problem known as *keypoint matching* (see Figure 1.1). This task is made difficult in practice by the strong visual perturbations that occur in long-term scenarios. Due to changes in illumination, weather or scene geometry, crowd-sourced images captured over extended periods of time can be particularly challenging to match. In this thesis, we propose a novel paradigm that aims at better modeling the errors and uncertainties made by learning-based keypoint matching methods. In Chapter 3, we introduce the sparse-to-dense paradigm which enables a dense modeling of matching probability distributions in an image. In Chapter 5, we go a step further and learn to hallucinate matching distributions of keypoints that are non-covisible across two images.

Structure-based visual localization is the task of predicting the 6-DoF camera pose of an unseen image (the *query* image) captured in a known environment by leveraging a prior knowledge of the world geometry (usually a sparse 3D model). In this framework, existing camera pose estimators minimize a reprojection error based on previously acquired 2D-to-3D correspondences, which has shown to deliver highly accurate results under stable capturing conditions. When relocalizing query images captured over extended periods of time (in a problem known as *long-term* visual localization) or in challenging conditions however, putative 2D-to-3D correspondences are often noisy and unreliable. In this thesis, we will study the limits of structure-based localization and propose a novel camera pose estimator to improve its performance. More specifically in Chapter 4, we will introduce a novel reprojection error which uses dense keypoint matching uncertainties and demonstrate it has strong benefits over standard reprojection errors.

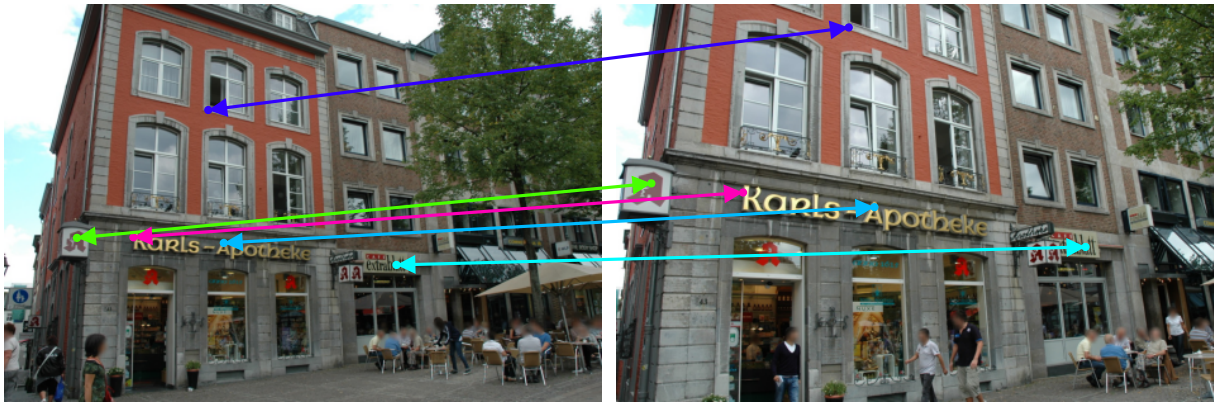
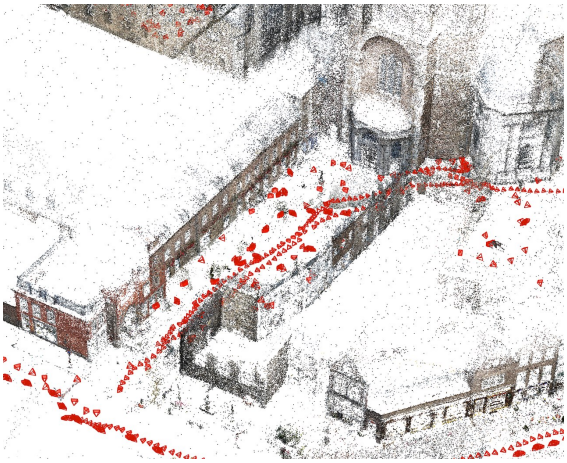
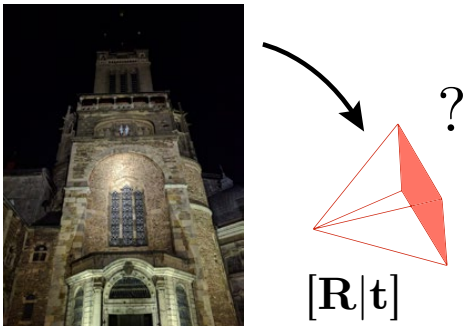


Figure 1.1: **Keypoint matching:** Given a pair of partially covisible images, the goal of keypoint matching is to find a set of 2D-to-2D matches across both images. The accuracy of such correspondences is critical for downstream computer vision tasks like visual localization. Here we show a few putative correspondences using SIFT (Lowe, 2004). SIFT detections are matched using a mutual nearest-neighbour algorithm on their respective SIFT descriptors and filtered using a ratio test (Lowe, 2004) (see Chapter 2 for more details). Images are taken from the Aachen Day-Night dataset (Sattler et al., 2018, 2012)



(a) Offline-computed 3D model with SfM using COLMAP (Schönberger & Frahm, 2016; Schönberger et al., 2016), reference camera poses are shown in red.



(b) A query image captured within the same environment, at a different time, with an unknown camera pose.

Figure 1.2: **Structure-based visual localization:** (a) Given a reference 3D model of the world and a set of reference (image, camera pose) pairs we aim at (b) estimating the camera pose of a previously unseen query image captured in that same environment. This image might have been taken much later on in time, under different lighting or weather conditions, or using a different device. Images and 3D model are from the Aachen Day-Night (Sattler et al., 2018, 2012) dataset.

1.2 Motivations

Visual localization along with its sub-problem of keypoint matching are underpinning backbones to many computer vision applications. Keypoint matching is commonly found in problems involving image alignment such as image registration or Structure-from-Motion (SfM). Visual localization has industrial application ranging from augmented or virtual reality, to robotics and autonomous driving, as illustrated in Figure 1.3.

Image registration. The task of image registration consists in aligning images into a unified coordinate system in order to compare or combine their information. This task can be encountered in applications such as alignment of satellite images (Bentoutou et al., 2005; Yang et al., 2018) captured over long periods of time or building wide panoramic images from multiple images of the same landscape (see Figure 1.3a). The problem of image registration can be tackled in different ways (Szeliski, 2004; Zitová & Flusser, 2003). One approach is to find local correspondences between image pairs and apply rigid or non-rigid transformations on either image to minimize the distance between pairs of matched keypoints. When dealing with strong visual perturbations commonly encountered in crowd-sourced images however, finding accurate keypoint correspondences can be very challenging for reasons which will be discussed in chapter 2.

Structure-from-Motion. Building a 3D representation of the world from a set of sparsely captured images can be done using Structure-from-Motion (Heinly et al., 2015; Lucas & Kanade, 1981; Schönberger & Frahm, 2016; Schönberger et al., 2016; Sweeney et al., 2016), or SfM for short (see Figure 1.3b). In SfM, we first apply keypoint matching to identify sets of local 2D-to-2D correspondences between image pairs. These sets of keypoint matches are in turn used to estimate both the camera poses and a sparse 3D model of the world, using epipolar constraints. Here again, having accurate keypoint matches is critical to the quality of both the 3D reconstruction and the estimated camera poses.

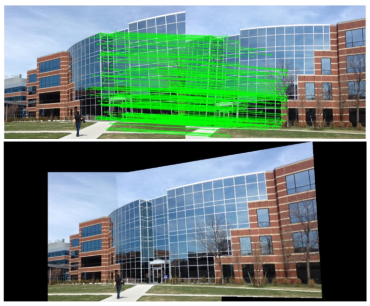
Augmented reality. Augmented Reality (AR) is an increasingly popular technology, which enables seeing overlayed information through a display capturing the world in real-time (e.g. a phone (Middelberg et al., 2014) or a see-through headset as shown in see Figure 1.3c). AR applications range from playing video games embedded in the real world to assisting a worker performing maintenance on a complex piece of engineering. At the core of any AR technology lies a positioning algorithm which estimates the position and orientation of the display w.r.t. the world. This algorithm is then used to render visually aligned information on the display. Critically, when turning on the device, one needs to have a first estimate of its 6 DoF pose in space. In practice this is often done with cameras and a visual localization algorithm. The robustness and precision of this algorithm is

crucial to prevent any visible drift in space and make for a seamless visual experience. In addition to being robust and accurate, visual localization needs to be computationally efficient given the limited resources offered by such low-power devices (Tran et al., 2019).

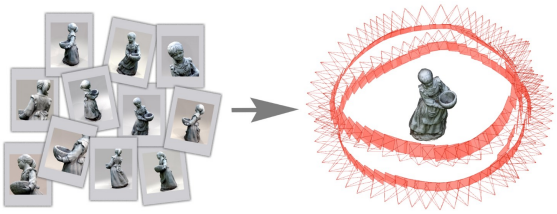
Virtual reality. Similar to AR, Virtual Reality (VR) uses a fully immersive headset to display an entirely virtual world to the user, matching his/her head motion in space to render plausible environments in real-time. Figure 1.3d shows such a VR display developed by Oculus which tracks users hands to perform interactions with a virtual screen. VR displays can be used to interact with entirely digital worlds and offer entertaining or even social experiences. As for AR, VR also relies on a head tracking technology which can be initialized using visual localization (to make sure the physical boundaries of one's room are respected for instance).

Robotics. Another frequently found application for visual localization is robotics, especially for those evolving in our physical space that need to be simultaneously aware of their surroundings (mapping) and of their position (localization). This task is often referred to as Simultaneous Localization and Mapping (Bailey & Durrant-Whyte, 2006; Csorba, 1997; Durrant-Whyte & Bailey, 2006; Durrant-Whyte et al., 1996), or SLAM for short. In Figure 1.3e we show an autonomous lawn mower prototype developed by Bosch which needs to map and navigate a garden to cover all of the grass to be mowed while avoiding collisions. While sensors acquiring 3D scene geometry such as Lidars have become increasingly popular to help with localization (Zhang et al., 2014; Zhang & Singh, 2014), their price is prohibitive for small-budget robots. Thus, robots are instead often equipped with simple RGB or RGB-D cameras which are cheaper (Engel et al., 2014; Mur-Artal et al., 2015). Given a prior model of the 3D world robots need to acquire an initial location estimate when being turned on, which is again done using visual localization. In addition, visual localization can be performed again sporadically through time to correct for any accumulated drift due to noise in other sensors (IMU, odometer, etc.), or for loop closure.

Autonomous driving. One last application leveraging visual localization is autonomous driving vehicles. Much like smaller robots, they are designed to autonomously evolve within unstructured environments. Their size however allows for a larger fleet of sensors whose information can be combined (sensor fusion) to make even more accurate predictions. The self-driving car prototype developed by Waymo relies on Lidars, cameras and other sensors to relocalize in space (see Figure 1.3f). In autonomous driving, accurate camera pose (and thus vehicle pose) estimation is critical to both passengers and surrounding people's safety, which stresses the need for reliable visual localization algorithms.



(a) SIFT (Lowe, 2004)-based image registration (taken from (Tareen & Saleem, 2018))



(b) Structure-from-Motion applied to a set of sparsely sampled images (taken from (Bianco et al., 2018))

(c) The HoloLens 2, an Augmented Reality display developed by Microsoft [a].

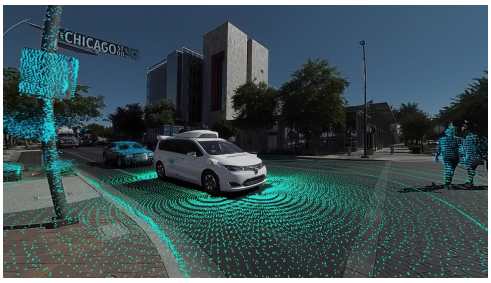
[a] https://microsoft.com/

(d) A Virtual Reality display developed by Oculus [a].

[a] https://oculus.com/

(e) An autonomous lawn mower developed by Bosch [a].

[a] https://bosch.com/



(f) An autonomous car developed by Waymo [a].

[a] https://waymo.com/

Figure 1.3: **Motivation:** Examples of commonly found practical applications involving keypoint matching (a-b) and visual localization (c-f), which are two problems studied in this thesis.

1.3 Approach and context

In this thesis we seek to improve the performance of structure-based visual localization algorithms. This is a particularly difficult problem due to the wide variety of visual perturbations that can occur when capturing images in the wild. It is such a difficult problem that to this day no method is able to accurately relocalize a night-time query image within a 3D model acquired during the day (Sattler et al., 2018). This can be explained by the lack of models powerful enough to learn appearance-invariant representations, as well as the lack of sufficient available training data. Another challenging problem occurs when pairs of query and reference images have very little visual overlap. In those cases, one has to rely on the limited covisible image regions to reason about the query camera pose.

While the problem of visual localization is still far from being solved under challenging conditions, it is important to note that the advent of deep learning has enabled tremendous progress in that area during the past few years.

This thesis thus comes to a crossroads, when three favorable factors have converged to improve keypoint matching and visual localization:

- Deep Learning models have shown outstanding progress, especially in the domain of image processing with Convolutional Neural Networks (CNNs) (Krizhevsky et al., 2012) which enable higher-level learning of image representations compared to handcrafted methods.
- Large collections of annotated data have been gathered and can now be used to train such models, such as the collection of over 1 million annotated internet images Megadepth (Li & Snavely, 2018).
- Efficient hardware (GPUs) powerful enough to train deep learning models efficiently have been widely democratized.

The angle of this thesis is thus to exploit advances in deep learning and available annotated data to derive a data-driven approach to camera pose estimation. In particular we will show how neural networks can be trained to model image-based keypoint matching distributions, and how these distributions can in turn be used to improve existing absolute camera pose estimation algorithms.

1.4 Challenges

In this section we present three commonly encountered challenges when trying to solve 2D-to-3D matching and absolute camera pose estimation in challenging conditions, which underpin the difficulty of these problems.

An ill-posed problem. Most cameras have by construction a limited field-of-view, which exclude most of the environment from the image plane. The dynamic nature of the world also frequently introduces occluders, which can partially or entirely hide a given scene. As a result, solving visual localization is inherently limited by the available information present in the image to relocalize. This can lead to frequent ill-posed configurations, where the visual content is either ambiguous (see Fig. 1.4e), noisy (see Fig. 1.4g) or incomplete (see Fig. 1.4h). A common example for this is the case of repetitive patterns, which are often found in corridors. One can easily imagine that in an infinite corridor with textureless walls and floor, any translation along the corridor axis could yield an equally probable pose. Solving the problem of visual localization thus implies taking such ambiguities into consideration and making the most out of the presented visual data and its global context.

Imbalanced and noisy annotations. Resorting to deep learning models to improve visual localization has shown to work very well to improve robustness to long-term visual disturbances (Sattler et al., 2018). However, such models rely on large amounts of training data which are (in the case of supervised training approaches) annotated. For the problem of visual localization annotations consist in ground-truth camera poses and 3D keypoint triangulations. These annotations can be obtained in a fully-automated way (e.g. using SfM) or with manual annotations. We find in practice however that such annotations are often inaccurate (SfM annotations come from keypoint matches which are prone to noise and errors) and imbalanced (in the case of Megadepth (Li & Snavely, 2018), there are far more daytime images than nighttime images for instance). Recent approaches propose to denoise SfM models (Dusmanu et al., 2020; Lindenberger et al., 2021; Zhang et al., 2021) however obtaining pixel-perfect SfM reconstructions under very challenging capturing conditions remains an open problem. Other methods like (Detone et al., 2018; Revaud et al., 2019) resort to the use of either synthetic images or data augmentation to palliate to the lack of precision and class imbalance. However this then raises the question of generalization of such methods to real images. Dealing with limited and noisy annotations is thus an important component to learning-based 2D-to-3D matching and therefore structure-based visual localization.

Partial data exploitation. When presented with two partially covisible images containing ground-truth keypoint correspondence annotations (e.g. through camera poses, calibration and reprojections of a 3D model), existing learning-based description and matching approaches are incapable of leveraging non-covisible ground-truth correspondences. As will be explained in Chapter 2, a popular paradigm to structure-based localization is to detect keypoints in both images and filter out non-covisible keypoints from the set of correspondences, as they are unlikely to be reliable. When trying to match image pairs with very low visual overlap however, that leaves both very little data to train from and to match/relocalize with. We argue this partial exploitation of image annotations is detrimental to the overall performance of deep learning models, especially in low-overlap scenarios. Harnessing the non-covisible data to train keypoint matching models is however a non-trivial task, which we adress in Chapter 5.

(a) Reference image

(b) Change in scale

(c) Change in viewpoint

(d) Change in illumination

(e) Change in focal length

(f) Change in focus

(g) Motion blur

(h) Occlusion

(i) Change in appearance

Figure 1.4: **Challenges of long-term visual localization:** We report crowd-sourced images from Megadepth (Li & Snavely, 2018) captured around Big Ben, London over an extended period of time. We find such images show strong visual disturbances which make visual localization particularly challenging. While state-of-the-art methods can handle to some extent cases relocalization of (b), (c) and (d), cases (e-i) are much more difficult even for a human. The natural symmetry of Big Ben for instead makes case (e) ambiguous, while (f) and (g) display significant blur which makes accurate localization unfeasible.

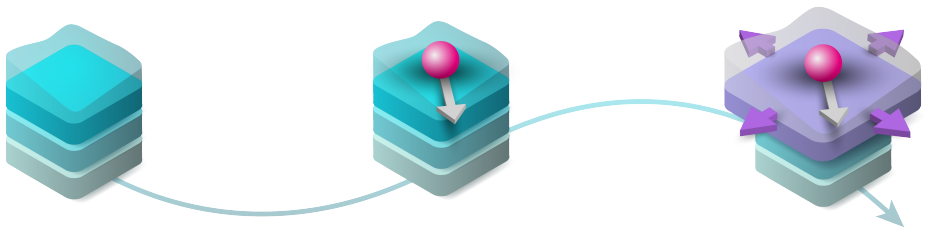
1.5 Contributions

In an attempt to improve the performance of structure-based visual localization algorithms, this manuscript presents three main contributions.

Sparse-to-dense matching. We first propose a novel asymmetric paradigm to perform keypoint matching, by detecting keypoints in one image and searching exhaustively for their correspondent in the other. We derive two formulations exploiting this paradigm, one weakly supervised and one strongly supervised. We show sparse-to-dense matching enables much more accurate retrieval of 2D correspondents from 3D source keypoints compared to state-of-the-art methods, while also being scalable and efficient.

The Neural Reprojection Error. In this thesis we also introduce the Neural Reprojection Error (NRE) which leverages dense matching distributions computed when performing sparse-to-dense matching. We propose a novel localization method that leverages the NRE for both accurate and efficient camera pose estimation. We find our approach significantly outperforms standard reprojection error-based camera pose estimators, including state-of-the-art ones.

Visual correspondence hallucination. Lastly, we go a step further and seek to find a model capable of hallucinating correspondences in non-covisible image areas (occluded or out of image bounds). We propose both a training pipeline and a network architecture to obtain such a model, which we call NeurHal. We not only find that NeurHal is indeed able to generalize correspondence hallucination on novel scenes, but that when coupled with the Neural Reprojection Error it is strongly beneficial for absolute camera pose estimation on low-overlap scenarios.



1.6 Outline

The outline of this thesis is as follows:

Chapter 2: Related Work. This chapter presents the existing work related to visual localization. We review seminal work as well as more recent state-of-the-art approaches. We cover deep learning and structure-based localization methods, as well as work on end-to-end localization and keypoint matching.

Chapter 3: Sparse-to-Dense Matching. In this chapter we introduce the sparse-to-dense matching paradigm, which will be reused heavily throughout the manuscript. We first present this paradigm and discuss its pros and cons with respect to other matching frameworks. We introduce the notion of correspondence maps, which are dense matching distributions over an image. We then present two approaches to learn to perform sparse-to-dense matching, one which is weakly supervised and one strongly supervised. We introduce S2DNet, a model designed to generate accurate dense matching distributions and find it is able to outperform sparse-to-sparse matching methods on both image matching and visual localization applications. We discuss two outlier filtering strategies and show how they can be used to obtain reliable keypoint correspondences. Lastly we report both quantitative and qualitative results demonstrating the potential of sparse-to-dense matching for long-term visual localization.

Chapter 4: Merging Feature Learning and Camera Pose Estimation. This chapter builds upon the previous one to leverage dense information in correspondence maps to perform camera pose estimation. Here we introduce a novel reprojection error (which we call the Neural Reprojection Error) that exploits densely predicted keypoint matching distributions. In this chapter we also introduce a novel coarse-to-fine method that makes the computation of the NRE efficient. We propose a novel camera pose estimation method based on the NRE, and quantitatively demonstrate it is able to outperform state-of-the-art camera pose estimators based on standard reprojection errors.

Chapter 5: Visual Correspondence Hallucination. Our last contribution is presented in this chapter, in which we reuse the NRE in the context of correspondence hallucination. We refer to correspondence hallucination as the task of finding keypoint matches in non-covisible image areas (i.e. due to occlusion or out-of-image-bound reprojection) across image pairs. We aim at answering two questions: (*i*) can we find a network architecture able to learn to hallucinate correspondences? and (*ii*) is correspondence hallucination beneficial for absolute pose estimation? We propose a novel architecture and training method for correspondence hallucination which produces compelling results. In

addition, we find that when coupled with the NRE our model is able to significantly improve absolute camera pose estimation on very low-overlap image pairs.

Chapter 6: Conclusion. In this chapter, we reflect on our contributions and summarize this manuscript takeaways. We also propose several paths for future research building upon our contributions.

1.7 List of Publications

We present four papers in this thesis:

- Germain, H., Bourmaud, G., & Lepetit, V. (2019). Sparse-To-Dense Hypercolumn Matching for Long-Term Visual Localization. In *International Conference on 3D Vision (3DV)*
- Germain, H., Bourmaud, G., & Lepetit, V. (2020). S2DNet: Learning Image Features for Accurate Sparse-to-Dense Matching. In *European Conference on Computer Vision (ECCV)*
- Germain, H., Lepetit, V., & Bourmaud, G. (2021a). Neural reprojection error: Merging feature learning and camera pose estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 414–423)
- Germain, H., Lepetit, V., & Bourmaud, G. (2021b). Visual correspondence hallucination: Towards geometric reasoning. In *arXiv Preprint*

The first paper presented at 3DV 2019 was originally part of a visual localization challenge in 2019 [1], and reached 2^{nd} place. A minimal source code to load and run S2DNet is available at this link. The source code for the Neural Reprojection Error is available at this link. I was fortunate to be given the opportunity to present some of my work in various research laboratories such as Willow (INRIA), BAIR (Berkeley), Matchlab (Imperial College of London), FRL (Facebook Reality Labs), Google X or Bosch AI Center.

During my PhD I also contributed to other papers which will not be discussed in this thesis:

- Sarlin, P.-E., Unagar, A., Larsson, M., Germain, H., Toft, C., Larsson, V., Pollefeys, M., Lepetit, V., Hammarstrand, L., Kahl, F., & Sattler, T. (2021). Back to the Feature: Learning robust camera localization from pixels to pose. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*
- Germain, H., Bourmaud, G., & Lepetit, V. (2018). Improving nighttime retrieval-based localization. In *arXiv Preprint*

[1] https://www.visuallocalization.net/

Chapter 2

Related Work

In this section we review existing work linked to the problem of learning-based absolute camera pose estimation. In Section 2.1 we briefly review the history of deep learning for computer vision and introduce architectures that will be reused throughout this manuscript. In Section 2.2 we review the progress 2D-to-2D keypoint matching methods, which are often directly transferred to the solve 2D-to-3D matching inside structure-based visual localization pipelines. Lastly in Section 2.3 we give an overview of the different absolute camera pose estimation methods, covering not only structure-based but also end-to-end localization approaches.

2.1 Deep learning

This thesis relies heavily on the power of deep learning-based models for computer vision tasks. Thus, we start by reviewing the history of such models since their inception. Deep learning-based models have played a critical role in helping to improve many computer vision tasks such as image classification (Deng et al., 2009), segmentation (Ronneberger et al., 2015) or image generation (Goodfellow et al., 2014). Many would argue their boom occurred around 2012 with the introduction of AlexNet (Krizhevsky et al., 2012) during the ImageNet (Deng et al., 2009) competition. Since then, deep learning research has tremendously grown. In this section we will briefly review deep learning architectures, optimizers and training efficiency.

2.1.1 Architectures

In this subsection we present a short history of deep convolutional neural networks.

2.1.1.1 A brief history of neural networks

The inception of deep learning models can be traced back to the Perceptron (Rosenblatt, 1957), a bio-inspired neural model applied to binary classification. Soon after, the first backpropagation algorithm using the chain rule was derived (Dreyfus, 1962) enabling a simple framework for the parameter updates, followed by the first derivation of a Multi-Layer Perceptron (Ivakhnenko & Lapa, 1966). The criticism of the Perceptron model in (Minsky & Papert, 1969) however subsequently stifled the progress of perceptron learning algorithms for close to thirty years (known as the "AI winter"). During this period some milestones were nonetheless achieved, such as the formulation of a convolutional neural network (CNN) prototype (Fukushima, 1980), the successful implementation of the backpropagation algorithm on a deep neural network (Rumelhart et al., 1986) or the training of a CNN for handwritten digit recognition (LeCun et al., 1989).

The rise in development of GPUs in the late 2000's and the introduction of the large-scale annotated dataset ImageNet (Deng et al., 2009) further motivated the application of deep convolutional neural networks for computer vision applications. The seminal work of AlexNet (Krizhevsky et al., 2012) showed using a GPU-trained CNN could yield significant improvement over the existing state-of-the-art, which then gave rise to a surge in research on deep learning architectures.

2.1.1.2 Convolutional models

Among the large zoo of existing CNN models a few are worth mentioning, some of which will be reused in this manuscript.

VGG (Simonyan & Zisserman, 2014): Although it was introduced 6 years ago, the VGG architecture is still often found in modern deep learning papers. VGG builds upon AlexNet by reusing the idea of a both deep and wide architecture, but proposes a simpler and more homogeneous topology with wider receptive fields. Its main drawback however lies in the size of the model, which induces long inference times and makes it hardly scalable to deeper versions (in particular due to the problem of vanishing gradients).

ResNet (He et al., 2016): ResNet models gave rise to a small breakthrough in training of deep learning models by introducing the concept of parallel residual connections. Such connections strongly temper vanishing gradients and thus enable the use of much deeper architectures (up to roughly 20 times deeper than VGG-16). ResNet models significantly outperformed existing architectures in the ImageNet competition and are still widely used to this day.

Inception-v3 (Szegedy et al., 2016a): The Inception-v3 model is an improved version of the previous GoogLeNet (Szegedy et al., 2015) and Inception-v2 (Szegedy et al., 2016b). The shared archetype of this line of models is to use parallel convolutional kernels with different sizes to increase model capacity without a burdensome computational cost. Inception-v3 further introduces the notion of asymmetric kernels and the use 1×1 convolutions for an improved performance on classification benchmarks.

U-Net (Ronneberger et al., 2015): The U-Net architecture has the specificity of preserving image resolution from end-to-end. This is a particularly suitable architecture for image segmentation tasks. Its architecture also features several residual (skipped) connections as well as a core bottleneck, which provide a very wide receptive field which is useful in many applications beyond segmentation.

As we will see in this thesis, many learning-based keypoint matching or visual localization approaches reuse either truncated or modified versions of the aforementioned models.

2.1.1.3 Transformers for vision

Among the mini-breakthroughs occurring in the field of deep learning, the introduction of Transformers (Vaswani et al., 2017) has been one of them. The Transformer architecture is based on the notion on multi-head attention, which force interactions between every element of an input vector. Transformers were initially applied for NLP tasks and showed tremendous improvements over the previous state-of-the-art. Applying Transformers for computer vision applications however is not straightforward, mainly due to the inherent quadratic cost in the input vector dimension which quickly makes computation intractable for large inputs. By sub-sampling images along regular grids or working at a deeper feature level however, recent approaches (Carion et al., 2020; Dosovitskiy et al., 2021) have shown promising improvements in deploying Transformers for computer vision. This can be attributed in large part to the global image understanding provided by attention, which is hardly achievable with fully-convolutional models that only reason about local features. In Chapter 5, we will use a Transformer-inspired architecture to perform visual correspondence hallucination.

2.1.2 Optimizers

In a supervised setting, training a Θ-parameterized deep learning model $\mathcal{F}_\Theta$ is done by minimizing the Empirical Risk:

$$\mathcal{R}(\mathcal{F}_\Theta, \mathcal{X}, \mathcal{Y}) = \mathbb{E}_\mathcal{X}[l(\mathcal{F}_\Theta)] = \frac{1}{N}\sum_{i=1}^{N} l(\mathcal{F}_\Theta(x_i), y_i) \tag{2.1}$$

where $\mathcal{X} = \{x_1, \ldots, x_N\}$ is a dataset of N samples labeled by $\mathcal{Y} = \{y_1, \ldots, y_N\}$, l is a loss function and Θ is the model parameters. Assuming a differentiable model, one can optimize for the model parameters using gradient descent on Eq. (2.1) with the following update rule:

$$\Theta_{k+1} = \Theta_k - \nu \sum_{i=1}^{N} \nabla l(f_{\Theta_k}(x_i), y_i) \tag{2.2}$$

where ∇ is the gradient operator of l w.r.t. Θ and ν is the step size (or learning rate). On large datasets however, computing Eq. (2.2) is too expensive and we can instead randomly sub-sample mini-batches of data to approximate the true gradient, in a process known as Stochastic Gradient Descent (SGD). Further improvements have been devel-

oped to improve the speed of convergence of gradient descent optimization, and avoid local minima. The first idea is to use momentum as a way of smoothing gradient updates, using a γ-parameterised linear combination of the previous and current gradients. One can then reuse the momentum term to estimate the next parameter update and preemptively update the gradient direction (Nesterov, 1983). AdaGrad (Duchi et al., 2011) rescales individual parameters adaptively based on their previously accumulated gradients to allow for a more uniform distribution of updates. RMSProp (Tieleman & Hinton, 2012) replaces the accumulation formulation of AdaGrad by an exponentially moving average. Lastly Adam (Kingma & Ba, 2015) behaves like RMSProp by replacing the formulation of momentum by an exponentially decaying average of gradients. Additional training tricks like learning rate scheduling (Smith, 2017) and weight decay (Loshchilov & Hutter, 2019) may be used to further improve training convergence.

2.1.3 Training frameworks

Training deep learning models relies on automatic differentiation ("autodiff") to efficiently perform backpropagation. A number of libraries were developed since the early 2000's (e.g. Torch, Theano, Caffe) to train neural architectures using autodiff. Since then, two libraries have overtaken the realm of deep learning training frameworks: TensorFlow and PyTorch. TensorFlow (v1) requires building a static graph (encapsulating the data flow through the model), before compiling and running training or inference. This can be an advantage when deploying models at scale, but limiting when doing modifications to the graph once the model is trained. PyTorch on the other hand relies on graphs built dynamically as computation is being done, which can be advantageous for a more flexible handling of architectures and data flow. Both frameworks have now become highly performant and go-to choices for researchers. The code used in this thesis was entirely written using PyTorch.

2.2 Keypoint matching

The problem of finding 2D-to-3D correspondences between a query image and a 3D model will be a fundamental backbone to structure-based localization. As will be discussed more extensively in the next section, 2D-to-3D matching is often treated as a 2D-to-2D matching problem between the query image and a nearby posed reference image for which depth is partially known (e.g. through SfM or dense depth estimation sensors).

Thus, we will now review computer vision methods that seek to establish correspondences between image pairs, which is a problem known as *keypoint matching* or *image matching*. We will review keypoint detection in Section 2.2.1, description in Section 2.2.2 and matching in Section 2.2.3.

2.2.1 Keypoint detection

Local interest points, also referred to as *keypoints* are 2D locations in an image that are of particular interest for a given application. In the context of image matching keypoints are often located at distinctive locations such as corners, that are both easily identifiable and accurately localizable. We will first present the desired properties of keypoint detectors for image matching, and review both classical and more recent keypoint detection algorithms.

2.2.1.1 Desired properties

Given an RGB image $\mathtt{I} \in \mathbb{R}^{H \times W \times 3}$ of width W and height H, let $\mathbf{p} \in \mathbb{R}^{[0,H]} \times \mathbb{R}^{[0,W]}$ be a 2D keypoint lying in the image space of $\mathtt{I}$. Let us write $\mathtt{I}'$ another image partially covisible with $\mathtt{I}$. We seek to perform keypoint matching on the image pair $(\mathtt{I}, \mathtt{I}')$. Among the infinite locations that $\mathbf{p}$ can have, only a small subset is typically relevant for image matching: finding this subset is known as keypoint detection. As formulated by (Tuytelaars & Mikolajczyk, 2008), an ideal keypoint detector should have several properties to ensure an optimal behaviour.

Repeatability is a key criterion to keypoint detection, which states that a keypoint detected in $\mathtt{I}$ should also be detected in $\mathtt{I}'$ (at its corresponding location). This is critically important in keypoint matching methods that perform detection on both $\mathtt{I}$ and $\mathtt{I}'$, as a non-repeatable detector would prevent any accurate correspondence to be drawn. Keypoint repeatability can be achieved by designing a keypoint detector invariant to both changes in viewpoint and appearance. We will see in Chapter 3 that in practice keypoint repeatability is very hard, especially in a long-term scenario.

Locality of a keypoint detector is the idea that detections should be placed on small,

local (planar-like) image regions. This is as opposed to larger semantic instances for which defining an anchor is more ambiguous. Larger objects might also be subject to occlusion. Relying on local regions thus facilitates the estimation of local transformations.

Accuracy of a keypoint detector is another criterion that stresses the importance of estimating precise keypoint coordinates regardless of image transformations. The accuracy of a keypoint detector is particularly important for downstream applications such as absolute camera pose estimation, as we will see in Chapter 3. This property goes hand-in-hand with both the repeatability and locality criterion.

Additional properties such as distinctiveness, quantity and efficiency are also preferred. With recent advances in learning-based detection we will also see that for visual localization applications, keypoint detections placed at semantically reliable locations is also a valuable property (although more computationally expensive).

2.2.1.2 Handcrafted keypoint detectors

Let us now review classical approaches to keypoint detection, some of which are still widely used today. Many keypoint detectors seek to find corners as they are often repeatable, local and accurate locations across images. Although not applied to natural images, initial attempts at corner detection involved finding high curvature points along edges (Ishimura et al., 1986; Rutkowski & Rosenfeld, 1978). Another line of work for corner detection consists in computing Hessian matrices on local intensities (Beaudet, 1978), and fitting a quadratic surface (Kitchen & Rosenfeld, 1982). Below are a few well-known keypoint detectors that have been built ever since.

Harris corner detector (Harris & Stephens, 1988): A first-order derivative method which has become widely popular is the Harris corner detector. The Harris corner detector computes local gradients using a Gaussian kernel which are subsequently smoothed by a Gaussian window, resulting in a second-order moment matrix M. The eigenvalues of this matrix are then used to compute a *cornerness* score, defined by $det(M) - \lambda tr(M)$. To remove duplicate detections in a small region, non-maxima suppression is applied. To achieve subpixellic accuracy, quadratic fitting can also be applied around a detection. The Harris corner detector is highly repeatable thanks to its invariance to 2D translation and in-plane rotation. However, it often triggers on undesirables areas such as T-junctions and fails to be robust to stronger visual perturbations, such as changes in scale, viewpoint or illumination.

Harris-Laplace (Mikolajczyk & Schmid, 2001): The Harris-Laplace detector performs

multi-scale Harris corner detection and subsequently selects the best scale (known as the characteristic scale) using the Laplacian operator as proposed by (Bretzner & Lindeberg, 1998; Lindeberg, 1993). This detector thus provides an increased robustness to changes in scale for a higher keypoint repeatability. The Harris-Laplace detector can further be extended to the Harris-Affine region detector (Mikolajczyk & Schmid, 2002), which refines detections using affine neighbourhood estimation.

SIFT (Lowe, 2004): The computation of Laplace-of-Gaussian (LoG) (Lindeberg, 1998) can be efficiently approximated using instead a Difference-of-Gaussian (DoG). After computing DoG on multi-scale images, SIFT (Lowe, 2004) proposes to find extrema in that scale-space which are in turn considered as detections candidates. The final list of keypoints along with their scale is obtained by thresholding those local extrema. In addition, SIFT makes detections robust to rotation by associating to each detection a principal orientation based on a local histogram of gradients. SIFT detections are thus both scale and rotation covariant. Note however that the detected keypoint locations no longer correspond to corners but rather blob-like image features.

SURF (Bay et al., 2006): The SURF keypoint detector aims at speeding up the computation done in SIFT to enable real-time applications. This is achieved by approximating the Gaussian filtering applied on images using instead a simpler box filter.

FAST (Rosten & Drummond, 2006): The Features from Accelerated Segment Test (FAST) detector is also built for computational efficiency. It works by retrieving the 16 pixel intensities surrounding candidate keypoint and reasoning about their relative value w.r.t. that candidate intensity using a decision tree. Non-maxima suppression is typically used to remove duplicates around interest regions.

ORB (Rublee et al., 2011): Lastly the ORB descriptor enriches FAST keypoint detections by attributing orientations based on local moments, as well as operating on multiple image scales, similar to SIFT.

2.2.1.3 Learning-based keypoint detectors

The use of learning-based detection can be traced back to (Dias et al., 1995), which derives neural network to perform corner detector from local edges. The approach used by FAST and (Sochman & Matas, 2007) learns to speed up detection using learned binary decision trees. It is only more recently however that learning-based detectors performing higher-level image reasoning have emerged. We review the most prominent ones below and report qualitative results of both handcrafted and learning-based detectors in Figure 2.1.

TILDE (Verdie et al., 2015): In order to improve robustness to illumination changes, the TILDE detector uses a CNN to regress fully-convolutional score maps. These dense score maps are then thresholded, and non-maxima suppression is applied to return the final list of keypoint candidates. Training is done on aligned webcam images captured over an extended period of time, displaying strong weather and illumination changes. These images do not feature however changes in viewpoint.

LIFT (Yi et al., 2016): LIFT is arguably the first attempt at jointly learning to perform keypoint detection and description (more in Section 2.2.2). The keypoint detector part of LIFT uses the same CNN architecture as TILDE, but is trained to reproduce SIFT keypoint reprojections coming from an SfM model built on crowd-sourced images. Subpixellic keypoint locations are further refined using the softargmax (Chapelle & Wu, 2009) operator, which also serves as a non-maxima suppressor.

LF-Net (Ono et al., 2018): LF-Net improves detection from LIFT by computing a feature-based scale-space which allows for an increased robustness to changes in scale. Non-maxima suppression is done using locally applied softmax operators, and subpixellic refinement is done following LIFT. LF-Net is also trained using SfM-annotated outdoor images, and an additional indoor training is performed using ScanNet (Dai et al., 2017).

SuperPoint (Detone et al., 2018): Unlike previous learning-based approaches, SuperPoint proceeds to regress keypoint detections using a three-stage training pipeline. First, a base keypoint detector (formulated with an encoder-decoder architecture) is trained on labeled synthetic images to return high responses at corner locations. Then, the model is finetuned on natural images that undergo homographic adaptation to compute interest points supersets, which are used as ground truth labels. Lastly, the encoder of the detector is jointly finetuned with the descriptor branch on warped images. A cell-based non-maxima suppression is further applied, resulting in fewer but well-located keypoint detections.

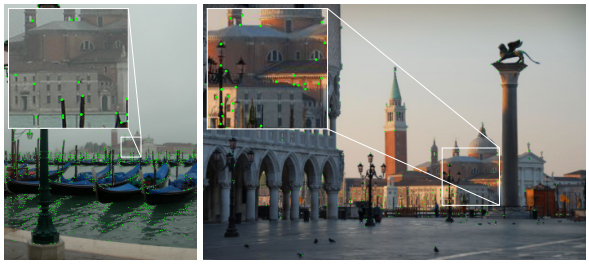
D2-Net (Dusmanu et al., 2019): D2-Net proposes to share the computation of keypoint detection and description by adopting a *detect-and-describe* approach (as opposed to *detect-then-describe* adopted by previous methods). In D2-Net, detection is done by computing a (soft) local-maximum score in descriptor space directly, resulting in differentiable detection score maps. As most CNN-based methods, D2-Net is not scale invariant and requires thus multi-scale inference, by constructing an image pyramid.

R2D2 (Revaud et al., 2019): R2D2 further extends the *detect-and-describe* paradigm by adding a dual loss on both keypoint repeatability and reliability maps. The repeatability loss encourages identical detection responses across images, while the reliability loss uses Average-Precision ranking to predict local descriptors discriminativeness. Keypoint detections are obtained by finding local maxima in the repeatability map and reweighting them using their corresponding reliability value. The final list of keypoints is obtained by taking the top-K scores.

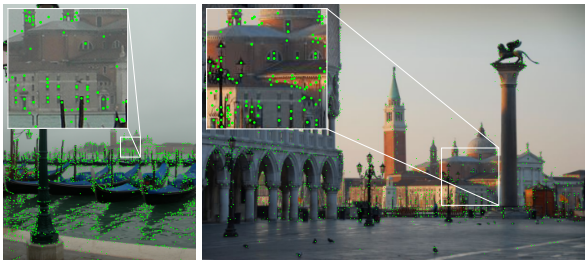
D2D (Tian et al., 2020a): The approach taken by D2D is to flip the *detect-then-describe* strategy to instead *describe-to-detect*. In D2D, keypoint detections are extracted from both absolute and relative salient locations in descriptor maps, as such locations are likely to be informative. When coupled with state-of-the-art keypoint descriptors, D2D outperforms D2-Net and SuperPoint in image matching benchmarks.

DISK (Tyszkiewicz et al., 2020): The problem of finding sparse sets of local keypoints is often done by sub-selecting locations from dense CNN responses. DISK leverages a reinforcement learning strategy to rephrase learning-based keypoint detection. Using dense correspondences (obtained through dense depth maps), a reward function is formulated based on the correctness of predicted keypoint matches between two images. The model is trained through policy gradient, and can not only serve as a keypoint detector but also as a keypoint matcher (see Section 2.2.3).

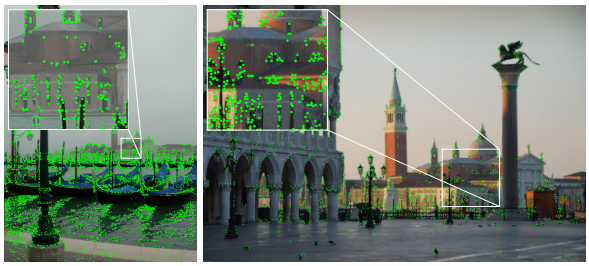
As shown in Figure 2.1, obtaining repeatable keypoint detections across images undergoing strong changes in both viewpoint and illumination remains a hard problem. In Chapter 3, we will show how breaking the symmetry in keypoint detection can lead to superior keypoint matching results.



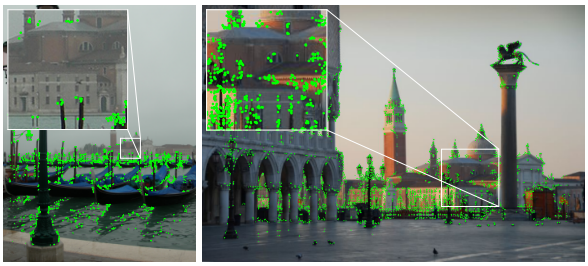
(a) Harris (Harris & Stephens, 1988)



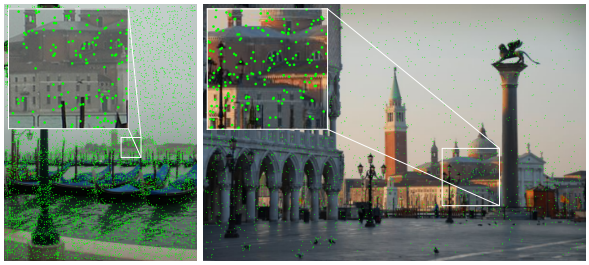
(b) SIFT (Lowe, 2004)



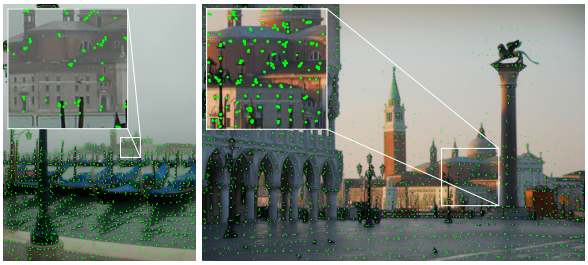
(c) FAST (Rosten & Drummond, 2006)



(d) ORB (Rublee et al., 2011)



(e) D2-Net (Dusmanu et al., 2019)



(f) R2D2 (Revaud et al., 2019)

(g) SuperPoint (Detone et al., 2018)

(h) DISK (Tyszkiewicz et al., 2020)

Figure 2.1: **The keypoint repeatability problem:** We report the keypoint detections obtained with popular methods on two images from Megadepth (Li & Snavely, 2018). We can see that repeatably detecting pixel-perfect interest points across viewpoint and illumination changes remains very challenging, even for state-of-the-art methods like SuperPoint or DISK.

2.2.2 Keypoint description

In order to match two sets of keypoint detections $\{\mathbf{p}_i\}_{i=1}^N$ and $\{\mathbf{p}'_i\}_{i=1}^M$ between $\mathtt{I}$ and $\mathtt{I}'$, keypoint descriptors are usually attributed to every detection for later matching. Let us write $\mathbf{h}_i \in \mathbb{R}^D$ the keypoint descriptor associated to $\mathbf{p}_i$, and $d : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}^+$ a distance function. Just like for keypoint detection, keypoint description has a set of ideal properties which are discussed in Section 2.2.2.1. We then review existing keypoint descriptors, both handcrafted (Section 2.2.2.2) and learning-based (Section 2.2.2.3).

2.2.2.1 Desired properties

Keypoint descriptors have to deal with three constraints which are in opposition with one another. Research in keypoint descriptors aim at improving on at least one of these properties:

Invariance. Keypoint descriptors should exhibit minimal changes under strong visual perturbations. For two corresponding keypoints $\mathbf{p}_i$ and $\mathbf{p}'_j$, this means that $d(\mathbf{h}_i, \mathbf{h}'_j)$ should be close to 0. While early methods were derived to be invariant to 2D translation and in-plane rotations, dealing with stronger perturbations such as out-of-plane rotation, changes in viewpoint or in appearance are incredibly more complex. We will see that the advent of CNNs have enabled tremendous progress in that area. As we will show in Chapter 5, modern architectures can in fact go so far as being robust to *out-of-visibility* disturbances.

Discriminativeness. In addition to invariance, keypoint descriptors should be highly discriminative. This means that for any two distinct descriptor $\mathbf{h}_i$ and $\mathbf{h}_j$, their distance $d(\mathbf{h}_i, \mathbf{h}_j)$ should be high. This is particularly challenging when presented with images containing repetitive patterns, as each pattern should have a unique descriptor but is in practice often visually identical. Solving this issue requires having a global understanding of the image context, which even state-of-the-art deep learning models struggle achieving.

Compactness. To enable efficient comparisons between two sets of descriptors, it is also desirable for D to be small. Having invariant, discriminative and very low-dimensional descriptors is particularly challenging as the descriptor function needs to be highly optimized for invariance and discriminativeness, while remaining generalizable. In practice, common descriptors have sizes ranging from 128 to 1024, although it is not uncommon to encounter lower or higher values.

2.2.2.2 Handcrafted keypoint descriptors

Let us first review handcrafted keypoint descriptors, which are still widely used for their robustness and efficiency in simple image matching cases.

SIFT (Lowe, 2004): The keypoint descriptor derived by SIFT relies on local orientation histograms. Around a given keypoint detection, a 16×16 window is extracted, subdivided in 4×4 cells and an 8-bin orientation histogram is computed for each of the 16 cells. The 16 8-bin values are then concatenated to form a 128-dimensional vector. Thanks to an additional estimation of the keypoint orientation using the orientation histogram, we can make the descriptor robust to in-plane rotations by substracting keypoint orientation to the local gradient orientations.

SURF (Bay et al., 2006): Following SIFT, SURF descriptors rely on an initial orientation estimation of keypoints through an iterative computation of Haar-wavelet function is several directions. A square region oriented along the previous estimation is subsequently extracted, and a grid-based sum of local wavelet responses are computed, forming a 64-dimensional descriptor.

BRIEF (Calonder et al., 2010): The BRIEF descriptor is a binary descriptor meant to be highly efficient to both compute and match. After applying a Gaussian smoothing on the image (for noise reduction), pixel value intensities are compared between the keypoint pixel and any other local keypoint (based on a given sampling strategy). This binary comparison results in a variable-sized binary descriptor, which typically ranges from 128 to 512.

ORB (Rublee et al., 2011): The main drawback of BRIEF is it is not robust to image rotations. ORB derives a rotated formulation of BRIEF based on the keypoint orientation estimation (provided by its keypoint detector).

2.2.2.3 Learning-based keypoint descriptors

The main limitation of handcrafted descriptors is the lack of robustness to strong appearance changes, particularly to different illuminations. CNNs can be used to vastly increase invariance to such perturbations.

LIFT (Yi et al., 2016): LIFT descriptors are computed with a small fully-convolutional network, which is trained using a simple contrastive loss (Simo-Serra et al., 2015) on pairs of positive and negative image patches with hard mining. A differentiable orien-

tation estimator is also derived to rectify patches prior to computing descriptors, which increases robustness to rotation. Results demonstrate strong quantitative improvements over handcrafted baselines.

LF-Net (Ono et al., 2018): LF-Net proceeds to learn descriptors similar to LIFT, but use a patch-wise triplet loss similar to (Balntas et al., 2016a,b), as well as a larger training set.

HardNet (Mishchuk et al., 2017): In order to improve contrastive descriptor learning on patches, the authors propose to enrich the standard triplet loss (Balntas et al., 2016b) by maximizing the distance between the closest positive and negative w.r.t. the anchor patch. The model (coined HardNet) uses an L2Net (Tian et al., 2017) backbone, and is still highly competitive to this day (Jin et al., 2021b).

SOSNet (Tian et al., 2019): SOSNet considers a second-order regularization term which is added to HardNet's first-order loss. This second-order term encourages pairs of negative samples to be equally distant from one another, which has a positive impact on the end descriptors.

HyNet (Tian et al., 2020b): HyNet also relies on a contrastive learning approach through a triplet margin loss, but adds additional regularization terms that provide better gradients under normalization as well as intermediate L2 normalizations throughout the network.

SuperPoint (Detone et al., 2018): SuperPoint descriptors are learned using a hinge loss (contrastive learning), with a hyperparameter to balance the ratio of positive and negative samples.

D2-Net (Dusmanu et al., 2019): D2-Net learns descriptors using a simple triplet loss, which is jointly optimized with keypoint detection.

R2D2 (Revaud et al., 2019): While jointly learning keypoint repeatability and reliability, R2D2 learns to output keypoint descriptors using an Average-Precision ranking loss (He et al., 2018), which has shown encouraging results in nearest neighbour retrieval.

2.2.3 Keypoint matching

Matching two sets of sparse keypoint locations (which we will later refer to as *sparse-to-sparse* matching) does not only consists in finding pairs of keypoints belonging to the

same image location, but also identifying keypoints which are not covisible (e.g. due to occlusion or camera movement). Finding such outlier correspondences can be done *a priori* based on keypoint descriptors and 2D locations, or *a posteriori* using additional geometric considerations in the case of visual localization (see Section 2.3.2.2).

2.2.3.1 Handcrafted keypoint matchers

The simplest algorithm to identify the keypoint correspondent $\mathbf{p}_i$ in $\mathtt{I}$ from the set of keypoints $\{\mathbf{p}'_j\}_{j=1}^{M}$ is to exhaustively search for its Nearest Neighbour (NN) based on descriptor distances (usually Euclidean, or using the Hamming distance for BRIEF and ORB). An alternative to speed-up this search is to use k-Nearest Neighbour (k-NN) algorithm. The simplicity of this matching approach implies great discriminativeness and invariance in the keypoint descriptors, which is often hard to achieve. In order to filter potential outlier keypoint correspondences however, one can resort to several heuristics. The first one is to employ a mutual nearest neighbour verification, which typically helps removing non-covisible keypoint pairs. A more sophisticated test to remove unreliable correspondences (coming from repetitive patterns for instance) is the Ratio test (Lowe, 2004), which compares the ratio between the first and second nearest-neighbour distances. GMS (Bian et al., 2017) shows strong improvements by deriving a simple grid-based heuristic to check local spatial consistency of keypoint matches.

2.2.3.2 Learning-based keypoint matchers

Learning to predict keypoint correspondences can also be tackled using neural networks. Given a set of putative correspondences (e.g. coming from a handcrafted matcher), (Moo yi et al., 2018) train a simple coordinate-based MLP to regress correspondence confidence scores. The approach taken by (Ranftl & Koltun, 2018) is to train a network to regress similar weights, trained for a homogeneous least-squares problems to solve fundamental matrix estimation. OANet (Zhang et al., 2019) also uses a simple neural network to regress confidence in valid keypoint correspondences, with a differentiable essential matrix estimation objective. Using a soft assignment matrix, OANet is also able to achieve invariance to the order to permutation in the correspondences order.

SuperGlue (Sarlin et al., 2020) uses an attentional graph neural network to directly learn to perform keypoint matching between two sets of SuperPoint (Detone et al., 2018) keypoints. It uses both keypoint detection coordinates and SuperPoint descriptors to find putative correspondences, through a differentiable optimal matching module. SuperGlue is currently a state-of-the-art sparse keypoint matching model. More recently Patch2pix (Zhou et al., 2020a) learns to not only regress outlier correspondences but also locally refine the keypoint coordinates of valid ones. Patch2pix consists of a correspondence network to filter matches, as well as a refinement network to predict local matches

within the prior keypoint detection area. Training is done in a weakly supervised way by using epipolar constraints from camera poses.

2.2.3.3 Image-based matchers

Another approach to this problem is image-based keypoint matching (which we will later refer to as *dense-to-dense* matchers). Such approaches skip the keypoint detection and description altogether, and directly regress keypoint correspondences across pairs of RGB images. This paradigm implies that both images are processed jointly which typically increases the overall computational cost of dense-to-dense approaches. NCNet (Rocco et al., 2018) computes 4D correlation volumes using dense feature maps outputted by fully-convolutional ResNets to subsequently regress keypoint correspondences. Due to the large tensors which are required to compute such volumes however, NCNet has a strong memory footprint and requires regressing keypoint coordinates at a lower resolution. To mitigate memory issues and allow for better localized correspondences, SparseNCNet (Rocco et al., 2020) turns the 4D correlation volume into a sparse tensor and further relies on sparse convolutions to regress more accurate correspondences. More recently LoFTR (Sun et al., 2021) runs several Transformer models on dense image features to perform both self and cross-attention operations. These updated features are in turn fed to a differentiable keypoint matching module. To avoid high memory consumption the Transformer models use a linear kernel (Katharopoulos et al., 2020), and LoFTR additionally resorts to a coarse-to-fine mechanism to allow for accurate correspondence prediction. Subsequent work on dense image-based matchers have been developed for tasks like dense flow estimation (Shen et al., 2020; Truong et al., 2020a, 2021, 2020b).

2.2.4 Discussion

Keypoint matching is frequently encountered in many computer vision applications, and has seen tremendous progress since the advent of deep learning models. It is important to notice however the permanent loss of information that occurs throughout this process. In running keypoint detection, the spatial locations covered by an image are reduced to a much smaller, sparse set of coordinates. After the description and matching stage, a set of putative correspondences is emitted, which is likely to contain *outlier* matches as well as globally valid but locally noisy keypoint correspondences. In a subsequent task such as visual localization (that will be presented below), converting dense image pairs to a set of sparse, partially invalid and noisy correspondences implies that a lot of filtering work will have to be done *a posteriori*. In this thesis, we argue that the permanent loss of information induced by the keypoint matching process should be avoided for the task of absolute camera pose estimation.

2.3 Visual localization

The task of visual localization (also sometimes referred to as relocalization) consists in predicting the absolute camera pose of a given image captured in a known environment. While there exists many approaches to tackle this problem, we distinguish methods which leverage the available 3D data (see Section 2.3.2) from those working from RGB images only (see Section 2.3.3).

2.3.1 Formalism

Let us first introduce notations which will be reused throughout this thesis. The (previously unseen) image to localize is called a *query* image which we will denote by $\mathtt{I}_q$. We will assume $\{\mathtt{I}_r^i\}_{i=1}^{N_r}$ a set of reference images, captured in the same environment as $\mathtt{I}_q$ and for which offline computation is permitted. For instance, by running SfM on reference images we can obtain their 6-DoF camera pose $\{\mathtt{P}_r^i\}_{i=1}^{N_r}$ where $\mathtt{P}_r^i = [\mathtt{R}_r^i|\mathbf{t}_r^i] \in \mathrm{SE}(3)$, as well as a sparse 3D reconstruction $\mathcal{M} = \{\mathbf{u}_i\}_{i=1}^{M}$ with $\mathbf{u}_i \in \mathbb{R}^3$. In visual localization, we will always consider the set of reference camera poses along with a 3D model of the scene to be known and consider them ground truth. We also assume the linear calibration matrices of both reference and query images to be known and we write them as $\mathtt{K}$[1]. Let π be the projection function which maps a 3D point to the camera plane defined by $\pi(\mathbf{u}) := [\mathbf{u}_x/\mathbf{u}_z, \mathbf{u}_y/\mathbf{u}_z, 1]^\intercal$. The 2D reprojection of $\mathbf{u}_i$ in the image plane of $\mathtt{I}_r^j$ is given by $\mathtt{K}\pi(\mathtt{R}_r^j\mathbf{u}_i + \mathbf{t}_r^j)$. In a slight abuse of notation, we do not distinguish a homogeneous 2D vector from a non-homogeneous 2D vector.

2.3.2 Structure-based localization

Structure-based localization aims at leveraging the available 3D model of the world to accurately estimate the query pose. 3D reconstructions provide strong geometric priors of the world which can be highly valuable when estimating absolute camera poses, which we will now review.

2.3.2.1 Linear pose estimation

We will first describe how finding explicit 2D-to-3D correspondences between $\mathtt{I}_q$ and $\mathcal{M}$ can be leveraged to estimate the query pose $\hat{\mathtt{P}}_q$. Given a set of M 2D-to-3D correspondences $\mathcal{U} = \{(\mathbf{u}_i, \mathbf{p}_i^q)\}_{i=1}^{M}$ between $\mathcal{M}$ and $\mathtt{I}_q$ we seek to find the camera pose estimate $\hat{\mathtt{P}}_q$ with respect to the 3D points reference frame (a.k.a the world). This is a well-studied problem often referred to as the Perspective-n-Pose problem (or PnP for short), where n

[1]We assume *w.l.o.g.* the images are rectified according to the pinhole camera model.

is the number of correspondences. The minimum number of correspondences to constraint this 6-DoF problem when the camera is calibrated is three (P3P), which can be solved in several ways through a linear system of equations (Haralick et al., 1991, 1994). P3P however leads to eight possible solutions (including four behind the camera plane). These solutions can be disambiguated with a fourth point, provided any subset of three out of four points are not coplanar with the camera center (Wrobel, 2001). When moving up to higher number of correspondences, the impact of noise in either 2D keypoint detections or 3D keypoint triangulations is reduced. Solving PnP with $n > 4$ is thus likely to provide more accurate camera pose estimates. (Quan & Lan, 1999) shows that for $n \geq 5$ the camera pose can be estimated by linearly solving a system of fourth degree polynomials. The computational cost non-iterative linear solving of the PnP being quadratic at best (Fiore, 2001), EPnP (Lepetit et al., 2008) expresses the coordinates of the n 3D points as a weighted sum of four virtual control points, which allows for a linear computational cost in $O(n)$. (Kneip et al., 2011) propose a faster closed-form solution for the P3P problem by avoiding to estimate a typically computed intermediate reference frame between the 3D points and the camera.

2.3.2.2 Geometric outlier filtering

Solving for the PnP using $\mathcal{U}$ can be problematic for two reasons. First in methods relying of few correspondences (e.g. P3P, P4P), there is a chance of coplanarity with the camera center which prevents from solving the pose without ambiguities. Second, encountering outlier correspondences in $\mathcal{U}$ will naturally lead to noisy or erroneous camera pose estimates.

RANSAC (Fischler & Bolles, 1981). A simple idea to avoid such pitfalls consists in using iterative random sampling of correspondence tuples followed by a linear solving of the pose, and selecting the most likely candidate based on given quality function. The most famous example is to solve PnP inside a RANdom SAmple Consensus (RANSAC) (Fischler & Bolles, 1981) loop. In RANSAC, camera pose hypotheses are computed from randomly subsampled correspondence tuples, and can be further refined based on additional samples supporting a given hypothesis. We define an *inlier threshold* (i.e. a reprojection error threshold in pixels) to compute an inlier count for a given hypothesis, and select the camera pose candidate that maximizes that number. The number of RANSAC iterations can be defined to ensure a probabilistic guarantee of finding the optimal model. We will denote the set of inlier correspondences as $\mathcal{I}$ such that $\mathcal{I} \subset \mathcal{U}$.

RANSAC variants. There exists many variants of RANSAC. MSAC and MLESAC (Torr & Zisserman, 2000) propose to model the inlier error as an unbiased Gaussian distri-

bution and replace the threshold hyperparameter by an target error tolerance instead. PROSAC (Chum & Matas, 2005) takes into account a given likelihood of every putative correspondence to improve the sampling strategy of tuples. LO-RANSAC (Chum et al., 2003; Lebeda et al., 2012) performs additional local optimization for a given hypothesis using an Iterative Least Squares (IRLS) solver. Graph-Cut RANSAC (Baráth & Matas, 2018) leverages the local spatial coherence in a given scene to improve the local optimization of LO-RANSAC. The likelihood ratio test proposed by (Cohen & Zach, 2015) enables control over the inlier noise in addition to model parameters. MAGSAC (Barath et al., 2019) aims at eliminating the need for an inlier threshold hyperparameter by marginalizing over its likelihood (assuming uniform outlier distributions). In this case inliers can be only defined through their likelihood and the termination factor is determined by a noise scale upper bound. MAGSAC++ (Baráth et al., 2020) provides a more efficient computation of the marginalized distribution by resorting to an IRLS solver, and removes the need for a uniform outlier distribution assumption.

Recent studies (Chum et al., 2020; Trulls et al., 2019) on RANSAC-based algorithms and implementations tend to show a strong disparities between methods. In particular, hyperparameter tuning of the inlier threshold seems to have a strong impact in the camera pose estimation accuracy. This is an indicator that identifying outlier correspondences might be a hardly generalizable task, which does not have a universal answer.

2.3.2.3 Non-linear optimization

While solving for the PnP linearly inside a RANSAC loop is efficient (and to some extent robust to outliers), it fails to capture the local non-linearities of the 3D models. A refinement step is thus frequently applied to the initial camera pose estimate, which aims at minimizing the *sum of reprojection errors* defined by:

$$\mathcal{L}_{\sigma}^{\mathrm{RE}}(\hat{\mathtt{R}}_q, \hat{\mathbf{t}}_q, \mathcal{U}, \mathcal{I}) = \sum_{(\mathbf{u},\mathbf{p})\in\mathcal{I}} \psi_{\sigma}\left(||\mathbf{p} - \mathtt{K}\pi(\hat{\mathtt{R}}_q\mathbf{u} + \hat{\mathbf{t}}_q)||\right) \tag{2.3}$$

where ψ is a parametric robust kernel such as Huber (Huber, 1964), Geman-McClure (Zach & Bourmaud, 2017) or Tukey's biweight (Barron, 2019). Minimizing Eq. (2.3) is a highly non-convex problem and it can be done through a non-linear least-squares log-likelihood maximization (Triggs et al., 1999), which is often solved in practice using a Levenberg-Marquardt optimizer. Note that in this problem we introduce the hyperparameter σ which in practice needs to be tuned depending on the data, as well as the need to choose a robust kernel.

2.3.2.4 Large-scale localization frameworks

Structure-based localization can thus often be summarized as (*i*) finding the set of 2D-to-3D correspondences $\mathcal{U}$ between $\mathtt{I}_q$ and $\mathcal{M}$ (*ii*) computing an initial camera pose estimate by linearly solving PnP inside a RANSAC loop and (*iii*) refining the camera pose estimate by minimizing the non-linear reprojection error. When performing relocalization over extended periods of time or working in a large-scale scenario, this framework can however be improved.

The challenge of *large-scale* localization comes from the fact that finding 2D-to-3D correspondences between a query image and a very large 3D model is both computationally expensive and prone to errors. The method proposed by (Sattler et al., 2011) uses a tree-based correspondence search coupled with a quantized vocabulary to quickly obtain 2D-to-3D matches. Active Search (Sattler et al., 2017a) further improves the correspondence search efficiency by performing both 2D-to-3D and 3D-to-2D prioritized searches using a similar vocabulary tree. City-Scale Localization (CSL) (Svärm et al., 2017) exploits the camera gravity direction and height to estimate an upper and lower bound for the number of inliers of a given correspondence. These bounds are in turn used to discard outlier correspondences and speed up the search of camera pose candidates with RANSAC.

Hierarchical localization: A now popular framework for large-scale localization is to use image retrieval as a 3D model sub-selection routine for faster and more reliable 2D-to-3D matching. State-of-the-art global image descriptors (Arandjelovic et al., 2016; Gordo et al., 2016a, 2017; Noh et al., 2017; Radenovic et al., 2018a,b; Torii et al., 2015) allow for both very fast and robust image retrieval. InLoc (Taira et al., 2018) runs image retrieval on the query image to first identify a set of nearest neighbour reference images. Dense matching is then used to estimate the query pose against every nearest neighbour, and an additional pose verification step based on novel-view synthesis is applied. To be even more efficient and enable city-scale localization, one can start by retrieving the 3D keypoint subsets visible in the top-N reference images (Irschara et al., 2009; Middelberg et al., 2014; Sarlin et al., 2019, 2018). These subsets are likely to also be covisible with the query image (and can optionally be clustered). In HLoc (Sarlin et al., 2019) for each top-ranked subset of 3D keypoints, 2D-to-3D matching is run followed by PnP +RANSAC. The best camera pose is taken as the one that maximizes the final inlier count. The authors propose a single CNN (HF-Net) that predicts hierarchical features to be used for hierarchical localization. Coupled with SuperPoint (Detone et al., 2018) and SuperGlue (Sarlin et al., 2020), the HLoc framework has shown state-of-the-art performance on visual localization benchmarks [2].

[2] https://www.visuallocalization.net/

Other frameworks: To further improve robustness to long-term visual perturbations such as cross-seasonal day-to-night or changes, SMC (Toft et al., 2018) uses semantic maps (which are in theory robust to such perturbations) to compute matches consistency. These scores are in turn used to weight sampling in the RANSAC camera pose solving. Subsequent work (Taira et al., 2019) reuses this idea performing dense pose verification, using both semantic maps and surface normal estimations. In (Piasco et al., 2019), the authors regress the query image dense depth map to provide additional geometric constraints when estimating the camera pose.

2.3.2.5 Direct alignment

The problem of structure-based camera pose estimation (or refinement) can be tackled in a different way. Instead of minimizing a 2D *geometric* reprojection error (see Eq. (2.3)), one can instead minimize a so-called *featuremetric* error. Let us write a dense feature extractor (e.g. a convolutional neural network) $\mathcal{F}_\Theta : \mathbb{R}^{H\times W\times 3} \rightarrow \mathbb{R}^{H\times W\times D}$, and $\mathtt{H}_q = \mathcal{F}_\Theta(\mathtt{I}_q)$ the dense feature response of the query image. Assuming D-dimensional sparse descriptors $\mathcal{D} = \{\mathbf{h}_i\}_{i=1}^M$ have been assigned to the 3D keypoints in the scene using F (e.g. using bilinear interpolation at their reprojected locations in the reference images), the Featuremetric Pose Refinement (FPR) cost function is defined by:

$$\mathcal{L}_\sigma^{\text{FPR}}(\hat{\mathtt{R}}_q, \hat{\mathbf{t}}_q, \mathcal{M}, \mathcal{D}) = \sum_{i=1}^{M} \psi_\sigma \left(||\mathbf{h}_i - \mathtt{H}_q(\mathtt{K}\pi(\hat{\mathtt{R}}_q\mathbf{u}_i + \hat{\mathbf{t}}_q))|| \right) \tag{2.4}$$

where ψ_σ is a parametric robust kernel and the notation $\mathtt{H}_q(\mathtt{K}\pi(\hat{\mathtt{R}}_q\mathbf{u}_i + \hat{\mathbf{t}}_q))$ corresponds to performing a bilinear interpolation in $\mathtt{H}_q$ at location $\mathtt{K}\pi(\hat{\mathtt{R}}_q\mathbf{u}_i + \hat{\mathbf{t}}_q)$. The key difference introduced by Eq. (2.4) is that explicit 2D-to-3D correspondences are no longer required. Instead, when minimizing Eq. (2.4) (e.g. with a second-order such as Gauss-Newton or Levenberg-Marquardt), local gradients are directly provided by the features. In a long-term or wide-baseline scenario, such features can be fairly robust and enable a subpixellic accuracy, breaking free from keypoint detections in the query image. On the other hand with poorly conditioned features or a poor camera pose initialization, this approach is likely to fall in local minima.

Photometric alignment: In the case where F is the Identity, minimizing Eq. (2.4) is a photometric alignment problem (Baker & Matthews, 2004; Lucas & Kanade, 1981), which is typically not robust to changes in illumination, viewpoint or noise and thus suffers from a small basin of convergence. To tackle these issues, various improvements to the camera pose optimizer have been proposed (Engel et al., 2018, 2014), including learning-based

methods (Clark et al., 2018; Lv et al., 2019; Ma et al., 2020; Tang & Tan, 2019; Wang et al., 2018; Xu et al., 2021). Despite these efforts, using image intensities shows little robustness to poor camera pose initializations and is mostly relevant for SLAM applications but not for long-term visual localization.

Featuremetric alignment: Using learned features to improve the robustness and accuracy of FPR has received attention in the past few years (Czarnowski et al., 2017; Park et al., 2017). GN-Net (von Stumberg et al., 2020) and LM-Net (Stumberg et al., 2020) derive differentiable optimizers that provide a loss for end-to-end training of deep models. More recently PixLoc (Sarlin et al., 2021) proposes a coarse-to-fine framework to learn compact image features tailored for featuremetric alignment. It only requires pose supervision and demonstrate promising generalization as well as wide baseline convergence abilities.

2.3.2.6 Discussion

Structure-based localization is very appealing for its efficient leverage of the available 3D geometry. The pre-computed scene geometry provides strong geometric priors and makes this framework competitive for localization in visually stable contexts (Sarlin et al., 2021; Sattler et al., 2018). In long-term scenarios or ambiguous scenes however, keypoint matching algorithms generate significant amount of outlier correspondences, which need to be filtered out prior to solving the camera pose. Eliminating these outlier correspondences do not only bring an additional computational cost, but also hyperparameters (e.g. the choice of a robust kernel and its parametrization) whose optimality is not broadly generalizable. Moreover in very challenging settings, one may end up with very little correspondences to localize with since outlier filtering is done *a posteriori*. This is the main pitfall of structure-based localization, which we propose to tackle in this thesis. More specifically, we argue that defining explicit 2D-to-3D keypoint correspondences followed by outlier removal is detrimental to the task of camera pose estimation. In Chapter 3 we propose a novel paradigm to learn dense matching distributions over whole images instead. In Chapter 5, we will show how despite being not covisible, 3D keypoints that are hidden in query images can still be leveraged to perform camera pose estimation.

2.3.3 End-to-end localization

The recent progress of deep learning models has motivated the research for end-to-end learning of localization. In end-to-end methods, the goal is to learn an implicit representation of the scene such that it can be leveraged to regress a camera pose from a single query image. In this subsection, we review existing end-to-end methods and discuss their

limitations.

2.3.3.1 Absolute pose regression

In the seminal work of PoseNet (Kendall & Cipolla, 2017; Kendall et al., 2015), a deep neural network is trained on reference images to directly regress the absolute camera poses from images. At test-time, this model can be reused to regress the pose of a query image. By design however absolute camera pose regression methods like this one is scene-specific. Thus relocalizing on a different scene will imply learning a new regression model, which causes scalability issues. In addition, PoseNet is not very accurate (Schönberger et al., 2017), and suffers from generalization issues when query images are captured from viewpoints differing from reference image trajectories (Sattler et al., 2019). Lastly the output of PoseNet is not interpretable, as there is little explainability to the network reasoning. More intricate methods like (Walch et al., 2017) have been developed but still lack accuracy and interpretability.

2.3.3.2 Relative pose regression

A natural alternative to absolute pose regression is relative pose regression. The goal of relative pose regression is to estimate the relative position and rotation between a query and reference image (e.g. identified using image retrieval). RelocNet (Balntas et al., 2018) is a model that performs such relative pose regression, and is trained with relative pose supervision. While results are encouraging they still significantly underperform structure-based localization results (Sattler et al., 2019). RelocNet is also implicitly trained up to a scaling factor, which means that a model trained on indoor images will not generalize to outdoor images. Similar work has been proposed (Chen et al., 2021; Ding et al., 2019; Laskar et al., 2017; Zhou et al., 2020b) but so far suffers from the same limitations.

2.3.3.3 Scene-coordinate regression

Instead of aiming to directly regress the query camera pose a richer, intermediate objective is to predict the 3D coordinates of the query image content. DSAC (Brachmann et al., 2017) performs such scene-coordinate regression and solves for the camera pose through a differentiable camera hypothesis selection scheme. While follow-up work (Brachmann & Rother, 2018, 2019a, 2021) brings improvement to the method, scaling to new scenes requires a cumbersome retraining or finetuning which sometimes fails to converge (Cavallari et al., 2019; Sattler et al., 2018; Taira et al., 2018). The work of (Li et al., 2020b) derives a hierarchical approach to scene coordinate regression which improves upon these limitations, but is still inherently limited by its scene-specificity. Lastly SANet (Yang

et al., 2019a) regresses dense coordinate maps in a scene-agnostic way, based on RGB images and an SfM 3D model.

2.3.3.4 Discussion

While coming up with a self-contained deep learning model to solve localization is appealing, it appears end-to-end methods have yet to outperform structure-based localization approaches. The lack of accuracy and scalability of end-to-end methods seem to indicate that an explicit representation of the world (such as a 3D point cloud) remains a lot more compelling than an implicit one.

Chapter 3

Sparse-to-Dense Matching

3.1 Introduction

As presented in Chapter 2, obtaining pixel-perfect correspondences in long-term scenarios where extreme visual changes can appear remains an unsolved problem (Sattler et al., 2018; Taira et al., 2018). In particular, illumination (e.g. daytime to nighttime), cross-seasonal and structural changes are very challenging factors for keypoint matching. While recent learning-based keypoint matching methods have shown promising results in providing robust correspondences under challenging conditions, they are often limited in terms of precision, due to both errors in detection and description. In this chapter, we aim at rethinking the traditional keypoint matching approaches to improve their robustness and accuracy, and introduce the sparse-to-dense matching paradigm.

In Section 3.2 we first distinguish three different matching paradigms and discuss what motivates a spase-to-dense matching approach for 2D-to-3D matching in a structure-based visual localization framework. In Section 3.3, we give a brief overview of the relevant localization-related work for this chapter, which concerns structure-based, retrieval-based and hierachical localization. In Section 3.4 we present a weakly supervised approach to sparse-to-dense matching, coined S2DHM. S2DHM demonstrates that when applied to absolute camera pose estimation the sparse-to-dense paradigm enables much higher inlier counts in RANSAC solvers compared to sparse-to-sparse methods. In Section 3.5 we present S2DNet, a strongly supervised method improves upon S2DHM by learning to perform highly accurate correspondence retrieval. Lastly we report quantitative and qualitative experiments and visualizations in Section 3.6. We show that at the time of publication both methods were highly competitive on several long-term visual localization datasets.

3.2 Matching paradigms

Given a pair of images $\mathtt{I}$ and $\mathtt{I}'$, we can distinguish three paradigms for keypoint matching, which are illustrated in Fig. 3.1.

3.2.1 The sparse-to-sparse paradigm

A traditional and very commonly used paradigm for keypoint matching between two images consists in detecting a set of keypoints in both images followed by a description stage in each image. Sparse sets of keypoints and their descriptors are then matched using for instance a simple nearest neighbours algorithm. This *sparse-to-sparse* matching approach has the main advantage of being both computationally and memory efficient: a given image can be summarized by a list of sparse keypoint detections and descriptors, and matching consists in performing sparse-to-sparse comparisons. Historically, limited

computing power has motivated the development of both lightweight and robust keypoint detectors and descriptors. For this reason to this day, sparse-to-sparse matching is the most widely used keypoint matching approach.

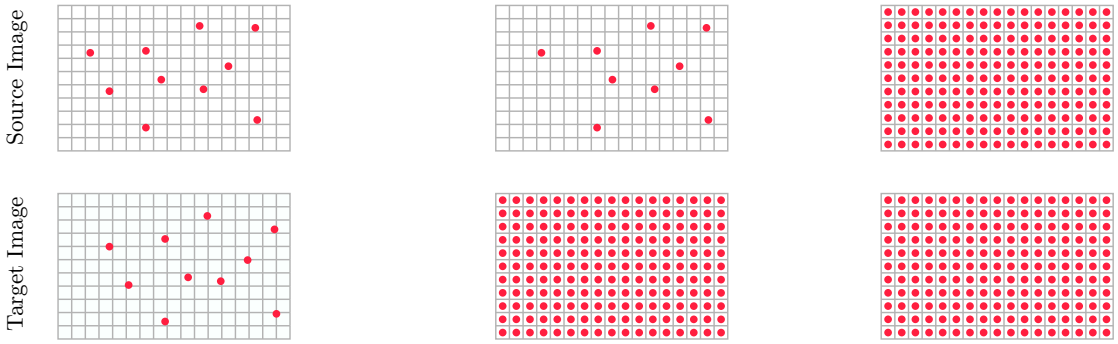
This paradigm however makes underlying assumption on detection, namely that keypoint detection is repeatable (see Section 2.2.1) and descriptors are both discriminative and invariant to common appearance perturbations (see Section 2.2.2). In long-term scenarios or when working with challenging image pairs however this is very challenging (see Fig. 2.1) even for advanced deep learning models (Dusmanu et al., 2019; Revaud et al., 2019; Tyszkiewicz et al., 2020). We argue that the symmetric nature of sparse-to-sparse methods is a strong design limitation which prevents further improvements in keypoint matching accuracy. Instead, we advocate for a transfer of information from one image to the other in an asymmetrical way prior to matching, to help solve the keypoint repeatability problem.

3.2.2 The sparse-to-dense paradigm

The *sparse-to-dense* matching paradigm which we introduced in (Germain et al., 2019) breaks this symmetry. The idea of sparse-to-dense matching is to perform keypoint detection in either image, and search for its correspondent in the other *exhaustively*. We will often refer to the image on which keypoint detection is applied as the *source* image, and the image in which we perform this exhaustive search as the *target* image.

In other words under the sparse-to-dense matching paradigm, every pixel in the target image is a candidate keypoint correspondent. This has strong implications, as given any covisible keypoint detection in the source image, we now know that there exists a corresponding pixel location (from which we can search from) in the target image that precisely matches that detection. This is often not the case in sparse-to-sparse matching, as keypoint detections lack repeatability under challenging conditions. As a result keypoint matching can be formulated as a pure feature learning problem, in which the goal is to identify correspondences *from* the source *to* the target.

Sparse-to-dense matching is particularly well suited for the problem of structure-based visual localization, for which we are only interested in matching a handful of 3D keypoints (which can also be seen as 2D reprojections of triangulated keypoints in a reference image) against a query image. Keypoint correspondences from other 2D locations in the reference image are uninformative to solve for the absolute query camera pose. In Section 3.5 we will show how casting the keypoint matching problem as a pure classification task can lead to improved matching accuracy.



(a) Sparse-to-sparse (b) Sparse-to-dense (c) Dense-to-dense

Figure 3.1: **Keypoint matching paradigms**: For a given (source, target) image pair, we show in red the keypoint locations that can be matched on both images for each paradigm.

3.2.3 The dense-to-dense paradigm

The *dense-to-dense* matching paradigm introduced in (Rocco et al., 2018) and later reused in (Sun et al., 2021) goes a step further and avoids performing keypoint detection altogether. Instead, this paradigm seeks to process both the source and target image pair jointly in order to return putative correspondences in a single forward pass.

This approach implies a joint processing of both images, however in practice this often comes at a strong computational cost (see Section 2.2.3.3) and diminished keypoint accuracy. In addition we argue seeking to find correspondences across every possible pixel of two images is rarely beneficial to practical computer vision tasks, and can thus be a waste of resources. For instance seeking keypoint matches across textureless areas (e.g. walls, sky, etc.) is prone to noisy predictions and will probably not be informative to downstream applications like structure-based localization.

Another implication of dense-to-dense matching is that both the source and target images have to be processed jointly, and very little computation can be placed offline. In the case of hierarchical localization for instance it is desirable to pre-compute (offline) 3D keypoint descriptors using reference images, so that the only feature extraction to perform at test-time is on the query image. With dense-to-dense approaches however when matching a query image against N nearest neighbours, every reference image pairs has to be re-processed against every new query image which limits scalability.

3.2.4 Tradeoffs

The main appeal for sparse-to-sparse matching is efficiency, although it comes at the cost of facing the keypoint repeatability problem. Sparse-to-dense matching aims at solving this issue, but requires performing dense operations over the target image. While the idea

of performing an exhaustive search over an image can seem overly computationally expensive, we will show in this chapter that by resorting to GPUs sparse-to-dense matching can actually be parallelized efficiently and bring little computational overhead compared to sparse-to-sparse approaches. Dense-to-dense matching methods on the other hand perform bi-directional dense operations and are often a lot more expensive (quadratic cost in feature resolution). In the case of visual localization such methods also perform uninformative operations, which are hardly scalable for large-scale scenarios. We argue that sparse-to-dense matching is an appropriate middleground between keypoint matching accuracy and computational cost.

3.3 Localization-related work

In this section we provide a brief overview of localization-related work that will serve as baseline in our experiments.

3.3.1 Structure-Based Localization

Structure-based methods regress the full 6 DoF camera pose of query images using direct 2D-3D correspondences. Such methods work by first acquiring a point-cloud model of the scene through *SfM*, and computing local feature descriptors like SIFT (Lowe, 2004), SuperPoint (Detone et al., 2018) or R2D2 (Revaud et al., 2019). These descriptors are in turn used to obtain 2D-to-3D correspondences, and the predicted camera can usually be inferred from those matches using a Perspective-n-Point (PnP) solver (Bujnak et al., 2010; Haralick et al., 1994; Kukelova et al., 2013; Lepetit et al., 2008) inside a RANSAC (Fischler & Bolles, 1981; Sattler et al., 2014) loop.

In consistent daytime conditions, such methods achieve very competitive results (Sattler et al., 2017a, 2018; Svärm et al., 2017; Walch et al., 2017). However, they rely heavily on the accuracy and robustness of the local 2D-3D correspondences. Research in structure-based approaches mostly focuses on improving descriptor matching efficiency (Choudhary & Narayanan, 2012; Larsson et al., 2016; Li et al., 2010; Lim et al., 2012; Lynen et al., 2015; Sattler et al., 2017a), speed (Donoser & Schmalstieg, 2014; Heisterklaus et al., 2014) and robustness (Li et al., 2016; Sattler et al., 2015, 2016; Svärm et al., 2017; Svärm et al., 2014; Zeisl et al., 2015). Yet, under strong condition changes, failures in direct matching start to appear and damage the localization performance (Sattler et al., 2018). In order to improve the robustness of local feature descriptors and thus increase long-term localization performance, recent methods have used semantic reasoning (Toft et al., 2018). Indeed, semantic maps are to some extent condition-invariant, and can enhance either the feature matching stage (Arandjelović & Zisserman, 2014; Kobyshev et al., 2014; Schön-

berger et al., 2017; Singh & Kosecka, 2016) or the pose estimation stage (Toft et al., 2018). While being accurate at small scale, feature-based methods bottleneck is scalability. In large-scale scenarios, both the construction of precise 3D models (and their maintenance) and local feature-matching is challenging and expensive (Sattler et al., 2017b).

3.3.2 Structure-free localization

In image-based localization methods, accuracy is traded-off for scalability. The scene is modeled as an image database containing ground-truth 6-DoF pose annotations. To infer the pose of a visual query, one can use compact image-level representations to retrieve the top-ranked image from the database and use their labels as pose approximation (Chen et al., 2011; Sattler et al., 2017b; Zamir & Shah, 2010; Zhang & Kosecka, 2006) (*retrieval-based* localization). The need for ground-truth 3D geometry is alleviated, and this method can easily generalize to large-scale environments.

To obtain robust global image descriptors, one can aggregate local features in the image into a fixed-size representation. VLAD (Arandjelovic & Zisserman, 2013) is a popular descriptor, computed by summing and concatenating many descriptors for affine-invariant regions. DenseVLAD (Torii et al., 2015) reformulates the VLAD architecture by densely sampling RootSIFT (Perdoch et al., 2009) descriptors in the image. Recent learning-based variants cast the task of image retrieval as a metric learning problem. NetVLAD (Arandjelovic et al., 2016) defines a differentiable VLAD layer as the final activation of a siamese network. Other activations layers (Babenko & Lempitsky, 2015; Gordo et al., 2017; Kalantidis et al., 2016; Radenovic et al., 2018b; Razavian et al., 2014; Tolias et al., 2015) coupled with siamese or triplet architectures, have shown to deliver competitive results for the task of image-retrieval (Radenovic et al., 2018a). SOLAR (Ng et al., 2020) uses a second-order loss similar to SOSNet (Tian et al., 2019) to improve the performance of global image descriptors.

In a very large database, unsupervised descriptor compression like PCA (Jégou & Chum, 2012) or Product Quantization (PQ) (Jégou et al., 2011) enables efficient approximate nearest-neighbor search with little loss in performance (Gordo et al., 2017).

Other image-based methods include end-to-end learning approaches, which avoid using explicit feature matching altogether and leverages CNNs to learn robust representations (Brachmann et al., 2017; Brachmann & Rother, 2018, 2019a, 2021; Bui et al., 2018; Kendall & Cipolla, 2017). These methods are however often hard to initialize (Sattler et al., 2018; Schönberger et al., 2017), struggle with large environments (Sattler et al., 2018) and/or provide overall limited performance (Balntas et al., 2018; Kendall et al., 2015; Walch et al., 2017).

3.3.3 Hierarchical Localization

For the problem of long-term localization, where strong appearance changes can occur because of the light or season differences, global descriptors have shown to provide robust pose initialization under strong visual changes (Germain et al., 2018; Sarlin et al., 2019; Sattler et al., 2018). Still, the main bottleneck of retrieval-based localization is their lack of accuracy in sparsely sampled databases. Several schemes can be implemented to refine the coarsely estimated pose. For instance, view synthesis (Taira et al., 2018; Torii et al., 2015) artificially generates intermediate samples, relative pose regression (Balntas et al., 2018; Taira et al., 2018) acts as a separate refinement step and multi-image methods (Balntas et al., 2018; Zamir & Shah, 2010; Zhang & Kosecka, 2006) combine the top ranked images to improve pose accuracy.

Image-retrieval can also be seen as a way to obtain a query's coarse location, before running a structure-based pose refinement algorithm. By doing so, 2D-3D matching is only run on a subset of the whole point cloud, leading to competitive results at small computational costs (Irschara et al., 2009; Middelberg et al., 2014; Sarlin et al., 2019, 2018). For its combination of accuracy, efficiency and scalability this is the hierarchical localization framework we will be considering in this chapter.

3.4 Sparse-to-Dense Hypercolumn Matching for Long-Term Visual Localization

In this section, we will describe our first attempt at learning to perform sparse-to-dense matching through the weak supervision of global image descriptors. More specifically we will report results that were presented in (Germain et al., 2019), applied to long-term visual localization. In this work, we introduce a method called S2DHM that reuses dense feature representations extracted from a CNN trained with weak supervision to run sparse-to-dense matching. We run S2DHM within a hierarchical localization pipeline and show it allows us to obtain both more numerous and reliable keypoint correspondences.

3.4.1 Problem statement

Let us remind the typical hierarchical localization framework. We assume that a database of registered reference images is available. More precisely, for each reference image $\mathtt{I}_i^{\text{ref}}$ of the database, we assume that the following is available:

- A normalized global image descriptor $\mathbf{g}_i$ computed as explained in Section 3.4.2, which we will use for the retrieval step.

- the calibration matrix $\mathtt{K}_i$ and the absolute camera pose $\mathtt{P}_{iw}$ expressed in the world coordinate system;
- a set of N_i 2D keypoints $\{\mathbf{p}_j^i\}_{j=1...N_i}$ detected using SuperPoint (Detone et al., 2018);
- the descriptor $\mathbf{h}_j^i$ for each feature point $\mathbf{p}_j^i$ computed as explained in Section 3.4.3;
- the 3D coordinates $\mathbf{u}_j^i$ of each feature point $\mathbf{p}_j^i$.

Given a query image $\mathtt{I}_q$ with known calibration matrix $\mathtt{K}_q$, and this database, we aim to predict the camera pose $\mathtt{P}_{qw}$.

3.4.2 Hierarchical framework

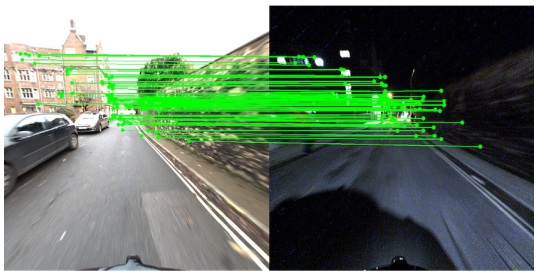
When performing localization in large-scale environments, matching a set of 2D detections with a large number of 3D keypoints can be difficult (Sattler et al., 2018). As explained in Section 3.3.3, one way to reduce the set of 3D points to match the image keypoints against is to first perform image retrieval. The returned top-ranked images in the database provide us with a subset of the large 3D point cloud for which performing local feature matching is much more efficient. Following previous work on global image matching (Arandjelovic et al., 2016; Babenko & Lempitsky, 2015; Gordo et al., 2017; Kalantidis et al., 2016; Radenovic et al., 2018b; Razavian et al., 2014; Tolias et al., 2015), we use a Siamese network to learn both compact and discriminative global image descriptors. In this work, we opt for the popular NetVLAD (Arandjelovic et al., 2016) pooling layer with a VGG-16 (Simonyan & Zisserman, 2014) backbone. We train our model using the same contrastive loss as (Arandjelovic et al., 2016). We define positive and negative labels $l(\mathtt{I}_i, \mathtt{I}_j) \in \{0, 1\}$ for pairs of images, based on the presence or absence of co-visibility between images. Once trained, the network provides a global descriptor $\mathbf{g}_i$ for each reference image, which can be stored offline. At test time, given a query image $\mathtt{I}_q$, we compute its descriptor $\mathbf{g}_q$ and retrieve its k nearest neighbors by computing the Euclidean distance between $\mathbf{g}_q$ and each stored descriptor $\mathbf{g}_i$. Such top-ranked images provide coarse camera poses which are sufficient to estimate a query's approximate location (Sattler et al., 2018).

3.4.3 S2DHM: Sparse-to-Dense Hypercolumn Matching

In order to obtain a more accurate camera pose estimate, we make use of the local 3D point clouds fetched at the image retrieval step. For each of the k nearest neighbors, we establish 2D-to-3D correspondences and subsequently solve for the pose using a PnP (Bujnak et al., 2010; Haralick et al., 1994; Kukelova et al., 2013) solver inside a RANSAC (Fischler & Bolles, 1981). The method we use to establish these correspondences is our main contribution, and we describe it below.

Motivation. As previously illustrated, sparse-to-sparse keypoint matching suffers from

(a) **Sparse-to-sparse matching** using SuperPoint (Detone et al., 2018) detections and two different descriptors : (left) SuperPoint descriptors [4 inliers], (right) our hypercolumn descriptors [5 inliers]



(b) **Sparse-to-dense matching** using Superpoint detections in the left image only and hypercolumn descriptors [87 inliers]

Figure 3.2: **Top images**: Despite recent progress repeatably detecting keypoint on two images captured under very different conditions remains extremely challenging, as shown on these images from the RobotCar (Maddern et al., 2017) dataset. **Bottom image**: Our key contribution is to show that it is much more robust to perform keypoint detection in only one image, and to search for their correspondents exhaustively in the other. This exhaustive search can be performed very efficiently using convolutional operations. Using the 3D locations of the detected keypoints in reference images, we can then compute the camera pose. We report the number of inlier matches found by PnP+RANSAC.

the lack of repeatability in keypoint detections (see Fig. 2.1). In even more extreme cases such as day-to-night changes, the effect becomes even more visible as shown in Fig. 3.2a: the lack of reliable keypoint detections in query images prevents us from finding sufficient inlier correspondences for a reliable pose estimate. We thus propose to skip the detection step in the query image and rather exhaustively search for correspondents in the query image using intermediate features coming from our image retrieval network.

Hypercolumn extraction. In order to perform robust matching, we rely on intermediate convolutional features that were used to compute the global query image descriptor. For each query image, we extract intermediate features from the VGG-16 network trained for image retrieval, and aggregate them in order to obtain a dense and rich representation of the image. We extract features from the layers `conv_3_3`, `conv_4_1`, `conv_4_3`, `conv_5_1`, `conv_5_3`. These representations are referred to as "hypercolumns" (Har-

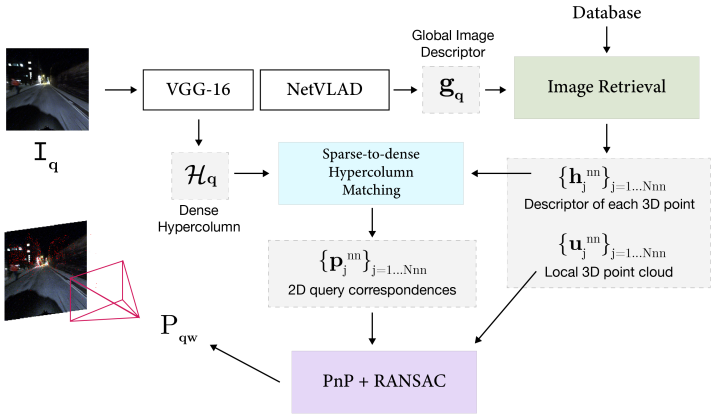


Figure 3.3: **Overview of our hierarchical localization pipeline.** Given a query image, we compute dense hypercolumns using VGG-16 (Simonyan & Zisserman, 2014) and a global image descriptor using NetVLAD (Arandjelovic et al., 2016). The extracted dense intermediate features are trained for image retrieval on images undergoing day-to-night changes. For each 3D keypoint, we extract sparse hypercolumns in reference images and exhaustively search for their correspondents in the query dense representation using correlation maps. This results in numerous robust correspondences suitable to perform PnP+RANSAC across changing conditions.

iharan et al., 2014). Each intermediate layer is upsampled using bilinear interpolation to match the resolution $W_{\mathcal{H}} \times H_{\mathcal{H}}$ of the earliest layer, before being concatenated along the channel axis and L2-normalized. We define the resulting hypercolumns for the query image $\mathtt{I}_q$ as $\mathcal{H}_q \in \mathbb{R}^{W_{\mathcal{H}} \times H_{\mathcal{H}} \times D}$. For each reference image $\mathtt{I}_i$, we are only interested in descriptors located at feature points. We thus only store in the database the hypercolumns at locations $\{\mathbf{p}_j^i\}_{j=1...N_i}$. We denote $\mathcal{S}_i = \{\mathbf{h}_j^i\}_{j=1..N_i}$ this set of sparse descriptors, where $\mathbf{h}_j^i \in \mathbb{R}^{1 \times 1 \times D}$.

Sparse-to-dense matching. To find correspondences between the set of sparse descriptors (from the reference image) $\mathcal{S}_i$ and the dense hypercolumns $\mathcal{H}_q$, we perform a simple dot product. These dot products can be efficiently implemented with a 1×1 convolution. We define the resulting cross-correlation map as $\breve{\mathtt{C}}_{q,j}^i = \mathcal{H}_q * \mathbf{h}_j^i \in \mathbb{R}^{W_{\mathcal{H}} \times H_{\mathcal{H}}}$. An illustration of our feature extraction process is shown in Fig. 3.4. To retrieve the final 2D keypoints in the query image, we first fetch the global maximum of the cross-correlation map and upsample the retrieved coordinates to match the query resolution. Consequently, this sparse-to-dense matching step always gives us N_i 2D-to-3D correspondences. The qualitative results of Fig. 3.2b shows that the combination is of hypercolumn descriptors with sparse-to-dense matching enables a significant increase in inlier correspondences. An overview of our method is illustrated in Fig. 3.3. As will be shown in experiments, thanks to the popularization and development of GPUs this exhaustive search can be done with

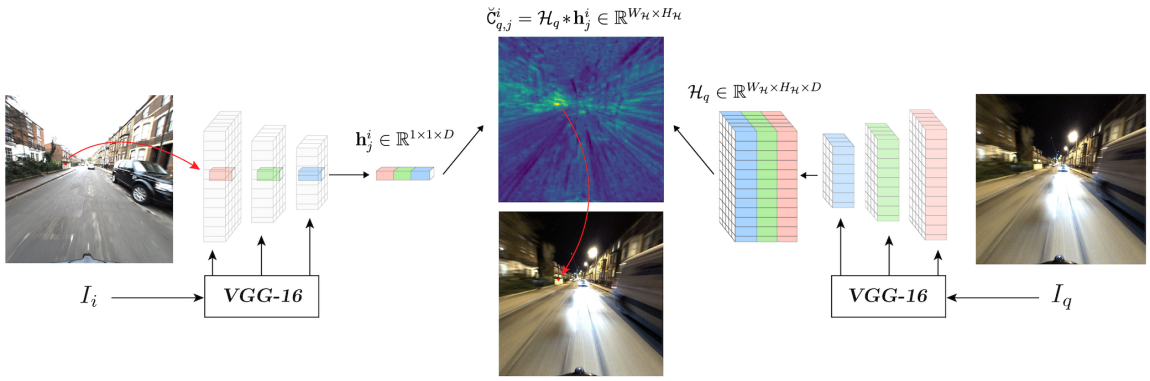


Figure 3.4: **Sparse-to-dense feature matching using hypercolumns.** For each detection $\mathbf{p}^i_j$ in the reference image $\mathtt{I}_i$ retrieved for query image $\mathtt{I}_q$, we extract a hypercolumn descriptor $\mathbf{h}^i_j$, which we cross-correlate against the dense hypercolumn $\mathcal{H}_q$. We then define the correspondent location of $\mathbf{p}^i_j$ in $\mathtt{I}_q$ as the image location of the maximum value in the resulting correlation map $\breve{\mathtt{C}}^i_{q,j} = \mathbf{h}^i_j * \mathcal{H}_q$.

a small computational overhead compared to other learning-based methods.

Ratio Test. Some detections in the reference image may fall in unreliable image regions (e.g. with repetitive patterns), or in areas that are occluded in the query image. When taking the argmax coordinate of the correlation map this may lead to erroneous correspondences. To discard matches with strong ambiguity, we apply a ratio test similar to (Lowe, 2004) which is defined as follows. For the cross-correlation map $\breve{\mathtt{C}}^i_{q,j}$, let $\bar{\mathtt{C}}^i_{q,j} \in \mathbb{R}^{(W_\mathcal{H}.H_\mathcal{H})}$ be the flattened and sorted by decreasing order map. For a 2D-to-3D match to be retained, we apply the following rule:

$$\frac{\bar{\mathtt{C}}^i_{q,j}[0]}{\bar{\mathtt{C}}^i_{q,j}[f \times (W_\mathcal{H} \times H_\mathcal{H})]} > \alpha, f \in [0;1]\,. \tag{3.1}$$

In practice, we use $\alpha = 0.9$, and adapt the factor f to the different datasets. Finding the value of $\bar{\mathcal{C}}^i_{q,j}[f \times (W_\mathcal{H} \times H_\mathcal{H})]$ actually does not require sorting the whole array, and adds negligible overload to the computational cost.

Qualitative and quantitative results for this approach will be presented in Section 3.6. Prior to that, we derive a strongly supervised alternative approach to perform sparse-to-dense matching.

3.5 S2DNet: Learning image features for accurate sparse-to-dense matching

In this section we tackle the problem of learning to perform sparse-to-dense matching using strong supervision. Instead of relying on global image descriptors to learn local features, we will leverage ground-truth pixel correspondences across images to obtain more accurate and peaked correlation matps. We introduce S2DNet (Germain et al., 2020), a feature matching pipeline, designed and trained to efficiently establish both robust and accurate correspondences. By leveraging the sparse-to-dense matching paradigm, we cast the correspondence learning problem as a supervised classification task to learn to output highly peaked correlation maps.

3.5.1 Motivation

While our first attempt at sparse-to-dense matching benefits from a simple, weakly supervised loss based on image retrieval, this loss does not locally constrain intermediate features to produce highly accurate correlation map. The accuracy of the 2D-to-3D correspondences plays a major role in the performance of PnP-based visual localization algorithms.

The noise perturbation experiment of Figure 3.5 (left) shows the highly damaging impacts of errors of just a few pixels on the final camera pose estimate. In order to improve the accuracy of keypoint correspondences, we propose in this section a novel architecture called S2DNet which is tailored for accuracy.

As illustrated in Fig. 3.5 (right), S2DNet is able to significantly improve keypoint matching accuracy compared to sparse-to-sparse methods. Instead of losses operating at a global descriptor level, we train S2DNet to learn both accurate and discriminative correlation maps. By casting the feature matching problem as a classification problem, we give rich feedback on correspondences errors at training time.

3.5.2 Method

In this subsection we introduce and describe our sparse-to-dense feature-matching pipeline, which we call S2DNet.

3.5.2.1 Strongly supervised loss

Given an image pair $(\mathtt{I}_\mathrm{A}, \mathtt{I}_\mathrm{B})$, our goal is to obtain a set of 2D-to-2D correspondences which we write as $\{(\mathbf{p}_j^\mathrm{A}, \mathbf{p}_j^\mathrm{B})\}_{j=1}^N$. Let us consider the case where a feature detector (*e.g.* the SuperPoint detector (Detone et al., 2018)) has been applied on image A, producing

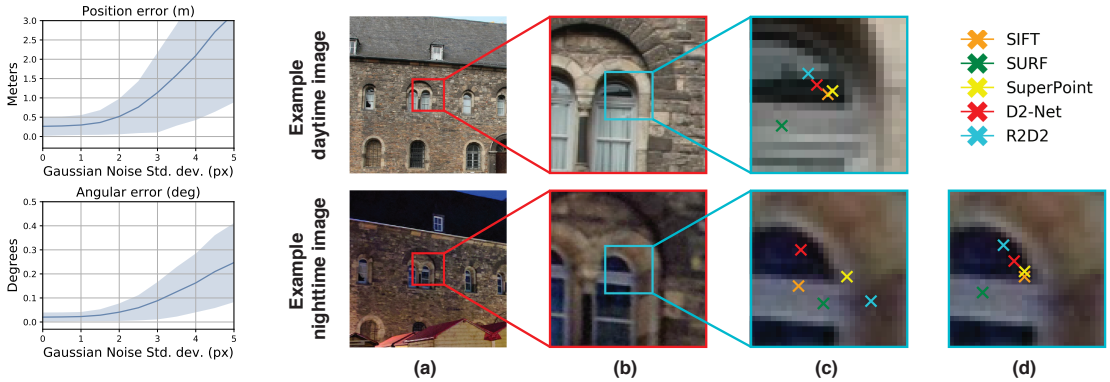


Figure 3.5: **Learning accurate correspondences.** On the left, we report the impact of adding a gaussian noise of increasing variance on ground-truth 2D-3D correspondences for the task of visual localization, on Aachen Day-Night (Sattler et al., 2018, 2012) images. This experiment highlights the importance of having very accurate correspondences, as offsets of a few pixels can lead to localization errors of several meters. Yet as shown on the right, sparse-to-sparse methods fail to make such accurate predictions. We show in (a) and (b) local regions of interest for a day-night image pair. In (c) [top], we display the keypoint detections being the nearest to the center of the patch in the daytime image for each detector; [bottom] we show the closest correspondent detected keypoints for each detector in the nighttime image. In (d), we show the correspondent image locations found by S2DNet in the nighttime image for daytime keypoint detections. S2DNet manages to find much more accurate correspondences than sparse-to-sparse methods.

a set of N keypoints $\{\mathbf{p}_j^{\mathrm{A}}\}_{j=1}^{N}$. In this case, the feature matching problem reduces to a *sparse-to-dense* matching problem of finding a correspondent $\mathbf{p}_j^{\mathrm{B}}$ in image B for each detection $\mathbf{p}_j^{\mathrm{A}}$. We propose to cast this correspondence learning problem as a supervised classification task by restricting the set of admissible locations to the pixel coordinates of $\mathtt{I}_{\mathrm{B}}$. This leads to the following categorical distribution:

$$p\left(\mathbf{p}_j^{\mathrm{B}} | \mathbf{p}_j^{\mathrm{A}}, \mathtt{I}_{\mathrm{A}}, \mathtt{I}_{\mathrm{B}}, \Theta\right) = \frac{\exp\left(\breve{\mathtt{C}}_j[\mathbf{p}_j^{\mathrm{B}}]\right)}{\sum_{\mathbf{q}\in\Omega} \exp\left(\breve{\mathtt{C}}_j[\mathbf{q}]\right)}\,, \tag{3.2}$$

where $\breve{\mathtt{C}}_j$ is a correlation map of the size of $\mathtt{I}_{\mathrm{B}}$ produced by S2DNet and Ω is the set of pixel locations of $\mathtt{I}_{\mathrm{B}}$. S2DNet takes as input $\mathbf{p}_j^{\mathrm{A}}$, $\mathtt{I}_{\mathrm{A}}$, $\mathtt{I}_{\mathrm{B}}$ and its parameters Θ. Equation 3.2 describes the likeliness of a pixel $\mathbf{p}_j^{\mathrm{A}}$ in $\mathtt{I}_{\mathrm{A}}$ to correspond to pixel $\mathbf{p}_j\mathrm{B}$ in $\mathtt{I}_{\mathrm{B}}$.

3.5.2.2 S2DNet

We introduce S2DNet, a pipeline built specifically to perform sparse-to-dense matching which we illustrate in Figure 3.6. Given a pair of images $(\mathtt{I}_{\mathrm{A}}, \mathtt{I}_{\mathrm{B}})$, we apply a convolutional backbone $\mathcal{F}$ on both images using shared network weights *i.e.* $\{\mathtt{H}_m^{\mathrm{A}}\}_{m=1}^{M} = \mathcal{F}\left(\mathtt{I}_{\mathrm{A}}; \Theta\right)$ and

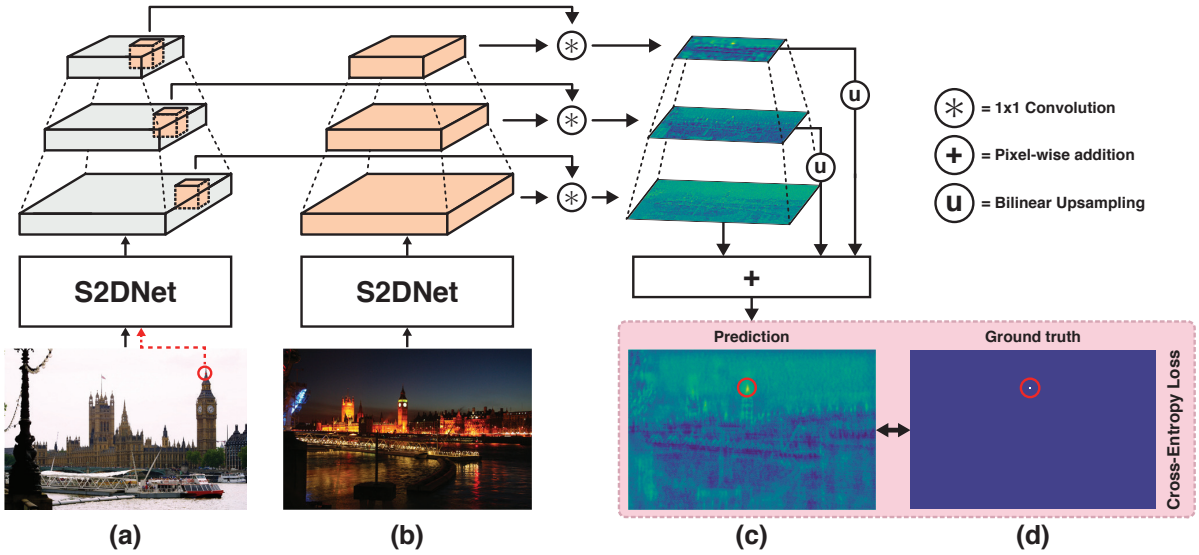


Figure 3.6: **S2DNet feature matching pipeline overview.** Given an image and a set of detections coming from an off-the-shelf keypoint detector (a), we first extract a set of sparse multi-level descriptors with S2DNet. We then compute dense feature maps for a covisible image (b), and compute multi-level correlation maps (c), which we aggregate using bilinear upsampling and addition. Correspondences can the be retrieved using a simple argmax operator. We explicitly train S2DNet to generate accurate and discriminative correlation maps using a supervised classification approach (d).

$\{\mathtt{H}^{\mathrm{B}}_m\}_{m=1}^{M} = \mathcal{F}(\mathtt{I}_{\mathrm{B}}; \Theta)$, where $\{\mathtt{H}^{\mathrm{A}}_m\}_{m=1}^{M}$ and $\{\mathtt{H}^{\mathrm{B}}_m\}_{m=1}^{M}$ correspond to intermediate feature maps extracted at multiple levels (see Fig. 3.7). Θ denotes the parameters of $\mathcal{F}$. While the earlier layers encode little semantic meaning, they preserve high-frequency local details which is crucial for retrieving accurate keypoints. Conversely max-pooling layers reduce the feature map resolutions but benefit from a wider receptive field and thus context.

For each detected keypoint $\mathbf{p}^{\mathrm{A}}_j$ in $\mathtt{I}_{\mathrm{A}}$, we extract a set of sparse descriptors in the dense intermediate feature maps $\mathtt{H}^{\mathrm{A}}_m$ and compute the dense correlation map $\breve{\mathtt{C}}_j$ against $\mathtt{H}^{\mathrm{B}}_m$, by processing each level independently, in the following way:

$$\breve{\mathtt{C}}_j = \sum_{m=1}^{M} \mathcal{U}\left(\mathtt{H}^{\mathrm{A}}_m\left[\mathbf{p}^{\mathrm{A}}_{j,m}\right] * \mathtt{H}^{\mathrm{B}}_m\right) , \tag{3.3}$$

where $\mathcal{U}$ refers to the bilinear upsampling operator to $\mathtt{I}_{\mathrm{B}}$ resolution, $\mathbf{p}^{\mathrm{A}}_{j,m}$ corresponds to downscaling the 2D coordinates $\mathbf{p}^{\mathrm{A}}_j$ to the resolution of $\mathtt{H}^{\mathrm{A}}_m$, and $*$ is the 1×1 convolution operator.

3.5.2.3 Training-time

While state of the art approaches employ either a local contrastive or a listwise ranking loss (Dusmanu et al., 2019; Ono et al., 2018; Revaud et al., 2019) to train their network, we directly optimize for the task of sparse-to-dense correspondence retrieval by maximizing the log-likelihood in eq.(3.2) which results in a single multi-class cross-entropy loss. From a practical point of view, for every training sample, this corresponds to computing the softmax of the correlation map [1] and evaluate the cross-entropy loss using the ground truth correspondence $\mathbf{p}_n^{\mathrm{B}}$. This strongly penalizes wrong predictions, regardless of their closeness to the ground-truth, forces the network to generate highly localized and peaked predictions and helps computing accurate correspondences.

3.5.2.4 Test-time

At test-time, to retrieve the correspondences in $\mathtt{I}_{\mathrm{B}}$, we proceed as follows for each detected keypoint $\mathbf{p}_j^{\mathrm{A}}$:

$$\mathbf{p}_j^{\mathrm{B}*} = \underset{\mathbf{p}_j^{\mathrm{B}}}{\operatorname{argmax}}\; p\left(\mathbf{p}_j^{\mathrm{B}} | \mathbf{p}_j^{\mathrm{A}}, \mathtt{I}_{\mathrm{A}}, \mathtt{I}_{\mathrm{B}}, \Theta\right) = \underset{\mathbf{p}}{\operatorname{argmax}}\; \breve{\mathtt{C}}_j\left[\mathbf{p}\right] \;, \tag{3.4}$$

where $\breve{\mathtt{C}}_j = \mathrm{S2DNet}(\mathbf{p}_j^{\mathrm{A}}, \mathtt{I}_{\mathrm{A}}, \mathtt{I}_{\mathrm{B}}; \Theta)$. By default, S2DNet does not apply any type of filtering and delivers one correspondence for each detected keypoint in the source image. Since we do not explicitly deal with covisibility issues, we filter out some ambiguous matches if the following condition is not satisfied:

$$p\left(\mathbf{p}_j^{\mathrm{B}*} | \mathbf{p}_j^{\mathrm{A}}, \mathtt{I}_{\mathrm{A}}, \mathtt{I}_{\mathrm{B}}, \Theta\right) > \tau \;, \tag{3.5}$$

where τ is a threshold between 0 and 1.

3.5.2.5 Architecture details

As S2DHM, we use a VGG-16 (Simonyan & Zisserman, 2014) architecture as our convolutional backbone. We place our intermediate extraction points at three levels, in `conv_1_2`, `conv_3_3` and `conv_5_3`, after the ReLU activations. Note that `conv_1_2` comes before any spatial pooling layer, and thus preserves the full image resolution. To both help with the convergence and reduce the final descriptors sizes, we feed these intermediate tensors to adaptation layers. They consist of two convolutional layers and a final batch-normalization (Ioffe & Szegedy, 2015) activation, with an output size of 128 channels. An illustration of our architecture can be seen in Fig. 3.7.

[1] In the next chapter, we will refer to the softmax of a correlation map as a *correspondence map*

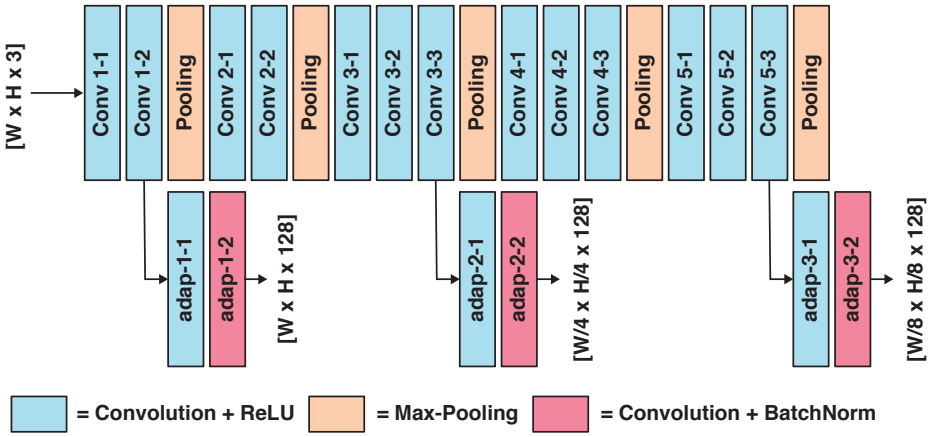


Figure 3.7: **S2DNet: Architecture overview.** We feed images through a standard VGG-16 (Simonyan & Zisserman, 2014) backbone, and set three extraction points to process intermediate features. These features are sent to small, adaptation layers which help with the convergence and provide more condensed descriptors.

3.5.3 Differences with S2DHM

Our previously described weakly supervised approach to learn hypercolumn descriptors benefits from a global-level loss which only requires to know whether two images are covisible or not. In S2DNet, we propose a supervised alternative which aims at directly learning accurate correlation maps. As we will show in our experiments, this leads to significantly superior performance in keypoint matching accuracy (at the cost of more expensive data annotation). Moreover, in its pipeline S2DHM upsamples and concatenates intermediate feature maps before computing correlation maps. In comparison, S2DNet computes correlation maps at multiple levels before merging the results by addition. We will later show that the latter approach is much more memory and computationally efficient.

3.6 Experiments

In this section, we will conduct experiments on both S2DHM and S2DNet. We will begin by demonstrating how the sparse-to-dense paradigm can help improve long-term visual localization performance even with a simple global contrastive loss (see Section 3.6.1). Then, we will show that bringing strong supervision in the loop with S2DNet helps achieving even better performance, outperforming the state-of-the-art at the time of publication (see Section 3.6.2).

Dataset	Training sequences	Condition	Training images	Reference images	Query images
RobotCar Seasons (Maddern et al., 2017; Sattler et al., 2018)	12 Dec 2014	overcast	20,965	6,954	3,978
	05 Dec 2014	overcast-rain	20,965		
	16 Dec 2014	night	19,376		
	03 Feb 2015	night	20,257		
Extended CMU-Seasons (Badino et al., 2011; Sattler et al., 2018)	Slices 2-8	urban	9,612	7,159	75,335
	Slices 9-17	suburban	24,728		
	Slices 18-25	park	16,148		

Table 3.1: **Detailed statistics** regarding the training and testing sequences used for each dataset. Reference images are used to triangulate 3D keypoints offline using SuperPoint (Detone et al., 2018) detections and descriptors. Note that for RobotCar Seasons, only rear images are considered.

3.6.1 Experiments on S2DHM

To begin we conduct experiments on S2DHM for long-term visual localization. In Section 3.6.1.1, we detail how our evaluation datasets were setup. We also discuss the evaluation methods and baselines we used for comparison. In Section 3.6.1.2, we show how our hierarchical method can solve camera poses accurately under challenging conditions and outperforms existing methods in such categories. Lastly, in Section 3.6.1.3, we run an ablation study, which demonstrates the improvements brought by the combination of sparse-to-dense matching and hypercolumn descriptors.

3.6.1.1 Evaluation Setup

We begin our evaluation by presenting the two challenging outdoor datasets introduced by (Sattler et al., 2018) which we will be using throughout this section.

Datasets. Our evaluation set consists of two outdoor datasets captured from vehicles or using hand-held mobile phone cameras. Each of the provided datasets contains a set of reference images, along with their ground truth camera poses. We are also given sparse 3D reconstructions pre-computed using RootSIFT (Perdoch et al., 2009) features by Sattler et al. (Sattler et al., 2018). In practice, we do not use the provided sparse 3D reconstruction and re-triangulated our own point clouds using SuperPoint (Detone et al., 2018) detections. We perform the triangulation using COLMAP (Schönberger & Frahm, 2016; Schönberger et al., 2016) on the reference images of each dataset, similarly to (Sarlin et al., 2019).

The first dataset is the Extended CMU-Seasons dataset (Sattler et al., 2018), which contains about 40% more images than the original CMU-Seasons dataset (Badino et al., 2011). It consists of 7,159 reference images and 75,335 query images, captured using two front-facing cameras mounted on a car, in the area of Pittsburgh. The images were

	Method	*RobotCar Seasons*						*Extended CMU-Seasons*								
		Day-All			*Night-All*			*Urban*			*Suburban*			*Park*		
		Threshold Accuracy			Threshold Accuracy			Threshold Accuracy			Threshold Accuracy			Threshold Accuracy		
		0.25m 2°	0.5m 5°	5m 10°	0.25m 2°	0.5m 5°	5m 10°	0.25m 2°	0.5m 5°	5m 10°	0.25m 2°	0.5m 5°	5m 10°	0.25m 2°	0.5m 5°	5m 10°
Structure-based	CSL (Svärm et al., 2017)	45.3	73.5	90.1	0.6	2.6	7.2	71.2	74.6	78.7	57.8	61.7	67.5	34.5	37.0	42.2
	AS (Sattler et al., 2017a)	35.6	67.9	90.4	0.9	2.1	4.3	-	-	-	-	-	-	-	-	-
	SMC (Toft et al., 2018)	**50.3**	**79.3**	**95.2**	**7.1**	**22.4**	45.3	**88.8**	**93.6**	**96.3**	**78.0**	**83.8**	89.2	**63.6**	**70.3**	77.3
Retrieval-based	FAB-MAP (Cummins & Newman, 2008)	2.7	11.8	37.3	0.0	0.0	0.0	-	-	-	-	-	-	-	-	-
	NetVLAD (Arandjelovic et al., 2016)	6.4	26.3	90.9	0.3	2.3	15.9	12.2	31.5	89.8	3.7	13.9	74.7	2.6	10.4	55.9
	DenseVLAD (Torii et al., 2015)	7.6	31.2	91.2	1.0	4.4	22.7	14.7	36.3	83.9	5.3	18.7	73.9	5.2	19.1	62.0
	ToDayGAN (Anoosheh et al., 2019)	7.6	31.2	91.2	2.2	10.8	**50.5**	-	-	-	-	-	-	-	-	-
Hierarchical	NV+SP (Sarlin et al., 2019)	**53.0**	**79.3**	95.0	5.9	17.1	29.4	**89.5**	**94.2**	**97.9**	**76.5**	**82.7**	**92.7**	**57.4**	64.4	**80.4**
	NV-r + S-D + H (Ours)	45.7	**78.0**	**95.1**	**22.3**	**61.8**	**94.5**	65.7	82.7	91.0	66.5	82.6	**92.9**	54.3	**71.6**	**84.1**

Table 3.2: **Localization results**. We report localization recalls in percent, for three translation and orientation thresholds (*high*, *medium*, and *coarse*) as in (Sattler et al., 2018). We highlight the **best** performance in red and **second-best** performance in blue for each threshold. Note that NetVLAD, ToDayGAN, and NV+SP all use pre-trained NetVLAD weights from Pittsburgh30k (Arandjelovic et al., 2016), while we retrained ours on other RobotCar sequences. We also include SMC, which uses additional semantic data and assumptions. For Extended CMU-Seasons, some methods did not provide results for the benchmark.

captured over the course of a year and the reference images depict different seasonal conditions. The *park* scene is particularly difficult as it was captured in a rural environment and faces strong vegetation changes over the year.

The second dataset is the RobotCar Seasons dataset (Maddern et al., 2017), which contains 6,954 daytime images captured by a rear-facing camera mounted on a car driving in Oxford. The 3,978 query images were taken over the course of a year, including some in very challenging conditions such at nighttime (Sattler et al., 2018). Note that in these experiments we do not consider the additional reference images taken by the two side-facing cameras. We report details about the exact sequences used for training for each dataset in Table 3.1.

Baselines. We compare our approach both against structure-based and retrieval-based methods, that were state-of-the-art at the time of publication. Localization results for these methods were provided by the authors of the benchmark (Sattler et al., 2018).

For structure-based methods, we compare our approach to Active Search (AS) (Sattler et al., 2017a) and City-Scale Localization (CSL) (Svärm et al., 2017). Both methods are direct 2D-3D matching techniques optimized for matching efficiency and robustness respectively, and have shown to deliver great accuracy in daytime conditions at a high precision threshold (Sattler et al., 2018). We also display results for Semantic Match Consistency (SMC) (Toft et al., 2018), which leverages semantic maps to filter outliers in the matching stage, and makes additional assumptions regarding the camera height and gravity vector.

We also compare our approach to retrieval-based methods, such as NetVLAD (pre-trained on Pittsburgh30k (Arandjelovic et al., 2016) with a VGG-16 (Simonyan & Zisserman, 2014) backbone), and to DenseVLAD (Torii et al., 2015). For these methods, we simply approximate the query image camera pose by the pose of its retrieved top-ranked database image. Details about their configuration and implementation details can be found in the original benchmark (Sattler et al., 2018). Additionally for RobotCar Seasons, we report the results obtained by performing night-to-day image translation using a GAN architecture (ToDayGAN) (Anoosheh et al., 2019), prior to running DenseVLAD.

Lastly, we show the results obtained by Sarlin et al. (Sarlin et al., 2019), which is a hierachical approach using a pre-trained NetVLAD backbone followed by SuperPoint (Detone et al., 2018) feature detection and local descriptors for 2D-3D matching (NV+SP). This method also uses co-visibility clusters to merge 3D points from neighbouring database images.

Metrics. We evaluate our approach using the same localization metric as (Sattler et al., 2018). Three precision thresholds are defined, accounting for both positional and rotational error. We refer to these thresholds as *high* (0.25m and 2°), *medium* (0.5m and 5°) and *coarse* (5m and 10°) precision. For each threshold, we report the localization recall in percent.

3.6.1.2 Large-scale localization

Having established our evaluation process, we now report the performance of our approach.

Training. For the NetVLAD retrieval backbone, we use different weights for both datasets. For RobotCar Seasons (Maddern et al., 2017), we retrained NetVLAD on tuples extracted from other RobotCar sequences, featuring for daytime and nighttime images (see Table 3.1). Positive and negative tuples were assembled using the provided GPS and INS data. Note that these sequences do not overlap with the test set. For Extended CMU-Seasons (Badino et al., 2011), we built training samples using all the provided annotated training data from the *urban*, *suburban* and *park* slices. When training NetVLAD, we use hard-negative mining at every epoch, to obtain for each query the hardest subset of all possible negatives in the database.

Methods. As presented in Section 3.4, we run our hierarchical localization pipeline by first ranking each query with respect to the reference images. We use the normalized global image descriptors produced by NetVLAD (NV), and obtain the rankings using a simple dot product. To account for potential image retrieval errors, for every query we run

the exhaustive matching step on each of the top-N nearest neighbors. The final predicted pose is picked as the one having the highest number of inliers in the RANSAC loop of the PnP. For RobotCar Seasons, we use $N = 15$ and for Extended CMU-Seasons, we use $N = 10$ because of the large amount of images to evaluate.

Method	*Day-All*			*Night-All*		
	Threshold Accuracy			Threshold Accuracy		
	0.25m 2°	0.5m 5°	5m 10°	0.25m 2°	0.5m 5°	5m 10°
NV (pre-trained)	6.4	26.3	90.9	0.3	2.3	15.9
NV-r (re-trained)	4.1	17.8	86.9	2.4	11.4	84.6
NV-r + S-S + SP	**52.9**	**78.5**	93.8	10.9	32.7	87.4
NV-r + S-S + H	49.0	77.9	93.6	14.8	44.5	89.7
NV-r + S-D + SP	50.3	77.5	92.9	14.4	43.2	87.8
NV-r + S-D + H	45.7	78.0	**95.1**	**22.3**	**61.8**	**94.5**

Table 3.3: **Ablation Study** on the RobotCar Seasons dataset. We first show the improvements coming from using a retrained NetVLAD (NV) (Arandjelovic et al., 2016) backbone. Then, we report localization performance using standard sparse-to-sparse (S-S) matching using SuperPoint detections and two different descriptors: SuperPoint descriptors (S-S + SP) and Hypercolumn descriptors (S-S + H), as well as the results of our sparse-to-dense (S-D) matching using SuperPoint descriptors (S-D + SP) and Hypercolumn descriptors (S-D + H). We report localization recall in percent, for three translation and orientation thresholds.

	Dense Query Hypercolumn Descriptors	*Sparse Reference Hypercolumn Descriptors (offline)*	*Correspondence Maps (Exhaustive search)*	*Ratio Test (non-optimized)*	*PnP Solving*
Runtime (ms)	107.29	114.71	10.8	169.14	3.08

Table 3.4: **Runtime measurements.** We report the average runtimes for our sparse-to-dense matching approach on RobotCar Season, with 512×512 input images. Operations in italic are run for each of the top-ranked images.

Implementation details. We use a Pytorch implementation of NetVLAD to compute the global image descriptors as well as the intermediate VGG-16 features used to compute the hypercolumns. As in (Sarlin et al., 2019), we reduce the dimensionality of all produced descriptors to a size of 1024 using PCA, learned on the reference set. When retraining NetVLAD on RobotCar Seasons and Extended CMU-Seasons, images are rescaled to a maximum size of 512 pixels, while preserving image ratio. At inference time, we again

rescale images to a maximum size of 512 pixels for all datasets, both to compute the global image descriptors and to extract intermediate dense features. The offline point cloud triangulation and the online 2D-3D correspondences are done using the original images resolutions.

We use different ratio test values for each dataset. For RobotCar Seasons we use a factor of $f = 0.006$. For Extended CMU-Seasons we use a value of 0.12, as we found much more ambiguous matches and using selective thresholds were leading to a high number of rejections. As in (Sarlin et al., 2019), for both datasets, the RANSAC (Fischler & Bolles, 1981) loop stops when a pose has a minimum number of inliers of 15.

Performance. We run our experiments on a PC equipped with an Intel(R) Xeon(R) E5-2630 CPU (2.20GHz) CPU with 128GB of RAM and an NVIDIA GeForce GTX 1080Ti GPU. We pre-compute compressed global image descriptors for a faster image retrieval at inference time. Our main bottleneck in terms of computation times in our current implementation lies in the VGG-16 inference. As shown in (Sarlin et al., 2019), this part can be sped up using a teacher network with little loss in accuracy. Our ratio test method could also be replaced by a faster, more traditional non-maxima suppression scheme computed on GPU. The computation of the correspondence map is done on GPU through a convolution operation. We report the average measured runtimes in Table 3.4.

Results. We report the localization results in Table 3.2. Our method outperforms all baselines in very challenging scenarios such as nighttime for RobotCar Seasons. We also show significant improvements for the *park* scene of Extended CMU-Seasons, which is arguably the most difficult with strong changes in vegetation, at *medium* and *coarse* precision thresholds. For other categories, the performance is usually on par with state-of-the-art structure-based or hierarchical methods such as SMC (Toft et al., 2018) or NV+SP (Sarlin et al., 2019) respectively. On easier categories, such as *day-all* for RobotCar Seasons or *urban* for CMU, our approach is not as accurate as other feature-point based approaches, especially at a finer threshold. It is therefore more adapted to complex correspondence problems. On less challenging cases, the standard approach which relies on a detector with sub-pixel accuracy for the query image can still be more accurate.

3.6.1.3 Ablation Study

Having presented the results of our full pipeline, we now evaluate the impact of each element of our pipeline in the localization step. We run this ablation study on RobotCar Season (Maddern et al., 2017) and report our results in Table 3.3.

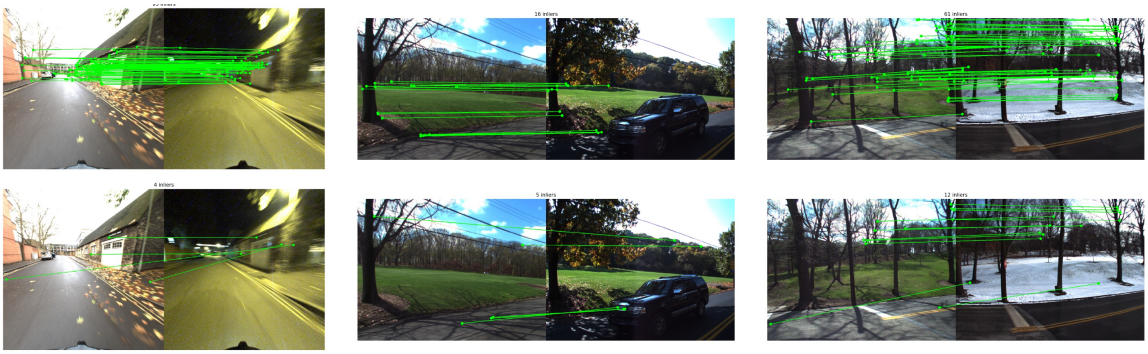


Figure 3.8: **Examples of inlier correspondences obtained using RANSAC+PnP.** Top-row shows correspondences obtained with S2DHM (sparse-to-dense matching), bottom row shows correspondences obtained with SuperPoint detection and descriptors (sparse-to-sparse matching).

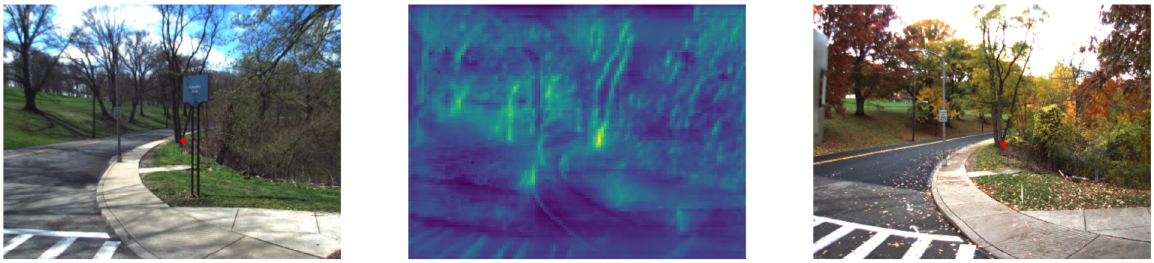


Figure 3.9: **Correlation map example.** Left image shows a Superpoint detection in the reference image. The corresponding sparse hypercolumn descriptor is used to compute the correlation map (middle) and retrieve the 2D correspondent in the query image (right).

NetVLAD backbones. We first discuss the impact of having a retrained image-retrieval backbone. As shown in Table 3.3, the pre-trained Pittsburgh30k (Arandjelovic et al., 2016) weights (NV) provides a good coarse pose estimation in daytime, but still very mild results at nighttime. We can already see that this will be a very limiting factor when performing 2D-3D matching, as the selected point cloud subsets will not be overlapping with the query image. When retraining NetVLAD (NV-r) on daytime and nighttime sequences from RobotCar, this gives a significant boost in performance in both conditions, especially at a coarse precision level (*N.B.:* the training and test set are distinct and do not overlap).

This strong performance is also tightly linked with the database spatial sampling: A dataset sampled much more sparsely would yield poor results at a coarse level even at daytime. We also tried retraining NetVLAD with a ResNet-50 (He et al., 2016) backbone, and / or a GeM (Radenovic et al., 2018b) layer activation, but this always yielded slightly poorer retrieval results than a VGG-16 (Simonyan & Zisserman, 2014) network with a

VLAD activation layer.

Sparse-to-sparse matching. We evaluate adding a subsequent camera pose estimation using 2D-3D matches coming from standard sparse-to-sparse (S-S) matching using SuperPoint (Detone et al., 2018) detections and two different descriptors: SuperPoint descriptors (S-S + SP) and Hypercolumn descriptors (S-S + H). Both approaches (S-S + SP) and (S-S + H) allow to significantly improve the daytime results. For nighttime results, even if the performance improved, they remain limited compared to daytime. We argue that this discrepancy between daytime and nighttime results comes from the difficulty to repeatably detect and match sparse keypoints extracted from two images captured under very different conditions. This motivates our novel sparse-to-dense matching approach. Finally, one can see that the aggregation of dense features into hypercolumns at different levels provides improvements. This shows the advantage of using hypercolumns for description rather than the Superpoint descriptors. This advantage is likely due to the large receptive fields of the hypercolumns computed by VGG, and the way they are learned to be condition-invariant.

Sparse-to-dense matching. We finally evaluate S2DHM by replacing the standard sparse-to-sparse matching module with our novel sparse-to-dense matching for both Superpoint descriptors (S-D + SP) and Hypercolumn descriptors (S-D + H). As shown in Table 3.3, our novel approach is a way to partially remove the nighttime detection bottleneck: Compared to sparse-to-sparse Hypercolumn matching (NV-r + S-S + H), our sparse-to-dense Hypercolumn matching (NV-R + S-D + H) increases the recall by 7.5% and 17.3% for the *high* and *medium* thresholds respectively at nighttime.

3.6.1.4 Qualitative results

We report qualitative results of PnP +RANSAC inlier correspondences in Fig. 3.8 and a correlation map in Fig. 3.9. More visualizations are available in the Appendix.

3.6.1.5 Discussion

We have introduced a novel hierarchical localization method based on the paradigm of sparse-to-dense matching and showed that by breaking the paradigm of detecting keypoints in both images to match, we can significantly improve the number of correct matches. While this approach was demonstrated in the context of visual localization, it is very likely to be generalizable to other computer vision tasks that resort to keypoint matching.

3.6.2 Experiments on S2DNet

In this section, we evaluate S2DNet on several challenging benchmarks. We first evaluate our approach on a popular keypoint matching benchmark, which displays changes in both viewpoint and illumination. We then evaluate the performance of S2DNet on long-term visual localization tasks, which display even more severe visual changes.

3.6.2.1 Training data

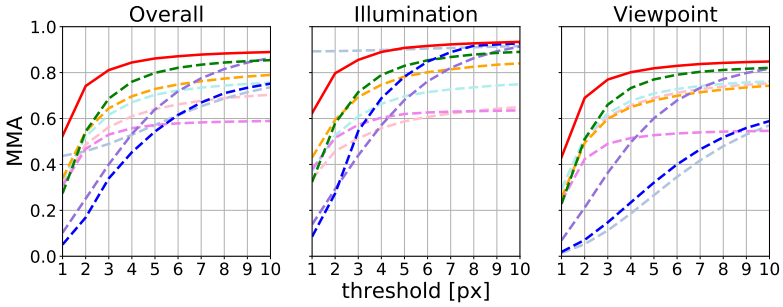
We use the same training data as D2-Net (Dusmanu et al., 2019) to train S2DNet, which comes from the MegaDepth dataset (Li & Snavely, 2018). This dataset consists of 196 outdoor scenes and $1,070,568$ images, for which *SfM* was run with COLMAP (Schönberger & Frahm, 2016; Schönberger et al., 2016) to generate a SIFT-based sparse 3D reconstruction. A depth-check is run using the provided depth maps to remove occluded pixels. As D2-Net, we remove scenes which overlap with the PhotoTourism (Thomee et al., 2016; Trulls et al., 2019) test set. Compared to D2-Net and to provide strong scale changes, we train S2DNet on image pairs with an arbitrary overlap. At each training iteration, we extract random crops of size 512×512, and randomly sample a maximum of 128 pixel correspondences. We train S2DNet for 30 epochs using Adam (Kingma & Ba, 2015). We use an initial learning rate of 10^{-3} and apply a multiplicative decaying factor of $e^{-0.1}$ at every epoch.

3.6.2.2 Image matching

We first evaluate our method on the popular image matching benchmark HPatches (Balntas et al., 2017). We use the same 108 sequences of images as D2-Net (Dusmanu et al., 2019), each sequence consisting of 6 images. These images either display changes in illumination (for 52 sequences) or changes in viewpoint (for 56 sequences). We consider the first frame of each sequence to be the reference image to be matched against every other, resulting in 540 pairs of images to match.

Evaluation protocol. We apply the SuperPoint (Detone et al., 2018) keypoint detector on the first image of each sequence. For each subsequent pair of images, we perform sparse-to-dense matching using S2DNet (see section 3.5.2.4). Additionally, we filter out correspondences which do not pass the cyclic check of matching back on their source pixel, which is equivalent to performing a mutual nearest-neighbor verification as it is done with D2-Net (Dusmanu et al., 2019) and R2D2 (Revaud et al., 2019).

We compute the number of matches which fall under multiple reprojection error thresholds using the ground-truth homographies provided by the dataset, and report the Mean Matching Accuracy (or MMA) in Figure 3.10.



Method	#Features	#Matches
Hes.det.+RootSIFT (Perdoch et al., 2009)	6.7K	2.8K
HAN+HN++ (Mishchuk et al., 2017)	3.9K	2.0K
LF-Net (Ono et al., 2018)	0.5K	0.2K
SuperPoint (Detone et al., 2018)	1.7K	0.9K
DELF (Noh et al., 2017)	4.6K	1.9K
S2DHM (Germain et al., 2019)	0.9K	0.4K
D2-Net (Dusmanu et al., 2019)	8.3K	2.8K
R2D2 (Revaud et al., 2019)	5.0K	1.8K
S2DNet (ours)	2.0K	0.8K

Figure 3.10: **HPatches Mean Matching Accuracy (MMA) comparison.** We report in this table the best results for S2DNet, obtained when combined with SuperPoint detections. S2DNet outperforms all other baselines, especially at thresholds of one or two pixels. This study highlights the power of working in a sparse-to-dense setting, where every pixel in the target image becomes a candidate keypoint. These results should however be taken with a grain of salt as different methods use different threshold levels and have various amounts of keypoint matches. Thus predicting a single but perfect correspondence would yield a score of 1.0.

We compare S2DNet to multiple sparse-to-sparse matching baselines. We report the performance of RootSIFT (Arandjelović & Zisserman, 2012; Perdoch et al., 2009) with a Hessian Affine detector (Mikolajczyk & Schmid, 2004; Perdoch et al., 2009) (Hes.det. + RootSIFT), HardNet++ (Mishchuk et al., 2017) coupled with HesAffNet regions (Mishkin et al., 2018) (HAN + HN++), DELF (Noh et al., 2017), LF-Net (Ono et al., 2018), SuperPoint (Detone et al., 2018), D2-Net (Dusmanu et al., 2019) and R2D2 (Revaud et al., 2019). We also include results from S2DHM.

On the validity of HPatches. It is important to note that the HPatches benchmark does not factor in the amount of filtering applied on keypoint correspondences or the various detections when computing the mean matching accuracy. Thus predicting a single but perfect correspondence for all images would yield a score of 1.0, while predicting far more correspondences with little noise would yield a much lower score. Thus, this benchmark tends to favor methods that apply a fairly strict correspondence filtering mechanism, and conclusions drawn from this study should be taken with a grain of salt.

Results. We find that the best results were achieved when combining SuperPoint (Detone et al., 2018) with a threshold of $\tau = 0.20$ (see Equation 3.5), which are the results reported in Figure 3.10. We experimentally found that above this threshold, some sequences obtain very few to no correspondence at all, which biases the results. We show that overall our method outperforms every baselines at any reprojection threshold. The gain in performance is particularly noticeable at thresholds of 1 and 2 pixels, indicating the correspondences we predict tend to be much more accurate. DELF (Noh et al., 2017) achieves

competitive results under changes in illumination, which can be explained by the fact that keypoints are sampled on a fixed grid and that the images undergo no changes in viewpoint. On the other hand, it performs poorly under viewpoint changes.

Keypoint detector influence. We run an ablation study to evaluate the impact of different feature detectors, confidence thresholds as well as using a sparse-to-sparse approach, and report the results in Table 3.5 (left). We find that S2DNet tends to work best when combined with SuperPoint (Detone et al., 2018). We also experimentally find $\tau = 0.2$ to be a good compromise of correspondence rejection while also maintaining a high number of matches.

Sparse-to-sparse vs. sparse-to-dense. We find that using S2DNet in a sparse-to-sparse setting (*i.e.* applying a detector on the image undergoing illumination or viewpoint changes) damages the results (see Table 3.5, left). This phenomenon translates the errors made by keypoint detectors, and motivates the sparse-to-dense setting. S2DNet efficiently leverages this paradigm and can find corresponding keypoints that would not have been detected otherwise. Conversely, we study the impact of using sparse-to-sparse learning-based methods D2-Net (Dusmanu et al., 2019) and R2D2 (Revaud et al., 2019) in a sparse-to-dense setting (see Table 3.5, right). In this setting, we define S2DNet sparse descriptors as the concatenated multi-scale feature vectors at the corresponding pixel locations. We find that using the sparse-to-dense paradigm systematically improves their performance under illumination changes, where images are aligned. This suggests that their descriptor maps are robust to illumination perturbations. On the other hand, performance is damaged for both methods under viewpoint changes, suggesting that their descriptor maps are not highly localized and discriminative. Concerning S2DHM, which was trained in a weakly supervised manner, running it in a sparse-to-sparse setting improves the accuracy. This highlights the importance of our main contribution, *i.e.* casting the sparse-to-dense matching problem as a supervised classification task.

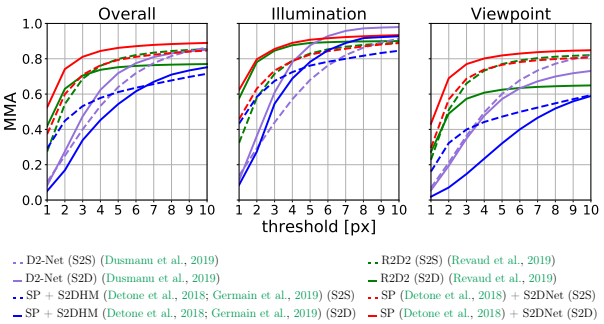
3.6.2.3 Long-Term Visual Localization

We showed that S2DNet provides correspondences which are overall more accurate than other baselines. We will now study its impact for the task of visual localization under challenging conditions. We report visual localization results under day-night changes and complex indoor scenes.

Datasets. We evaluate our approach on two challenging outdoor localization datasets which feature day-to-night changes, and one indoor dataset. The first dataset is Aachen Day-Night (Sattler et al., 2018, 2012). It features 4,328 daytime reference images taken

Detector	Matching	τ	MMA@1	MMA@2	MMA@3	MMA@10
Harris (Harris & Stephens, 1988)	S2D	0.20	**0.511**	**0.733**	**0.805**	**0.888**
	S2D	0.0	0.441	0.626	0.690	0.787
	S2S	-	0.278	0.464	0.565	0.763
SURF (Bay et al., 2006)	S2D	0.20	**0.511**	**0.742**	**0.823**	**0.902**
	S2D	0.0	0.436	0.639	0.718	0.828
	S2S	-	0.302	0.506	0.619	0.829
SIFT (Lowe, 2004)	S2D	0.20	**0.487**	**0.700**	**0.771**	**0.851**
	S2D	0.0	0.441	0.626	0.690	0.787
	S2D	-	0.386	0.559	0.642	0.818
SuperPoint (Detone et al., 2018)	S2D	0.20	**0.563**	**0.747**	**0.815**	**0.895**
	S2D	0.0	0.469	0.623	0.686	0.788
	S2S	-	0.373	0.599	0.709	0.847
D2-Net (Dusmanu et al., 2019)	S2D	0.20	**0.467**	**0.716**	**0.805**	**0.911**
	S2D	0.0	0.330	0.522	0.604	0.764
	S2S	-	0.118	0.285	0.425	0.777
R2D2 (Revaud et al., 2019)	S2D	0.20	**0.478**	**0.715**	**0.799**	**0.901**
	S2D	0.0	0.341	0.522	0.598	0.746
	S2S	-	0.316	0.546	0.652	0.819

(a) **S2S vs. S2D - S2DNet descriptors**



(b) **S2S vs. S2D - Other descriptors**

Table 3.5: **Ablation study on HPatches.** In **(a)**, we evaluate the performance of several detectors in both a sparse-to-dense (S2S) and sparse-to-sparse (S2S) setting using S2DNet descriptors. We find that S2DNet works best in the S2D setting, coupled with SuperPoint (SP) (Detone et al., 2018) detections, and a confidence threshold of $\tau = 0.20$. In **(b)**, we study the impact of using sparse-to-sparse learning-based methods in a sparse-to-dense setting. Results lead to the conclusion that D2-Net (Dusmanu et al., 2019) and R2D2 (Revaud et al., 2019) descriptor maps are robust to illumination changes but not highly discriminative locally.

with a handheld smartphone, for which ground truth camera poses are provided [2]. The dataset also provides a 3D reconstruction of the scene (Sattler et al., 2018), built using SIFT (Lowe, 2004) features and *SfM*. The evaluation is done on 824 daytime and 98 nighttime images taken in the same environment. The second dataset is RobotCar Seasons (Maddern et al., 2017). It features 6,954 daytime reference images taken with a rear-facing camera mounted on a car driving through Oxford. Similarly, ground truth camera poses and a sparse 3D model of the world is provided (Sattler et al., 2018) and we localize 3,978 images captured throughout a year. These images do not only exhibit nighttime conditions, but also cross-seasonal evolutions such as snow or rain. Lastly, we evaluate our pipeline on the challenging InLoc (Taira et al., 2018; Wijmans & Furukawa, 2016) dataset. This indoor dataset is difficult because of its large scale, illumination and long-term changes as well as the presence of repetitive patterns such as corridors (see Figure 3.11). It contains 9,972 database and 356 high-resolution query images, as well as dense depth maps which can be used to perform dense pose verification. We report for each datasets the pose recall at three position and orientation thresholds for daytime and nighttime query images, as per (Sattler et al., 2018).

Indoor Localization. The InLoc (Taira et al., 2018) localization benchmark comes

[2] After the time of publication an updated version of Aachen Day-Night was released with improved camera pose and 3D model estimates. We report results from the former version of the dataset i.e. as it was at the time of publication.

Method	*InLoc* Threshold Accuracy		
	0.25m / 2°	0.5m / 5°	5m / 10°
Direct PE - Aff. RootSIFT (Mikolajczyk & Schmid, 2004; Perdoch et al., 2009)	18.5	26.4	30.4
Direct PE - D2-Net (Dusmanu et al., 2019)	27.7	40.4	**48.6**
Direct PE - S2DNet (ours)	**29.3**	**40.9**	48.5
Sparse PE - Aff. RootSIFT (Mikolajczyk & Schmid, 2004; Perdoch et al., 2009)	21.3	32.2	44.1
Sparse PE - D2-Net (Dusmanu et al., 2019)	35.0	48.6	62.6
Sparse PE - S2DNet (ours)	**35.9**	**49.0**	**63.1**
Sparse PE + Dense PV - Aff. RootSIFT (Mikolajczyk & Schmid, 2004; Perdoch et al., 2009)	29.5	42.6	54.5
Sparse PE + Dense PV - D2-Net (Dusmanu et al., 2019)	38.0	**56.5**	65.4
Sparse PE + Dense PV - S2DNet (ours)	**39.4**	53.5	**67.2**
Dense PE + Dense PV - InLoc (Taira et al., 2018)	38.9	56.5	69.9

Method	*Aachen Day-Night* Threshold Accuracy		
	0.25m 2°	0.5m 5°	5m 10°
RootSIFT (Perdoch et al., 2009)	3.7	52.0	65.3
HAN+HN (Mishkin et al., 2018)	37.8	54.1	75.5
SuperPoint (Detone et al., 2018)	42.8	57.1	75.5
DELF (Noh et al., 2017)	39.8	61.2	85.7
D2-Net (Dusmanu et al., 2019)	44.9	66.3	**88.8**
R2D2 (Revaud et al., 2019)*	**45.9**	66.3	**88.8**
S2DNet (ours)	**45.9**	**68.4**	**88.8**

Table 3.6: **InLoc (Taira et al., 2018) (top) and Local Features Benchmark (Sattler et al., 2018) (bottom) results**. We report localization recalls in percent, for three translation and orientation thresholds. On InLoc, S2DNet outperforms both baselines at the finest threshold for the sparse categories. We also include Dense PE baseline results for reference. R2D2 authors did not provide results on this benchmark. On the local features benchmark (a pre-defined localization pipeline), S2DNet achieves state-of-the-art results at the medium precision threshold. Due to the relatively small number of query images however, recent methods like D2-Net and R2D2 are saturating around the same performance. **Note that R2D2 was trained on Aachen database images, and that there now exists an improved version of the Aachen Day-Night ground truth data with additional query images, results are reported as is from the former version.*

with a pre-defined code base and several pipelines for localization. The first one is called Direct Pose Estimation (Direct PE) and performs hierarchical localization using the set of top-ranked database images obtained using image retrieval, followed by P3P-LO-RANSAC (Fischler & Bolles, 1981; Lebeda et al., 2012). The second variant applies an intermediate spatial verification step (Philbin et al., 2007) to reject outliers, referred to as (Sparse PE). On top of this second variant, Dense Pose Verification (Dense PV) can be applied to re-rank pose candidates by using densely extracted RootSIFT (Perdoch et al., 2009) features. In each variant, we use S2DNet to generate 2D-2D correspondences between queries and database images, which are then converted to 2D-3D correspondences using the provided dense depth maps. We use a SuperPoint (Detone et al., 2018) detector

	Method	*RobotCar Seasons*						*Aachen Day-Night*					
		Day-All			*Night-All*			*Day*			*Night*		
		Threshold Accuracy			Threshold Accuracy			Threshold Accuracy			Threshold Accuracy		
		0.25m 2°	0.5m 5°	5m 10°	0.25m 2°	0.5m 5°	5m 10°	0.25m 2°	0.5m 5°	5m 10°	0.25m 2°	0.5m 5°	5m 10°
Structure -based	CSL (Svärm et al., 2017)	45.3	73.5	90.1	0.6	2.6	7.2	52.3	80.0	94.3	24.5	33.7	49.0
	AS (Sattler et al., 2017a)	35.6	67.9	90.4	0.9	2.1	4.3	57.3	83.7	96.6	19.4	30.6	43.9
	SMC (Toft et al., 2018) *	50.3	79.3	95.2	7.1	22.4	45.3	-	-	-	-	-	-
Retrieval -based	FAB-MAP (Cummins & Newman, 2008)	2.7	11.8	37.3	0.0	0.0	0.0	0.0	0.0	4.6	0.0	0.0	0.0
	NetVLAD (Arandjelovic et al., 2016)	6.4	26.3	90.9	0.3	2.3	15.9	0.0	0.2	18.9	0.0	2.0	12.2
	DenseVLAD (Torii et al., 2015)	7.6	31.2	91.2	1.0	4.4	22.7	0.0	0.1	22.8	0.0	2.0	14.3
Hierar -chical	HF-Net (Sarlin et al., 2019)	53.0	79.3	95.0	5.9	17.1	29.4	79.9	88.0	93.4	40.8	56.1	74.5
	S2DHM (Germain et al., 2019) *	45.7	78.0	95.1	22.3	61.8	94.5	56.3	72.9	90.9	30.6	56.1	78.6
	D2-Net (Dusmanu et al., 2019)	**54.5**	80.0	95.3	**20.4**	40.1	55.0	**84.8**	**92.6**	**97.5**	43.9	66.3	85.7
	S2DNet (ours)	53.9	**80.6**	**95.8**	14.5	**40.2**	**69.7**	84.3	90.9	95.9	**46.9**	**69.4**	**86.7**

Table 3.7: **Localization results**. We report localization recalls in percent, for three translation and orientation thresholds (*high*, *medium*, and *coarse*) as in (Sattler et al., 2018). We put in bold the **best** and underline the second-best performances for each threshold. S2DNet outperforms every baseline in nighttime conditions, except at the finest threshold of RobotCar Seasons. This can be explained by the extreme visual changes and blurriness that these images undergo. At daytime, S2DNet performance is on par with D2-Net (Dusmanu et al., 2019). **Note that S2DHM was trained directly on RobotCar sequences, which explains the high nighttime performance. SMC (Toft et al., 2018) also uses additional semantic data and assumptions. R2D2 (Revaud et al., 2019) authors did not provide localization results on these benchmarks. There now exists an improved version of the Aachen Day-Night ground truth data with additional query images, results are reported as is from the former version.*

and mutual nearest-neighbour filtering.

InLoc localization results are reported in Table 3.6. We compare our approach to the original InLoc baseline which uses affine covariant (Mikolajczyk & Schmid, 2004) detections and RootSIFT (Perdoch et al., 2009) descriptors, as well as results provided by D2-Net (Dusmanu et al., 2019). We find that S2DNet outperforms both sparse baselines at the finest threshold, and is on par with other methods at the medium and coarse thresholds. In the sparse setting, best results are achieved when combined with geometrical and dense pose verification (Sparse PE + Dense PV). In addition we include localization results that were computed by the benchmark authors using dense-to-dense feature matching (Dense PE). Due to the nature of our pipeline and the very high memory and computational consumption of this variant, we choose to limit our study to sparse correspondence methods. It is interesting to note however that S2DNet outperforms the original (Dense PE + Dense PV) InLoc baseline at the finest precision threshold, using a much lighter computation.

Method	Network Backbone	Descriptor Size	Forward pass on $\mathtt{I}_q$	Detection step on $\mathtt{I}_q$	Matching per keypoint	Total online computational time		
			t_A	t_B	t_C	$t_A + t_B + N \times K \times t_C$		
						$N = 1$ $K = 1000$	$N = 5$ $K = 1000$	$N = 15$ $K = 1000$
D2-Net (Dusmanu et al., 2019)	VGG-16	512	17.8ms	5.474s	0.4μs	5.492s	5.494s	5.498s
R2D2 (Revaud et al., 2019)	L2-Net	128	19.1ms	479.6ms	0.2μs	0.499s	**0.499s**	**0.501s**
S2DHM (Germain et al., 2019)	VGG-16	2048	326ms	-	0.33ms	0.656s	1.976s	5.276s
S2DNet	VGG-16 + adap.	3×128	28.2ms	-	0.31ms	**0.338s**	1.578s	4.678s

Table 3.8: **Computational Time Study for visual localization.** We compare the time performance of our method against other learning-based approaches in a visual localization scenario. For a given query image $\mathtt{I}_q$ and N reference images of size 1200×1600 with K detections each, we report the average measured time to perform image matching against each of them. In this standard setting, keypoint locations and descriptors have already been extracted offline from the reference images.

Day-Night Localization. We report day-night localization results with S2DNet using two localization protocols. Localization results reported in Table 3.6 show that S2DNet achieves state-of-the-art results, outperforming all other methods at the medium precision threshold. It is important to note that R2D2 was finetuned on Aachen database images.

We then report in Table 3.7 localization results using a hierarchical approach, similar to S2DHM and (Sarlin et al., 2019; Sattler et al., 2018). Contrary to Table 3.6, these results do not allow to compare the keypoint matching approaches alone since localization pipelines are different. Even the comparison with D2Net is difficult to interpret since their full localization pipeline was not released. Still, S2DNet achieves state-of-the-art results in Aachen nighttime images, and outperforms all baselines that were not trained on RobotCar nighttime images at medium and coarse precision thresholds. At daytime, where detecting repeatable and accurate keypoints is easier, S2DNet is on par with other learning-based methods. At the finest nighttime RobotCar threshold it is likely that S2DNet features struggle to compute accurate correspondences, which can be explained by the extreme visual changes these images undergo (see Figure 3.11). Overall, this study shows that S2DNet achieves better performance in particularly challenging conditions such as nighttime, compared to other sparse-to-sparse alternatives.

3.6.2.4 Discussion

Runtime performance. To compare S2DNet against state-of-the-art approaches, we time its performance for the scenario of visual localization. We run our experiments on a machine equipped with an Intel(R) Xeon(R) E5-2630 CPU at 2.20GHz, and an NVIDIA GeForce GTX 1080Ti GPU. We report the results in Table 3.8. In a localization setting,

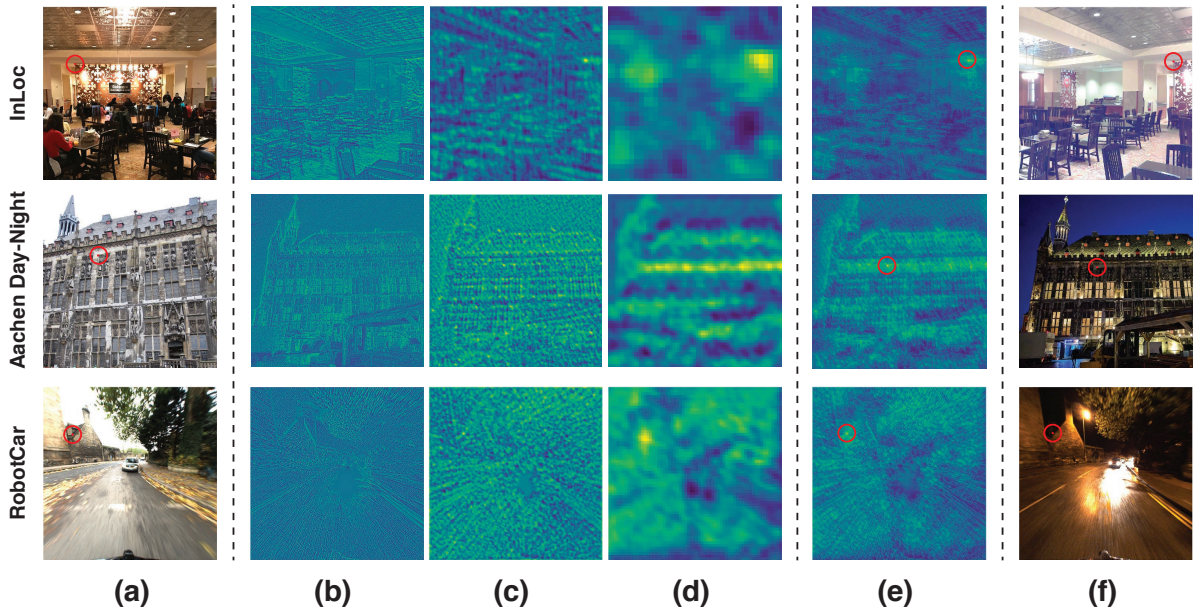


Figure 3.11: **Correspondence maps examples.** From left to right: Reference image with a keypoint detection (a), intermediate correspondence maps predicted by S2DNet (b, c, d), aggregated pre-softmax correspondence map (e) and retrieved correspondent in the query image (f). From top to bottom: images from InLoc (Taira et al., 2018), Aachen Day-Night (Sattler et al., 2012) and RobotCar Seasons (Maddern et al., 2017).

we consider the keypoint detection and description step to be pre-computed offline for reference images. Thus for an incoming query image, only sparse-to-sparse methods need to perform the keypoint detection and descriptor extraction step. We find this very step to be the bottleneck of learning-based methods like D2-Net (Dusmanu et al., 2019) or R2D2 (Revaud et al., 2019). Indeed, these methods are slowed down by the non-maxima suppression operations, which are in addition run on images of multiple scales. For S2DHM and S2DNet, no keypoint detection is performed on the incoming query image and most of the computation lies in the keypoint matching step. As expected however, the matching step is more costly for these sparse-to-dense methods. Still, for 1000 detections and 1 retrieved image, S2DNet is the fastest method while for 15 retrieved image, it is on par with D2-Net.

Current limitations of the sparse-to-dense paradigm. One limitation of our current sparse-to-dense matching formulation appears for the task of multiview 3D reconstruction. Indeed, the standard approach to obtain features tracks consists in (i) detecting and describing keypoints in each image, (ii) matching pairs of images using the previously extracted keypoints descriptors and (iii) creating tracks from these matches. In our

S2D matching paradigm, every pixel becomes a detection candidate which is not compatible with the standard 3D reconstruction pipeline previously described. This limitation opens novel directions of research for rethinking the standard tracks creation pipeline and enabling the use of S2D matching in 3D reconstruction frameworks.

Compatibility with learning-based matchers. Learning-based matching methods like NG-RANSAC (Brachmann & Rother, 2019b), OANet (Zhang et al., 2019) or SuperGlue (Sarlin et al., 2020) process putative correspondences to return inlier confidence scores. These methods could easily work as a post-processing step of S2DNet, to further improve matching results.

3.7 Conclusion

In this chapter we introduced the sparse-to-dense matching paradigm, which breaks the popular symmetry in keypoint detection to regress dense correlation maps and treat the keypoint matching as a pure feature learning task.

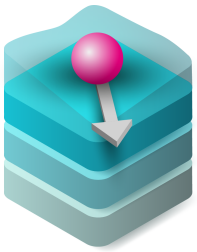
In S2DHM, we derived a weakly supervised approach that shows significant improvement over sparse-to-sparse approaches for the task of structure-based long-term visual localization. Despite being trained using an image-retrieval loss applied on global image descriptors, we find that intermediate hypercolumns are powerful descriptors for image matching. Our experiments show that using such hypercolumns in a sparse-to-dense matching framework provides much better camera pose estimates when relocalizing under challenging conditions compared to sparse-to-sparse alternative.

In S2DNet, we focused on the problem of generating highly accurate correlation maps in an efficient way. By casting the keypoint matching problem as a classification task our model is able to regress peaky maps and significantly improve the keypoint correspondences accuracy compared to S2DHM. In contrast to other sparse-to-sparse methods we showed that this novel pipeline achieves superior performance in terms of accuracy, which helps improve subsequent long-term visual localization tasks. Under visually challenging conditions, S2DNet reaches state-of-the-art performance for image matching and localization, and advocates for the development of sparse-to-dense methods.

So far we have only studied the sparse-to-dense matching problem as a mean of obtaining explicit 2D-to-2D or 2D-to-3D correspondences. In both S2DHM and S2DNet, we apply an argmax operator to correlation maps to identify the keypoint correspondent. In doing so we get rid of a significant amount of information, and ignore all values in both the vicinity of the argmax and at other possible modes. In the next chapter, we will thus focus on preserving this dense information from end-to-end specifically for the task of camera pose estimation by introducing a novel reprojection error.

Chapter 4

Merging Feature Learning and Camera Pose Estimation



4.1 Introduction

In this chapter we will leverage the sparse-to-dense matching paradigm to derive a purely learning-based approach to the problem of absolute camera pose estimation, which was presented in (Germain et al., 2021a).

Given a pre-acquired 3D model of the world, we aim at estimating the most accurate camera pose of an unseen query image. In practice, as already explained in Chapter **??** this problem is often addressed by sequentially solving two distinct subproblems: First, a feature matching problem that seeks to establish putative 2D-3D correspondences between the 3D point cloud and the image to be localized (a.k.a the matching stage), and then a Perspective-n-Point (PnP) problem that uses these correspondences as inputs to minimize a sum of so-called reprojection errors w.r.t. the camera pose (a.k.a the PnP stage).

Problem statement. Let us recall that the Reprojection Error (RE) presented in Eq. (2.3) in Section 2.3.2.3 is a function of a 2D-3D correspondence and the camera pose. It consists in reprojecting the 3D point, using the camera pose, into the query image plane, computing the euclidean distance between this reprojection and its putative 2D correspondent, and applying a robust loss function, such as Geman-McClure or Tukey's biweight (Barron, 2019; Zach & Bourmaud, 2017). The robust loss allows to reduce the influence of erroneous 2D-3D correspondences (*outliers*).

We argue that this strong decoupling of the matching stage from the PnP stage limits both the accuracy and the robustness of the camera pose estimate. Generating putative 2D-3D correspondences leads to an important loss of information since the 3D model and the query image are summarized into a set of 2D-3D coordinates. This loss of information needs to be compensated as far as possible within RE through the choice of a robust loss and the tuning of its hyperparameters, which usually depend on both the visual content and the amount of outliers generated by the matching stage. Moreover, outlier correspondences convey erroneous data to the pose estimator (see Fig. 4.1).

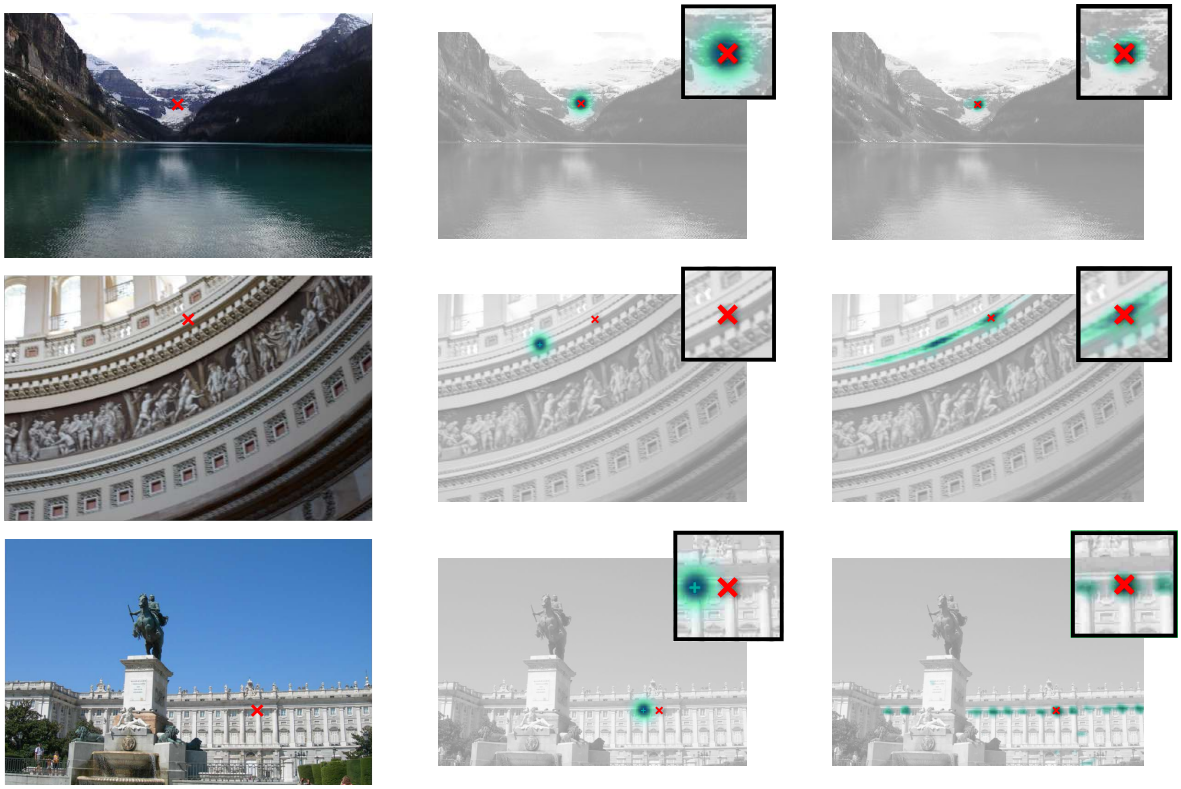
Contributions. In this chapter, we make the following contributions:

(i) We propose the *Neural Reprojection Error* (NRE) as a substitute for RE. NRE does not require a 2D-3D correspondence as input but relies on a *dense loss map*. A *dense loss map* contains much more information than a simple 2D-3D correspondence and conveys rich and expressive data to the pose estimator. As a result, the need for choosing a robust loss and its hyperparameters is also eliminated. Computing a *dense loss map* essentially involves cross-correlations between descriptors that are extracted using a neural network, hence the name *Neural Reprojection Error*.

(ii) Our derivation of NRE makes it differentiable not only w.r.t. to the camera pose but also w.r.t. the descriptors. Thus, providing ground-truth camera poses and minimizing NRE w.r.t. the descriptors yields a well-posed feature learning problem tailored for pose estimation. NRE merges the feature learning problem and the camera pose estimation problem in a new way and allows to rethink the end-to-end direct feature metric pose refinement methods.

(iii) To estimate the camera pose efficiently, we propose to minimize a sum of NRE terms in a coarse-to-fine manner. As a result, we never compute or store any high-resolution dense loss map. We also describe how to perform the optimization using an M-estimator sample consensus approach followed by a graduated non-convexity procedure. We experimentally demonstrate that our novel NRE-based pose estimator is a good substitute for RE-based pose estimators as it significantly improves both the robustness and the accuracy of the camera pose estimate while being computationally and memory highly efficient.

The outline of this chapter is as follows: In Section 4.2 we introduce some notations and describe our method in Section 4.3. In Section 4.6 provide a detailed discussion to highlight the differences between NRE and existing approaches. We finally present our evaluation results in Section 4.7.



(a) (b) (c)

Figure 4.1: **Neural Reprojection Error (NRE) as a substitute for Reprojection Error (RE): (a)** Given a 3D point $\mathbf{u}$, a query image $\mathtt{I}$ and its ground truth camera pose, $\mathbf{u}$ can be reprojected into the image plane of $\mathtt{I}$ to obtain a 2D point $\times$. **(b)** RE takes as input a camera pose and a putative 2D-3D correspondence between $\mathbf{u}$ and a 2D location $+$ in $\mathtt{I}$, reprojects $\mathbf{u}$ to obtain a 2D point $\mathbf{q}$, computes the euclidean distance between $+$ and $\mathbf{q}$ and finally applies a robust loss function (shown in turquoise as a function of $\mathbf{q}$). In ambiguous (middle) or multimodal (bottom) cases, generating a 2D-3D correspondence may lead to a loss function that conveys erroneous data to the pose estimator. **(c)** NRE does not rely on 2D-3D correspondences, thus $+$ does not exist anymore. Instead, NRE employs a dense loss map (shown in turquoise as a function of $\mathbf{q}$) that contains much more information than RE, especially in ambiguous and multimodal cases. As a result, a pose estimator is significantly more accurate and robust using NRE than RE.

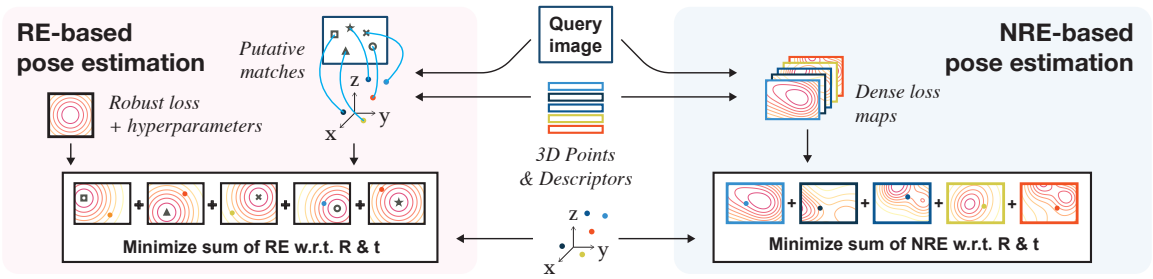


Figure 4.2: NRE-based pose estimator vs. RE-based pose estimator. **Left:** In an RE-based pose estimator, putative 2D-3D correspondences are initially established. Then, a sum of RE terms is minimized, w.r.t. the camera pose, using these correspondences, a robust loss and its hyperparameters as inputs. **Right:** In an NRE-based pose estimator, dense loss maps are computed instead of 2D-3D correspondences. Then the query pose is estimated by minimizing a sum of NRE terms, effectively leveraging richer information than simple 2D-3D correspondences and alleviating the need for choosing a robust loss and its hyperparameters.

4.2 Background and notations

Let us begin by recalling previous notations and set the context we will be working in. In this chapter, we assume a sparse 3D point cloud $\mathcal{M} = \{\mathbf{u}_n^{\mathtt{G}}\}_{n=1...M}$, whose coordinates are expressed in a global coordinate system $\mathtt{G}$, as well as a database $\mathcal{D}$ of geo-localized (w.r.t. $\mathtt{G}$) reference images are given, and we seek to estimate the pose (i.e. the rotation matrix $\mathtt{R}_{\mathtt{QG}}$ and the translation vector $\mathbf{t}_{\mathtt{QG}}$) of a query image $\mathtt{I}_{\mathtt{Q}}$ coming from a calibrated camera.

Dense descriptors $\mathtt{H}_{\mathtt{Q}}$ of $\mathtt{I}_{\mathtt{Q}}$ are extracted using a convolutional neural network $\mathcal{F}$ with parameters Θ: $\mathtt{H}_{\mathtt{Q}} := \mathcal{F}(\mathtt{I}_{\mathtt{Q}}; \Theta)$. Similarly, $\mathcal{F}$ is used to compute a set of descriptors $\{\mathbf{h}_n\}_{n=1...M}$ for each 3D point $\{\mathbf{u}_n^{\mathtt{G}}\}_{n=1...M}$ in the database $\mathcal{D}$.

The warping function $\omega(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}) := \mathtt{K}\pi(\mathtt{R}_{\mathtt{QG}}\mathbf{u}_n^{\mathtt{G}} + \mathbf{t}_{\mathtt{QG}})$ allows to warp a 3D point $\mathbf{u}_n^{\mathtt{G}}$ to obtain a 2D point $\mathbf{p}_n^{\mathtt{Q}}$ onto the image plane of $\mathtt{I}_{\mathtt{Q}}$, i.e. $\mathbf{p}_n^{\mathtt{Q}} = \omega(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}})$, where $\mathtt{K}$ is the camera calibration matrix and $\pi(\mathbf{u}) := [\mathbf{u}_x/\mathbf{u}_z, \mathbf{u}_y/\mathbf{u}_z]^\mathtt{T}$ is the projection function.

Let us now introduce the concept of *correspondence map*. In this chapter, the correspondence map $\mathtt{C}_{\mathtt{Q},n}$ of $\mathbf{u}_n^{\mathtt{G}}$ in $\mathtt{I}_{\mathtt{Q}}$ is computed as follows: $\mathtt{C}_{\mathtt{Q},n} = g(\mathbf{h}_n * \mathtt{H}_{\mathtt{Q}})$ where g is the softmax function and $*$ is the spatial convolution operator. The value $\mathtt{C}_{\mathtt{Q},n}(\mathbf{p}_n^{\mathtt{Q}})$ describes how likely it is that pixel location $\mathbf{p}_n^{\mathtt{Q}}$ in $\mathtt{I}_{\mathtt{Q}}$ corresponds to $\mathbf{u}_n^{\mathtt{G}}$. $\mathtt{C}_{\mathtt{Q},n}$ also has an extra category $\mathbf{p}_n^{\mathtt{Q}} = \mathbf{out}$ that corresponds to the case where $\mathbf{u}_n^{\mathtt{G}}$ is not seen in $\mathtt{I}_{\mathtt{Q}}$. By definition, $\mathtt{C}_{\mathtt{Q},n}(\mathbf{p}_n^{\mathtt{Q}} = \mathbf{out}) := 0$. Thus, $\mathtt{C}_{\mathtt{Q},n}$ has $|\mathring{\Omega}_{\mathtt{Q}}| = 1 + H_{\mathtt{Q}} \times W_{\mathtt{Q}}$ categories, where $H_{\mathtt{Q}}$ and $W_{\mathtt{Q}}$ are the number of rows and columns of $\mathtt{H}_{\mathtt{Q}}$, $\Omega_{\mathtt{Q}}$ is the set of all the pixel locations in $\mathtt{H}_{\mathtt{Q}}$ and $\mathring{\Omega}_{\mathtt{Q}} := \{\Omega_{\mathtt{Q}}, \mathbf{out}\}$.

The following notations will also be useful: $[\![\cdot]\!]$ is the Iverson bracket ($[\![\text{True}]\!] = 1$ and $[\![\text{False}]\!] = 0$), $\lfloor\cdot\rfloor$ is the floor function and $\|\cdot\|$ is the L2 norm.

4.3 The Neural reprojection error

In this section, we first introduce the standard RE and then we present our novel NRE.

4.3.1 Reprojection error

Following the definition of the sum of Reprojection Errors (RE) (see Eq. (2.3)), let us define the RE for a single 2D-to-3D correspondence by:

$$\mathrm{RE}\left(\mathbf{u}_n^{\mathtt{G}}, \mathbf{p}_n^{\mathtt{Q}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right) := \psi_\sigma\left(\left\|\mathbf{p}_n^{\mathtt{Q}} - \omega\left(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right)\right\|\right) \tag{4.1}$$

where $\{\mathbf{p}_n^{\mathtt{Q}}, \mathbf{u}_n^{\mathtt{G}}\}$ is a 2D-3D correspondence and $\psi_\sigma(\cdot)$ is a parametric robust loss, such as Geman-McClure or Tukey's biweight (Barron, 2019; Zach & Bourmaud, 2017), that allows to reduce the influence of large residuals. This RE formulation is used by most absolute camera pose estimators. Estimating the camera pose by minimizing a sum of RE terms enforces the 3D model and the query image to be summarized into a set of putative correspondences which results in a significant and irreversible loss of information. This loss of information needs to be compensated as far as possible through the choice of a robust loss and the tuning of its hyperparameters, that usually depend on both the visual content and the outliers distribution. Moreover, outlier correspondences convey erroneous data to the pose estimator. On the contrary, our novel loss, which we introduce in the next section, leverages richer information from the 3D model and the query image than RE and as a result eliminates the need for choosing a robust loss and its hyperparameters.

4.3.2 Our novel loss

Instead of computing the loss as a robust parametric function of the euclidean distance between the reprojected 3D point and its putative 2D correspondent in the query image, our novel loss function evaluates the discrepancy between two probability mass functions (pmf): the *matching* pmf and the *reprojection* pmf. In the rest of this section, we first define these two pmf and then introduce our novel loss.

Matching probability mass function: This pmf describes how likely it is that the descriptor at the 2D image location $\mathbf{p}_n^{\mathtt{Q}}$ in $\mathtt{H}_{\mathtt{Q}}$ corresponds to the descriptor $\mathbf{h}_n$ of the 3D point $\mathbf{u}_n^{\mathtt{G}}$.

$$q_{\mathrm{m}}\left(\mathbf{p}_n^{\mathtt{Q}} | s_n, \mathtt{H}_{\mathtt{Q}}, \mathbf{h}_n\right) := s_n\, \mathtt{C}_{\mathtt{Q},n}\left(\mathbf{p}_n^{\mathtt{Q}}\right) + \frac{1 - s_n}{|\mathring{\Omega}_{\mathtt{Q}}|}\,, \tag{4.2}$$

where the binary selector variable $s_n \in \{0, 1\}$ allows to choose between two components: the predicted correspondence map and the outlier uniform pmf. The latter component introduces robustness against erroneous correspondence maps that may occur because of non-covisibility, occlusions, failure of the deep network, etc. We show in Fig. 4.3(b) an example of the negative logarithm of a correspondence map.

Reprojection probability mass function: This pmf describes how likely it is that a 2D location $\mathbf{p}_n^{\mathtt{Q}} \in \mathring{\Omega}_{\mathtt{Q}}$ corresponds to the reprojection of a 3D point $\mathbf{u}_n^{\mathtt{G}}$ using camera pose $\mathtt{R}_{\mathtt{QG}}$ and $\mathbf{t}_{\mathtt{QG}}$.

$$\begin{aligned} q_{\mathrm{r}}\left(\mathbf{p}_n^{\mathtt{Q}} | \mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right) := \\ \mathtt{w}_{00,n} \left[\!\left[\mathbf{p}_n^{\mathtt{Q}} = \lfloor \omega\left(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right) \rfloor \right]\!\right] + \\ \mathtt{w}_{10,n} \left[\!\left[\mathbf{p}_n^{\mathtt{Q}} = \lfloor \omega\left(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right) \rfloor + [1, 0]^{\mathtt{T}} \right]\!\right] + \\ \mathtt{w}_{01,n} \left[\!\left[\mathbf{p}_n^{\mathtt{Q}} = \lfloor \omega\left(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right) \rfloor + [0, 1]^{\mathtt{T}} \right]\!\right] + \\ \mathtt{w}_{11,n} \left[\!\left[\mathbf{p}_n^{\mathtt{Q}} = \lfloor \omega\left(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right) \rfloor + [1, 1]^{\mathtt{T}} \right]\!\right] , \end{aligned} \tag{4.3}$$

where the weights $\mathtt{w}_{i,j}$ are bilinear interpolation coefficients, i.e.

$$\begin{aligned} \mathtt{w}_{00,n} &:= (1 - x_n)(1 - y_n), & \mathtt{w}_{10,n} &:= x_n (1 - y_n), \\ \mathtt{w}_{01,n} &:= (1 - x_n) y_n, & \mathtt{w}_{11,n} &:= x_n y_n, \end{aligned}$$

with

$$\begin{aligned} x_n &:= \left(\lfloor \omega\left(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right) \rfloor - \omega\left(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right)\right)_x , \\ y_n &:= \left(\lfloor \omega\left(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right) \rfloor - \omega\left(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right)\right)_y . \end{aligned}$$

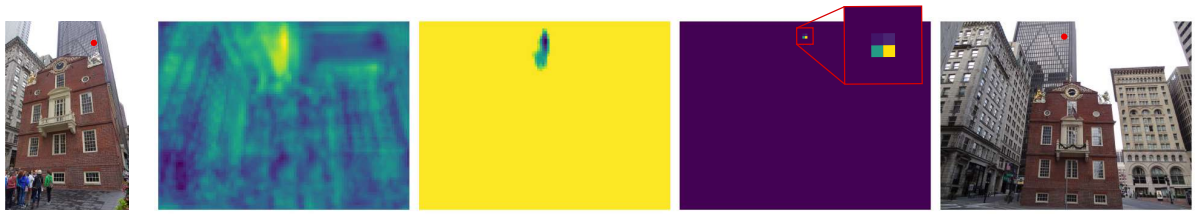
Equation (4.3) sets a non-zero weight to the four image locations surrounding the reprojection of the 3D point $\mathbf{u}_n^{\mathtt{G}}$ under camera pose parameters $\mathtt{R}_{\mathtt{QG}}$ and $\mathbf{t}_{\mathtt{QG}}$, and a zero weight to the rest of the image. In a slight abuse of notation, if a reprojection $\omega\left(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right)$ falls outside of the image boundaries or if the 3D point has negative depth, i.e. $\left(\mathtt{R}_{\mathtt{QG}} \mathbf{u}_n^{\mathtt{G}} + \mathbf{t}_{\mathtt{QG}}\right)_z \leq 0$, we consider that

$$\begin{aligned} \lfloor \omega\left(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right) \rfloor + [\cdot, \cdot]^{\mathtt{T}} &:= \textbf{out} \text{ and} \\ \mathtt{w}_{00,n} := 1, \ \mathtt{w}_{10,n} = \mathtt{w}_{01,n} = \mathtt{w}_{11,n} &:= 0. \end{aligned}$$

We show in Fig. 4.3(d) an example of a reprojection pmf.

Assuming perfect descriptors and a perfect camera pose, the two pmf should be the same. This analysis is the fundamental idea of this chapter (a) given ground truth camera pose, we will make the *matching* pmf fit the *reprojection* pmf to learn descriptors tailored for pose estimation, (b) given descriptors, we will make the *reprojection* pmf fit the *matching* pmf to estimate the camera pose.

We propose to evaluate the discrepancy between the *matching* pmf (Eq. (4.2)) and



(a) (b) (c) (d) (e)

Figure 4.3: **Visualizations of maps involved in the derivation of NRE:** We show an example of **(a)** a source image and a reprojected 3D point $\mathbf{u}_n^{\mathtt{G}}$ using ground truth camera pose, **(b)** the non-robust dense loss map $\tilde{\mathtt{C}}_{\mathtt{Q},n}$ with respect to the target image **(e)**, **(c)** the robust dense loss map $\mathtt{L}_{\mathtt{Q},n}$, and **(d)** the target reprojection probability mass function used at training time.

the *reprojection* pmf (Eq. (4.3)) using the following Cross-Entropy (CE):

$$\begin{aligned}
&\mathrm{CE}\left(q_{\mathrm{r}}\left(\mathbf{p}_n^{\mathtt{Q}}|\mathbf{u}_n^{\mathtt{G}},\mathtt{R}_{\mathtt{QG}},\mathbf{t}_{\mathtt{QG}}\right)||q_{\mathrm{m}}\left(\mathbf{p}_n^{\mathtt{Q}}|s_n,\mathtt{H}_{\mathtt{Q}},\mathbf{h}_n\right)\right)\\
&= -\sum_{\mathbf{p}_n^{\mathtt{Q}}\in\mathring{\Omega}_{\mathtt{Q}}} q_{\mathrm{r}}\left(\mathbf{p}_n^{\mathtt{Q}}|\mathbf{u}_n^{\mathtt{G}},\mathtt{R}_{\mathtt{QG}},\mathbf{t}_{\mathtt{QG}}\right)\ln\left(q_{\mathrm{m}}\left(\mathbf{p}_n^{\mathtt{Q}}|s_n,\mathtt{H}_{\mathtt{Q}},\mathbf{h}_n\right)\right)\\
&= -\sum_{\mathbf{p}_n^{\mathtt{Q}}\in\mathring{\Omega}_{\mathtt{Q}}} q_{\mathrm{r}}\left(\mathbf{p}_n^{\mathtt{Q}}|\mathbf{u}_n^{\mathtt{G}},\mathtt{R}_{\mathtt{QG}},\mathbf{t}_{\mathtt{QG}}\right)\ln\left(s_n\,\mathtt{C}_{\mathtt{Q},n}\left(\mathbf{p}_n^{\mathtt{Q}}\right)+\frac{1-s_n}{|\mathring{\Omega}_{\mathtt{Q}}|}\right)\\
&= s_n\tilde{\mathtt{C}}_{\mathtt{Q},n}\left(\omega\left(\mathbf{u}_n^{\mathtt{G}},\mathtt{R}_{\mathtt{QG}},\mathbf{t}_{\mathtt{QG}}\right)\right)+(1-s_n)\ln|\mathring{\Omega}_{\mathtt{Q}}|\\
&:= \mathrm{NRE}\left(\mathbf{u}_n^{\mathtt{G}},\mathtt{H}_{\mathtt{Q}},\mathbf{h}_n,\mathtt{R}_{\mathtt{QG}},\mathbf{t}_{\mathtt{QG}},s_n\right)
\end{aligned}\tag{4.4}$$

where $\tilde{\mathtt{C}}_{\mathtt{Q},n}(\mathbf{p}) := -\ln\left(\mathtt{C}_{\mathtt{Q},n}(\mathbf{p})\right)\ \forall\mathbf{p}\in\mathring{\Omega}_{\mathtt{Q}}$ is called a *dense loss map*. The notation $\tilde{\mathtt{C}}_{\mathtt{Q},n}\left(\omega\left(\mathbf{u}_n^{\mathtt{G}},\mathtt{R}_{\mathtt{QG}},\mathbf{t}_{\mathtt{QG}}\right)\right)$ corresponds to performing a bilinear interpolation at location $\omega\left(\mathbf{u}_n^{\mathtt{G}},\mathtt{R}_{\mathtt{QG}},\mathbf{t}_{\mathtt{QG}}\right)$ in $\tilde{\mathtt{C}}_{\mathtt{Q},n}$.

From the point of view of the 3D point $\mathbf{u}_n^{\mathtt{G}}$, Eq. (4.4) is a reprojection loss that depends on descriptors extracted by a convolutional neural network $\mathcal{F}$ (see Section 4.2). Thus, we will refer to Eq. (4.4) as the *Neural Reprojection Error*.

From a practical point of view, given query dense descriptors $\mathtt{H}_{\mathtt{Q}}$ as well as 3D points and descriptors $\{\mathbf{u}_n^{\mathtt{G}},\mathbf{h}_n\}_{n=1\ldots M}$, it is possible to estimate the camera pose by minimizing a sum of NRE terms w.r.t. $\mathtt{R}_{\mathtt{QG}}$, $\mathbf{t}_{\mathtt{QG}}$ and $\{s_n\}_{n=1\ldots M}$ (see Section 4.4). Here, the NRE relies on the dense loss maps directly which significantly reduces the amount of lost information compared to RE. Consequently, the need for choosing a robust loss and its hyperparameters is eliminated and all the information is kept available to estimate the camera pose.

Our novel NRE is differentiable not only w.r.t. to the camera pose but also w.r.t. the descriptors $\mathtt{H}_{\mathtt{Q}}$ and $\mathbf{h}_n$. Thus, providing ground-truth camera poses and minimizing NRE w.r.t. the descriptors yields a well-posed feature learning problem tailored for the pose

estimation (see Section 4.5). NRE merges the feature learning problem and the camera pose estimation problem in a new way and allows to rethink the recent end-to-end feature metric pose refinement (see Section 4.6.2).

4.4 Camera pose estimation

Our novel NRE can be used to estimate the camera pose. Given a query image, from which query dense descriptors $\mathtt{H}_\mathtt{Q}$ are extracted, as well as 3D points and descriptors $\{\mathbf{u}^\mathtt{G}_n, \mathbf{h}_n\}_{n=1\dots M}$, we obtain a camera pose estimate by minimizing the following sum of NRE terms (Eq. (4.4)) w.r.t. $\mathtt{R}_\mathtt{QG}$ and $\mathbf{t}_\mathtt{QG}$:

$$\begin{aligned}
&\mathcal{L}\left(\mathtt{R}_\mathtt{QG}, \mathbf{t}_\mathtt{QG}\right) \\
&= \min_{s_1,s_2,\dots,s_M} \sum_{n=1}^{M} \mathrm{NRE}\left(\mathbf{u}^\mathtt{G}_n, \mathtt{H}_\mathtt{Q}, \mathbf{h}_n, \mathtt{R}_\mathtt{QG}, \mathbf{t}_\mathtt{QG}, s_n\right) \\
&= \sum_{n=1}^{M} \min\left(\ln|\mathring{\Omega}_\mathtt{Q}|, \tilde{\mathtt{C}}_{\mathtt{Q},n}\left(\omega\left(\mathbf{u}^\mathtt{G}_n, \mathtt{R}_\mathtt{QG}, \mathbf{t}_\mathtt{QG}\right)\right)\right) \qquad (4.5) \\
&\approx \sum_{n=1}^{M} \mathtt{L}_{\mathtt{Q},n}\left(\omega\left(\mathbf{u}^\mathtt{G}_n, \mathtt{R}_\mathtt{QG}, \mathbf{t}_\mathtt{QG}\right)\right) \qquad (4.6)
\end{aligned}$$

where the loss maps $\mathtt{L}_{\mathtt{Q},n}$ are defined as follows:

$$\mathtt{L}_{\mathtt{Q},n}\left(\mathbf{p}\right) := \min\left(\ln|\mathring{\Omega}_\mathtt{Q}|, \tilde{\mathtt{C}}_{\mathtt{Q},n}\left(\mathbf{p}\right)\right) \forall \mathbf{p} \in \mathring{\Omega}_\mathtt{Q}\,. \qquad (4.7)$$

Instead of performing a bilinear interpolation in $\tilde{\mathtt{C}}_{\mathtt{Q},n}$ followed by a truncation as in Eq. (4.5), we apply a truncation to each element of $\tilde{\mathtt{C}}_{\mathtt{Q},n}$ once (Eq. (4.7)) and then perform a bilinear interpolation (Eq. (4.6)). This approximation enables both a sparse storage of each loss map $\mathtt{L}_{\mathtt{Q},n}$ and an efficient smoothing procedure (see Section 4.4.2).

Our loss function is robust against outliers, since large values in $\tilde{\mathtt{C}}_{\mathtt{Q},n}$ are truncated at $\ln|\mathring{\Omega}_\mathtt{Q}|$, and does not rely on hyperparameters. We show in Fig. 4.3(c) an example of a robust *dense loss map* ($\mathtt{L}_{\mathtt{Q},n}$).

Minimizing Eq. (4.6) is a non-convex optimization problem, thus we proceed in two steps: a sampling-based initialization step followed by gradient-based refinement step.

4.4.1 Initialization step

To obtain an initial pose estimate, we employ an *M-estimator SAmple Consensus* approach (MSAC) (Torr & Zisserman, 2000). The method is very similar to a RANdom

SAmple Consensus approach (RANSAC) (Fischler & Bolles, 1981)) but does not require any user defined inlier/outlier threshold. Each iteration consists of 1) randomly sampling 3 loss maps, 2) estimating a putative camera pose from these 3 loss maps and 3) evaluating Eq. (4.6) with that putative camera pose. Step 2 can be efficiently implemented using a standard P3P solver since:

$$\underset{\mathtt{R}_{\mathtt{QG}},\mathbf{t}_{\mathtt{QG}}}{\arg\min} \sum_{n=1}^{3} \mathtt{L}_{\mathtt{Q},n}\left(\omega\left(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right)\right) = \mathrm{P3P}\left(\left\{\mathbf{u}_n^{\mathtt{G}}, \underset{\mathbf{p}}{\arg\min}\, \mathtt{L}_{\mathtt{Q},n}\left(\mathbf{p}\right)\right\}_{n=1\ldots3}\right). \tag{4.8}$$

4.4.2 Refinement step

Refining the initial camera pose remains a difficult optimization problem since each loss map in Eq. (4.6) may have plateaus and local minima (see Fig. 4.1 middle and bottom rows) and the initial pose estimate may not be accurate enough for a gradient-based method to avoid a poor local minimum.

Thus, we employ a Graduated Non-Convexity approach (GNC) (Blake & Zisserman, 1987) that builds a sequence of successively smoother (and therefore easier to optimize) approximations of the original loss function. The optimization scheme consists of optimizing the sequence of loss functions, with the solution from the previous objective used as starting point for the next one. However, Eq. (4.6) is not a standard robust optimization problem (Zach & Bourmaud, 2018). Therefore, we propose to apply a Gaussian-homotopy-like method (Mobahi & Fisher, 2015) and consider the following smoothed version of the original loss function (a derivation of that equation is given in the appendix):

$$\begin{aligned}&\breve{\mathcal{L}}_{\sigma}\left(\mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right) := \\ &\sum_{n=1}^{M} \sum_{\mathbf{q}\in\Gamma_{\mathtt{Q},n}} \text{-}\Big(\ln\left|\mathring{\Omega}_{\mathtt{Q}}\right|\text{-}\mathtt{L}_{\mathtt{Q},n}\left(\mathbf{q}\right)\Big) k_{\sigma}\Big(\left\|\mathbf{q}\text{-}\omega\left(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}}\right)\right\|\Big)\end{aligned} \tag{4.9}$$

where $k_{\sigma}\left(\|\mathbf{r}\|\right) := \frac{1}{2\pi\sigma^2} e^{-\frac{\|\mathbf{r}\|^2}{2\sigma^2}}$ is an isotropic Gaussian kernel with standard variation σ and $\Gamma_{\mathtt{Q},n}$ is the set of pixel locations whose corresponding values in $\tilde{\mathtt{C}}_{\mathtt{Q},n}$ have not been truncated in Eq. (4.7). In Eq. (4.9), a large value of σ leads to a highly smoothed version of the original loss function while a small value of σ corresponds to a loss function that is very similar to Eq. (4.6). Therefore, in practice, we will start the optimization with a value of σ that is large enough, to avoid getting stuck in a poor local minimum and progressively decrease its value. Since Eq. (4.9) is a standard robust optimization problem, we employ an Iterated Reweighted Least Squares (IRLS) approach to minimize each optimization problem within the GNC (Blake & Zisserman, 1987) and use the stopping

criterion proposed in (Zach & Bourmaud, 2018).

4.4.3 Coarse-to-fine strategy

From a practical point of view, the robustness and the accuracy of the camera pose estimate directly depends on the loss maps, especially their resolution. However, producing high resolution loss maps is an inefficient strategy: most of the computational time would be spent computing cross-correlations in regions distant from the true correspondent locations. Instead, we propose a coarse-to-fine strategy: we first estimate a coarse camera pose using low-resolution loss maps and then refine it using local high-resolution ones.

For a given query image of size $H \times W \times 3$, we proceed as follows: 1) Coarse dense descriptors of size $H/16 \times W/16 \times 1280$ are extracted using a *coarse* network ($\mathcal{F}_{\text{coarse}}$). 2) Low-resolution loss maps of size $H/16 \times W/16$ are computed. 3) We run an MSAC (Torr & Zisserman, 2000)+P3P to obtain an initial coarse pose estimate. 4) We apply a GNC (Blake & Zisserman, 1987) procedure (still using low-resolution correspondence maps) to refine that initial coarse estimate. 5) Fine dense descriptors of size $H/2 \times W/2 \times 288$ are extracted using a *fine* network ($\mathcal{F}_{\text{fine}}$). 6) Local high-resolution loss maps of size 64×64 are computed at the location of the reprojected 3D points using the coarse pose estimate. 7) We apply a GNC (Blake & Zisserman, 1987) procedure starting from the coarse pose estimate to obtain our final pose estimate.

Let us provide some implementation details. Step 6 of our coarse-to-fine strategy consists in computing local high-resolution loss maps of size 64×64 at the location of the reprojected 3D points using the coarse pose estimate. The idea of that step is to transform the low-resolution loss maps into high-resolution loss maps to obtain a much more accurate pose estimate. The question is: How can we combine a low-resolution robust loss map with a local high-resolution discriminative loss map ? We proceed as follows:

1. A coarse correspondence map $\mathtt{C}_{\text{coarse}}$ is of size $H/16 \times W/16$. Let us recall that by definition $\sum_{\mathbf{p} \in \mathring{\Omega}_{\text{coarse}}} \mathtt{C}_{\text{coarse}}(\mathbf{p}) = 1$.

2. Compute the local high resolution correspondence map $\mathtt{C}_{\text{fine}}$ of size 64×64 at the location of the reprojected 3D points (using the coarse pose estimate) $\mathbf{q}$:

   (a) Extract a 64×64 region in the dense fine descriptors around $\mathbf{q}$.

   (b) Compute the dot product with the fine descriptor of the 3D point and apply a softmax to obtain $\mathtt{C}_{\text{fine}}$.

   Thus by definition $\sum_{\mathbf{p} \in \mathcal{N}_{64\times64}(\mathbf{q})} \mathtt{C}_{\text{fine}}(\mathbf{p}) = 1$.

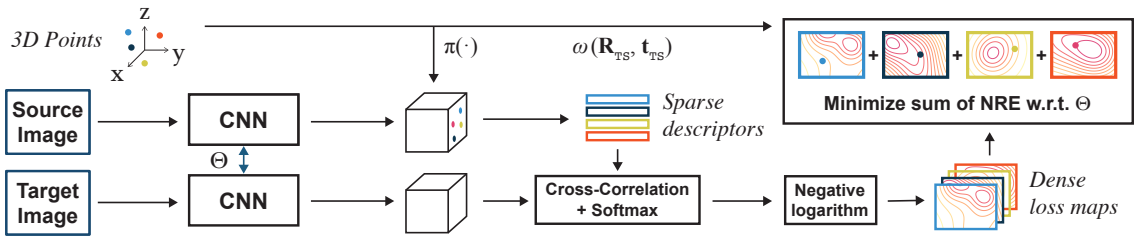


Figure 4.4: **Learning image descriptors tailored for camera pose estimation:** Given a pair of target/source images ($\mathtt{I}_\mathtt{T}$, $\mathtt{I}_\mathtt{S}$), 3D points $\left\{\mathbf{u}_n^\mathtt{S}\right\}_{n=1\ldots M}$ (seen in both $\mathtt{I}_\mathtt{S}$ and $\mathtt{I}_\mathtt{T}$) and ground truth camera poses ($\mathtt{R}_\mathtt{TS}$ and $\mathbf{t}_\mathtt{TS}$), we first extract dense representations for both images. For each 3D, we compute dense loss maps and minimize the NRE loss with respect to the backbone parameters Θ.

3. $\mathtt{C}_{\text{fine}}$ corresponds to a region of size 8x8 in $\mathtt{C}_{\text{coarse}}$. Compute the sum of these 64 pixels in $\mathtt{C}_{\text{coarse}}$. We call this scalar $norm_{coarse}$.

4. Multiply $\mathtt{C}_{\text{fine}}$ by $\frac{norm_{coarse}}{64}$ to obtain $\mathtt{C}_{\text{fine norm}}$. $\mathtt{C}_{\text{fine norm}}$ is a local high-resolution version of $\mathtt{C}_{\text{coarse}}$.

5. The final local high resolution loss map is obtained classically: $\mathtt{L}_{\text{fine}} = \min\left(\ln|\mathring{\Omega}_{\text{fine}}|, -\ln\left(\mathtt{C}_{\text{fine norm}}\right)\right)$. By definition, outside of the 64×64 region, the value of the loss is $\ln|\mathring{\Omega}_{\text{fine}}|$.

This coarse-to-fine strategy allows to obtain a camera pose estimate very efficiently while significantly reducing the amount of required memory, since we never compute or store any high-resolution loss map (see Tab. 4.4).

4.5 Learning image descriptors

Our novel camera pose estimation method (see Section 4.4) essentially consists in minimizing a sum of NRE terms, w.r.t. the camera pose, assuming that the underlying descriptor extractor networks $\mathcal{F}_{\text{coarse}}$ and $\mathcal{F}_{\text{fine}}$ provide robust and discriminative descriptors. Therefore, we need to learn these networks. Let us recall that NRE (Eq. (4.4)) is differentiable w.r.t. the descriptors $\mathtt{H}_\mathtt{Q}$ and $\mathbf{h}_n$. Thus we can learn to extract descriptors using NRE as training loss. We provide pairs of target/source images ($\mathtt{I}_\mathtt{T}$, $\mathtt{I}_\mathtt{S}$), 3D points $\{\mathbf{u}_n^\mathtt{S}\}_{n=1\ldots M}$ (seen in both $\mathtt{I}_\mathtt{S}$ and $\mathtt{I}_\mathtt{T}$) and ground truth camera poses ($\mathtt{R}_\mathtt{TS}$ and $\mathbf{t}_\mathtt{TS}$). For each pair of images, we perform gradient descent over the following loss function (see Fig. 4.4):

$$\mathcal{L}(\Theta) = \sum_{n=1}^{M} \text{NRE}\left(\mathbf{u}_n^\mathtt{S}, \mathtt{H}_\mathtt{T}, \mathbf{h}_n, \mathtt{R}_\mathtt{TS}, \mathbf{t}_\mathtt{TS}, s_n = 1\right), \tag{4.10}$$

with $\mathtt{H}_\mathtt{T} = \mathcal{F}(\mathtt{I}_\mathtt{T}; \Theta)$, $\mathtt{H}_\mathtt{S} = \mathcal{F}(\mathtt{I}_\mathtt{S}; \Theta)$ and $\mathbf{h}_n = \mathtt{H}_\mathtt{S}(\mathtt{K}\pi(\mathbf{u}_n^\mathtt{S}))$. The selector variable s_n is set to one in order to ease the gradient propagation. In practice, this loss function resembles the loss function of S2DNet very closely, and can thus be trained using the same databases and annotations. The difference w.r.t. to S2DNet lies in the bilinear interpolation of the target reprojection pmf, and the usage of a coarse-to-fine procedure. As explained in Section 4.4.3, in practice, we employ two networks: a *coarse* network $\mathcal{F}_{\text{coarse}}$ and a *fine* network $\mathcal{F}_{\text{fine}}$. Thus we need to train two networks with different architectures, which are detailed in the appendix.

4.6 Discussion

4.6.1 RE is a special case of NRE

In RE-based pose estimation, we are given 2D-3D correspondences $\{\mathbf{u}_n^\mathtt{G}, \mathbf{p}_n^\mathtt{Q}\}_{n=1\ldots M}$. Let us consider a single 2D-3D correspondence. Assuming that $\mathbf{p}_n^\mathtt{Q}$ has integer pixel coordinates, we can build a one-hot-encoded correspondence map $\mathtt{C}_{\mathtt{Q},n}$ such that $\mathtt{C}_{\mathtt{Q},n}(\mathbf{p}_n^\mathtt{Q}) = 1$ and zeros everywhere else. In this case, Eq. (4.7) is a dense loss map $\mathtt{L}_{\mathtt{Q},n}$ that equals zero at the location $\mathbf{p}_n^\mathtt{Q}$ and $\ln|\mathring{\Omega}_\mathtt{Q}|$ everywhere else, and Eq. (4.9) becomes:

$$\mathcal{L}_\sigma(\mathtt{R}_\mathtt{QG}, \mathbf{t}_\mathtt{QG}) = \sum_{n=1}^{M} -\ln|\mathring{\Omega}_\mathtt{Q}| k_\sigma\left(\left\|\mathbf{p}_n^\mathtt{Q} - \omega\left(\mathbf{u}_n^\mathtt{G}, \mathtt{R}_\mathtt{QG}, \mathbf{t}_\mathtt{QG}\right)\right\|\right). \tag{4.11}$$

In Eq. (4.11), each term within the sum corresponds to Eq. (4.1) with a negative gaussian function as robust loss, whose shape is similar to the truncated quadratic kernel (Zach & Bourmaud, 2017). Thus, RE is a special case of NRE. In the experiments, we will consider minimizing Eq. (4.11) to fairly compare RE vs. NRE.

4.6.2 NRE vs. End-to-end feature metric pose refinement

End-to-end Feature metric Pose Refinement (FPR) methods (Lv et al., 2019; Tang & Tan, 2019; Von Stumberg et al., 2020) seek to minimize a loss of the following form at "test-time":

$$\mathcal{L}_\sigma(\mathtt{R}_\mathtt{QG}, \mathbf{t}_\mathtt{QG}) := \sum_{i=1}^{M} \psi_\sigma\left(\left\|\mathbf{h}_n - \mathtt{H}_\mathtt{Q}\left(\omega\left(\mathbf{u}_n^\mathtt{G}, \mathtt{R}_\mathtt{QG}, \mathbf{t}_\mathtt{QG}\right)\right)\right\|\right). \tag{4.12}$$

In Eq. (4.12), each term within the sum consists in reprojecting a 3D point into the query image plane but taking the distance in the space of descriptors. From this point of view, FPR is similar to NRE as it tries to leverage richer image information than simple 2D-3D correspondences. However FPR still requires choosing/learning a robust loss and

tuning/learning its hyperparameters, so from this point of view it has the same limitations as RE.

But the major difference between FPR and NRE is that minimizing Eq. (4.12) w.r.t. the descriptors does not yield a well-posed feature learning problem. In order to learn descriptors tailored for pose estimation, FPR methods must consider at least two losses. In (Von Stumberg et al., 2020), a pixelwise contrastive loss is added (as well as a term involving the Hessian of the pose), while (Lv et al., 2019; Sarlin et al., 2021; Tang & Tan, 2019) unroll several steps of an optimizer to obtain a computational graph and use a distance between the ground truth pose and the predicted pose to supervise the training. On the contrary, minimizing NRE w.r.t. the descriptors yields a well-posed feature learning problem. Thus NRE is the first method to unify the feature learning problem and the camera pose estimation problem in a single loss and allows to rethink the end-to-end FPR strategy.

4.7 Experiments

In this section, we experimentally demonstrate that our novel NRE-based pose estimator significantly outperforms state-of-the art RE-based pose estimators. We also show that our coarse-to-fine strategy markedly reduces the amount of required memory and the overall computational time of our NRE-based pose estimator.

4.7.1 Dataset and method

We assembled an evaluation dataset of 3000 Megadepth (Li & Snavely, 2018) image pairs, sampled from the validation set. Using the provided SfM model reconstructed using SIFT (Lowe, 2004), we create image pairs which contain at least 50 covisible 3D points. We evenly split them based on their viewpoint distances to create three difficulty categories, which we name *Easy*, *Medium* and *Hard*. At test-time for every pair of source and target images, we aim at predicting the absolute camera pose of the target image, based on the 3D points visible in the source image. We report the pose estimation error for several precision thresholds.

4.7.2 RE-based vs. NRE-based pose estimator

In this first evaluation, we compare RE-based pose estimators against our novel NRE-based pose estimator. In order to have a fair comparison, we use S2DNet (Germain et al., 2020) features for all methods evaluated in this study.

Baselines: We compare our NRE-based pose estimator against multiple state-of-the-

Features	Pose estimator	Hyperparam.	Translation Error			Rotation Error		
			0.25m	1m	5m	2°	5°	10°
S2DNet	RE *LO-RANSAC* (Chum et al., 2003)	$\tau = 4$	0.54 (+23%)	0.45 (+32%)	0.33 (+32%)	0.54 (+23%)	0.47 (+27%)	0.45 (+32%)
S2DNet	RE *GC-RANSAC* (Baráth & Matas, 2018)	$\tau = 4$	0.54 (+23%)	0.43 (+26%)	0.31 (+24%)	0.53 (+20%)	0.47 (+27%)	0.43 (+26%)
S2DNet	RE *MAGSAC++* (Baráth et al., 2020)	N/A	0.51 (+16%)	0.43 (+26%)	0.31 (+24%)	0.51 (+16%)	0.45 (+22%)	0.42 (+24%)
S2DNet	RE *Minimize Eq. (4.11)*	$\sigma = 5$	0.53 (+20%)	0.44 (+29%)	0.31 (+24%)	0.52 (+18%)	0.46 (+24%)	0.43 (+26%)
S2DNet	NRE	N/A	**0.44** (+ 0%)	**0.34** (+ 0%)	**0.25** (+ 0%)	**0.44** (+ 0%)	**0.37** (+ 0%)	**0.34** (+ 0%)

Table 4.1: **NRE-based vs. RE-based pose estimators:** We evaluate the gain in performance of our novel NRE-based pose estimator against state-of-the-art RE-based pose estimators on the MegaDepth dataset (Li & Snavely, 2018). For a fair comparison, each method employs S2DNet (Germain et al., 2020) features, even our NRE-based pose estimator. For the methods that have an hyperparameter, we optimized it and report the best results. We report the error at several thresholds for translation and rotation (lower is better). The scores between brackets show the relative deterioration w.r.t. to NRE. We find that our NRE-based pose estimator significantly outperforms all the RE-based estimators.

art RE-based pose estimators. This includes LO-RANSAC (Chum et al., 2003), GC-RANSAC (Baráth & Matas, 2018) and MAGSAC++ (Baráth et al., 2020), which all aim at finding inlier correspondences from putative matches. We also add the minimization of Eq. (4.11) and Eq. (4.12).

For all RE-based pose estimators, we follow S2DNet (Germain et al., 2020) and provide raw putative 2D-to-3D matches based on the correspondence map argmax location. For our NRE estimator, we use the same correspondence maps but preserve all the information. For all methods requiring hyperparameter tuning, we run several evaluations to find the optimal one on our dataset. More details are provided in the appendix.

Results: We report pose estimation errors for the aforementionned methods in Tab. 4.1. We find our NRE-based pose estimator consistently provides significant improvements over other RE-based estimators. In addition as shown in Fig. B.1 in the appendix, we find hyperparameter tuning has a significant impact on performance for parametric RE estimators. Our NRE-estimator however, requires no tuning.

4.7.3 NRE-based pose estimator vs. Feature metric Pose Refinement

We compare our novel NRE-based pose estimator against Feature-Metric Pose Refinement (FPR) methods. As explained in Section 4.6.2, FPR methods seek to minimize Section 4.12. As such, FPR benefits from dense information contained in query feature maps, but requires to choose a robust loss function and tune its hyperparameters.

To complement our RE-based vs. NRE-based pose estimators study presented in

Features	Pose estimator	Fusion	ψ	Translation Error			Rotation Error		
				0.25m	1m	5m	2°	5°	10°
S2DNet	RE *MAGSAC++*	N/A	N/A	0.51 (+ 16%)	0.43 (+ 26%)	0.31 (+ 24%)	0.51 (+ 16%)	0.45 (+ 22%)	0.42 (+ 24%)
S2DNet	FPR *Min. Eq. 4.12*	C2F	Huber (Huber, 1964)	0.70 (+ 59%)	0.65 (+ 91%)	0.52 (+108%)	0.69 (+ 57%)	0.63 (+ 70%)	0.58 (+ 71%)
S2DNet	FPR *Min. Eq. 4.12*	C2F	Barron (Barron, 2019)	0.55 (+ 25%)	0.44 (+ 29%)	0.30 (+ 20%)	0.55 (+ 25%)	0.48 (+ 30%)	0.43 (+ 26%)
S2DNet	FPR *Min. Eq. 4.12*	Concat.	Huber (Huber, 1964)	0.49 (+ 11%)	0.42 (+ 24%)	0.30 (+ 20%)	0.48 (+ 9%)	0.44 (+ 19%)	0.42 (+ 24%)
S2DNet	FPR *Min. Eq. 4.12*	Concat.	Barron (Barron, 2019)	0.49 (+ 11%)	0.42 (+ 24%)	0.30 (+ 20%)	0.48 (+ 9%)	0.44 (+ 19%)	0.42 (+ 24%)
S2DNet	NRE	N/A	N/A	**0.44** (+ 0%)	**0.34** (+ 0%)	**0.25** (+ 0%)	**0.44** (+ 0%)	**0.37** (+ 0%)	**0.34** (+ 0%)

Table 4.2: **NRE-based pose estimator vs. Feature-Metric Pose Refinement:** We evaluate the gain in performance of our novel NRE-based pose estimator against the Feature-Metric Pose Estimation (FPR) variant on the MegaDepth dataset. Here FPR consists in minimizing Eq. 4.12 using as initialization the camera pose estimate from RE *MAGSAC++* (Baráth et al., 2020). We find here that minimizing Eq. 4.12 allows to improve the camera pose estimate from *MAGSAC++*, however our novel NRE again shows superior performance, while requiring no robust kernel selection. The scores between brackets show the relative deterioration w.r.t. to NRE.

Tab. 4.1, we propose to reuse S2DNet (Germain et al., 2020) features to perform FPR, initialized from our best RE pose estimator (MAGSAC++ (Baráth et al., 2020)). To merge information from all three feature extraction levels from S2DNet (Germain et al., 2020), we try upsampling and concatenating descriptors, as well as a coarse-to-fine alternative in which we iteratively refine predictions from the previous (coarser) level.

We report pose estimation errors in Tab. 4.2 for FPR and NRE estimators. We show results using the Huber (Huber, 1964) robust loss as well as the Barron (Barron, 2019) loss. We find that NRE performs consistently better while eliminating the need for choosing a robust loss.

4.7.4 Coarse-to-fine experiment

We provide an ablation study in Fig. 4.5 of our coarse-to-fine strategy. We find that each step of our NRE-based estimator brings significant improvements. We now compare the performance coupling the NRE estimator with NRE features trained on the same training set as S2DNet (Germain et al., 2020), using our coarse-to-fine strategy. We report in Tab. 4.3 the pose estimation error on all categories from our Megadepth (Li & Snavely, 2018) benchmark. We find that using NRE features brings an additional leap in performance, by up to 200%. Thanks to our coarse-to-fine formulation, this is all achieved at a fraction of the cost of S2DNet (Germain et al., 2020). As reported in Tab. 4.4, NRE features have a memory footprint which is over 16 times lighter, while also performing a lot faster. This is a key component for practical applications, or when scaling up to larger amount of keypoints or images.

Category	Features	Pose estimator	Translation Error			Rotation Error		
			0.25m	1m	5m	2°	5°	10°
Easy	S2DNet	NRE	0.17 (+ 42%)	0.12 (+100%)	0.09 (+200%)	0.16 (+ 45%)	0.13 (+ 86%)	0.10 (+100%)
	NRE Features	NRE	**0.12** (+ 0%)	**0.06** (+ 0%)	**0.03** (+ 0%)	**0.11** (+ 0%)	**0.07** (+ 0%)	**0.05** (+ 0%)
Medium	S2DNet	NRE	0.29 (+ 53%)	0.20 (+ 67%)	0.15 (+150%)	0.27 (+ 60%)	0.22 (+ 69%)	0.19 (+ 90%)
	NRE Features	NRE	**0.19** (+ 0%)	**0.12** (+ 0%)	**0.06** (+ 0%)	**0.17** (+ 0%)	**0.13** (+ 0%)	**0.10** (+ 0%)
Hard	S2DNet	NRE	0.44 (+ 30%)	0.34 (+ 42%)	0.25 (+108%)	0.44 (+ 33%)	0.37 (+ 37%)	0.34 (+ 42%)
	NRE Features	NRE	**0.34** (+ 0%)	**0.24** (+ 0%)	**0.12** (+ 0%)	**0.33** (+ 0%)	**0.27** (+ 0%)	**0.24** (+ 0%)

Table 4.3: **NRE-based pose estimator using NRE features vs. NRE-based pose estimators using S2DNet features:** We evaluate the gain in performance of our NRE features against S2DNet (Germain et al., 2020) features using the same NRE-based pose estimator. We compare pose estimation on Megadepth (Li & Snavely, 2018) images evenly split in three difficulty categories. We report the error at several thresholds for translation and rotation (lower is better). The scores between brackets show the relative deterioration w.r.t. to NRE features. We show that using our NRE features, the resulting estimated pose is markedly more accurate than using S2DNet features.

4.7.5 Experiments on Aachen Night

So far, we evaluated the performances of our NRE-based pose estimator on MegaDepth (Li & Snavely, 2018). Here, we run a similar study on the Aachen Night (Sattler et al., 2018, 2012) dataset. This challenging outdoor dataset consists of $4,328$ sparsely sampled daytime database images, and 98 nighttime query images. To have a fair comparison between NRE-based and RE-based pose estimators, we pair each query image with an oracle nearest-neighbor database image and use all of its visible 3D points to predict the query pose. Similar to the MegaDepth study, we report results for RE-based, FPR-based and NRE-based pose estimators, using S2DNet features in Tab. 4.5. For FPR-based pose estimators we pick the best configuration from 4.2.

As in the MegaDepth experiment, our NRE-based pose estimator consistently provides significant improvement over other pose estimators. We also compare the performance coupling the NRE-based pose estimator with NRE features trained on the same training set as S2DNet (Germain et al., 2020). We report in Tab. 4.6 the pose estimation errors. We again find that using NRE features brings an additional leap in performance.

4.7.6 Experiments on InLoc

To evaluate the generalization capabilities in an indoor scenario, we run the same experiment on the InLoc (Taira et al., 2018) dataset. This dataset consists of 329 query images, for $9,972$ database images. Unlike Aachen Night, we have access to dense aligned depth maps for all database images. To provide a fair comparison, we also pair each query image

Features Pose estimator	S2DNet RE	S2DNet NRE	NRE NRE
Feature extraction	28.2ms	28.2ms	N/A
Feature extraction *coarse*	N/A	N/A	15.5ms
Feature extraction *fine*	N/A	N/A	7.2ms
Compute correspondence maps	300ms	300ms	N/A
Compute *coarse* correspondence maps	N/A	N/A	8ms
Compute *local fine* correspondence maps	N/A	N/A	3ms
Pose initialization (single iteration)	0.9ms	1.1ms	1.1ms
Pose refinement	0.11s	0.61s	N/A
Pose refinement *coarse*	N/A	N/A	0.15s
Pose refinement *fine*	N/A	N/A	0.28s
Total features memory	2949MB	2949MB	591MB
Total correspondence maps memory	7680MB	7680MB	46MB

Table 4.4: **Computational time and memory requirement study:** We report the average inference time on Megadepth (Li & Snavely, 2018) images with 1000 3D points. We show that our coarse-to-fine approach enables a much faster pose estimation and allows for larger scene scaling.

with an oracle nearest-neighbor database image and use SuperPoint (Detone et al., 2018) detections (lifted to 3D using the depth maps) in the database images as inputs. Results are reported in Tab. 4.5.

We find that our NRE-based pose estimator provides consistent improvements at the coarsest threshold, and overall competitive performance on the medium and fine ones. The fact the relative improvement brought by our NRE-based pose estimator is not as significant as for the other datasets can be attributed to the domain shift with respect to the training images. Nonetheless, despite being trained on outdoor images we find that our NRE features bring additional improvements compared to S2DNet (Germain et al., 2020) features, as shown in Tab. 4.6.

4.7.7 Qualitative results

In Fig. 4.6, we show several examples of query images from the MegaDepth (Li & Snavely, 2018) validation set with a reprojected 3D point and the corresponding coarse dense loss map computed using our coarse NRE features. It highlights that the dense loss maps keep much more information than RE. As a consequence, as we show in our experiments, our novel NRE-based pose estimator significantly outperforms RE-based pose estimators.

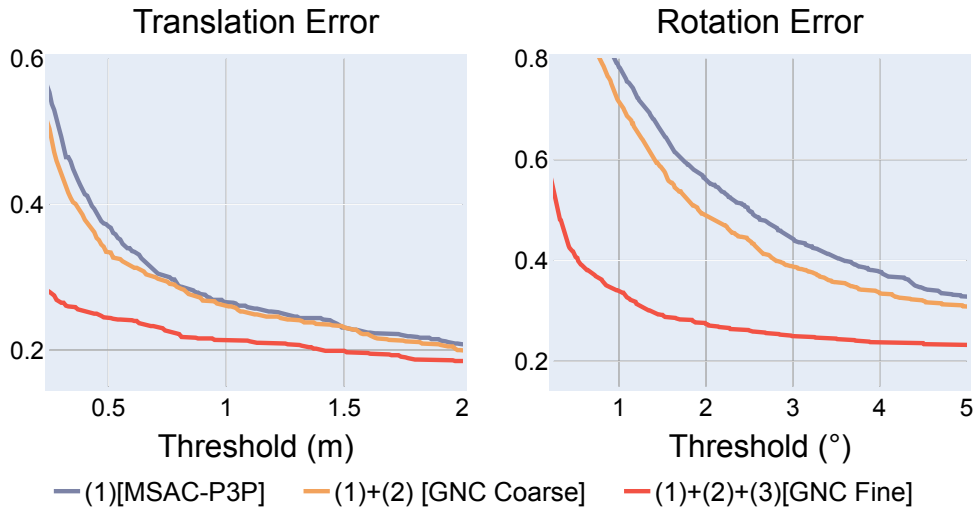


Figure 4.5: **Ablation study:** We report the cumulative error curves in pose estimation (lower is better), on the hardest category of our Megadepth study. We find that each step of our NRE-based coarse-to-fine estimator brings significant improvements.

Figure 4.6: **Qualitative results:** These qualitatives results correspond to additional examples for columns (a) and (b) in Fig.4.1. It highlights that the dense loss maps keep much more information than RE. As a consequence our novel NRE-based pose estimator significantly outperforms RE-based pose estimators.

Features	Pose Estimator	Aachen Night			InLoc-DUC1			InLoc-DUC2		
		0.25m, 2°	0.5m, 5°	5m, 10°	0.25m, 2°	0.5m, 5°	5m, 10°	0.25m, 2°	0.5m, 5°	5m, 10°
S2DNet	MAGSAC++	0.46 (+ 55%)	0.28 (+ 80%)	0.10 (+229%)	0.62 (+ 3%)	0.41 (+ 2%)	0.31 (+ 11%)	0.70 (+ 11%)	0.44 (+ 5%)	0.30 (+ 2%)
S2DNet	RE *Min. Eq. 10*	0.32 (+ 7%)	0.20 (+ 27%)	0.08 (+165%)	**0.58** (- 4%)	0.40 (+ 1%)	0.31 (+ 13%)	0.66 (+ 6%)	0.47 (+ 13%)	0.39 (+ 31%)
S2DNet	FPR *Min. Eq. 11*	0.32 (+ 7%)	0.20 (+ 27%)	0.06 (+ 97%)	0.61 (+ 1%)	0.41 (+ 4%)	0.29 (+ 4%)	0.63 (+ 1%)	**0.41** (- 4%)	0.31 (+ 5%)
S2DNet	NRE	**0.30** (+ 0%)	**0.15** (+ 0%)	**0.03** (+ 0%)	0.60 (+ 0%)	**0.39** (+ 0%)	**0.28** (+ 0%)	**0.62** (+ 0%)	0.42 (+ 0%)	**0.29** (+ 0%)

Table 4.5: **NRE-based vs. RE-based vs. FPR-based pose estimators on Aachen Night (Sattler et al., 2018) and InLoc (Taira et al., 2018):** We evaluate the gain in performance of our novel NRE-based pose estimator against state-of-the-art RE-based and FPR-based pose estimators. For a fair comparison, *each method uses the same oracle nearest-neighbor database image for each query image*. Moreover, each method employs S2DNet (Germain et al., 2020) features, even our NRE-based pose estimator. For the methods that have an hyperparameter, we optimized it and report the best results. We report the error at several thresholds for translation and rotation (lower is better). The scores between brackets show the relative deterioration w.r.t. to NRE. On Aachen, there is no strong domain shift w.r.t. MegaDepth images that are used to train S2DNet, as a result the dense loss maps are accurate and our NRE-based pose estimator significantly outperforms its competitors. On InLoc, there is a strong domain shift (InLoc is an indoor dataset), as a result the dense loss maps are not very informative and our NRE-based pose estimator does not significantly outperform its competitors.

Features	Pose Estim.	Aachen Night			InLoc-DUC1			InLoc-DUC2		
		0.25m, 2°	0.5m, 5°	5m, 10°	0.25m, 2°	0.5m, 5°	5m, 10°	0.25m, 2°	0.5m, 5°	5m, 10°
S2DNet	NRE	0.30 (+ 12%)	0.15 (+ 37%)	0.03 (+ 55%)	0.60 (+ 1%)	0.40 (+ 3%)	0.28 (+ 10%)	0.63 (+ 1%)	0.42 (+ 10%)	0.30 (+ 3%)
NRE Features	NRE	**0.26** (+ 0%)	**0.11** (+ 0%)	**0.02** (+ 0%)	**0.59** (+ 0%)	**0.39** (+ 0%)	**0.25** (+ 0%)	**0.62** (+ 0%)	**0.38** (+ 0%)	**0.29** (+ 0%)

Table 4.6: **NRE features vs. S2DNet features for NRE-based pose estimators on Aachen Night (Sattler et al., 2018) and InLoc (Taira et al., 2018):** We evaluate the gain in performance of our NRE features against S2DNet (Germain et al., 2020) features using the same NRE-based pose estimator. We compare pose estimation on Aachen Night (Sattler et al., 2018) and InLoc (Taira et al., 2018) images. For a fair comparison, *each method uses the same oracle nearest-neighbor database image for each query image*. We report the error at several precision thresholds for translation and rotation (lower is better). The scores between brackets show the relative deterioration w.r.t. to NRE features. On Aachen, there is no strong domain shift w.r.t. MegaDepth images that are used to train both S2DNet and our NRE feature, as a result the dense loss maps are accurate and we obtain improvements similar to the ones we obtained in our MegaDepth experiment. On InLoc, there is a strong domain shift (InLoc is an indoor dataset), as a result neither S2DNet dense loss maps nor the dense loss maps obtained using our NRE features are very informative. As a result, the pose estimated using NRE features is not markedly more accurate than the pose obtained using S2DNet features.

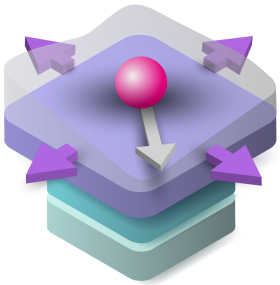
4.8 Conclusion

In this chapter, we introduced the *Neural Reprojection Error* (NRE) as a substitute for the widely used Reprojection Error (RE). NRE allows to perform absolute camera pose estimation by leveraging richer information than RE and eliminates the need for choosing a robust loss and its hyperparameters. We also proposed a coarse-to-fine optimization strategy that allows to very efficiently minimize a sum of NRE terms w.r.t. the camera pose. We experimentally demonstrated that replacing RE with NRE significantly improved the accuracy and the robustness of the camera pose estimate while being computationally and memory highly efficient. Our derivation of NRE merges the feature learning problem and the absolute camera pose estimation problem in a new way that allows to rethink the end-to-end feature-metric pose refinement strategy. From a broader point of view, we believe this new way of merging deep learning and 3D geometry may be useful in other computer vision applications.

So far we have always assumed keypoints correspondences to be covisible at training-time, thus when presented with keypoints in the source image that are not visible in the target image at test-time the NRE suffers from noisy loss maps. Instead of trying to estimate covisibility across images, in the next chapter, we will reuse the NRE pose estimation framework for a more ambitious task: hallucinating correspondences in *non-covisible* image areas.

Chapter 5

Visual Correspondence Hallucination



5.1 Introduction

Establishing correspondences between two partially overlapping images is a fundamental computer vision problem with many applications. As discussed in Chapter 2, state-of-the-art methods for visual localization from an input image rely on keypoint matches between the input image and a reference image (Revaud et al., 2019; Sarlin et al., 2019, 2020; Sattler et al., 2018). However, these local feature matching methods will still fail when few keypoints are *covisible*, *i.e.* when many image locations in one image are outside the field of view or become occluded in the second image. This is also valid for the NRE-based camera pose estimator, which assumes all keypoints to be covisible ($s_n = 1$).

These failures are to be expected since these methods are pure pattern recognition approaches that seek to *identify* correspondences, i.e. to find correspondences in covisible regions, and consider the non-covisible regions as noise. By contrast, humans explain the presence of these non-covisible regions through geometric reasoning and consequently are able to *hallucinate* correspondences at those locations.

Geometric reasoning has already been used in computer vision for image matching, but usually as an *a posteriori* processing (Baráth & Matas, 2018; Barath et al., 2019; Baráth et al., 2020; Chum et al., 2003, 2005; Fischler & Bolles, 1981; Luong & Faugeras, 1996). These methods seek to remove outliers from the set of correspondences produced by a local feature matching approach using only limited geometric models such as epipolar geometry or planar assumptions.

Contributions. In this chapter we tackle the problem of correspondence hallucination. In doing so we seek to answer two questions: (i) can we derive a network architecture able to learn to hallucinate correspondences? and (ii) is correspondence hallucination beneficial for absolute pose estimation?

The answer to these questions is the main novelty of this paper. More precisely, we consider a network that takes as input a pair of partially overlapping source/target images and keypoints in the source image, and outputs for each keypoint a probability distribution over its correspondent's location in the target image plane. We propose to train this network to both identify and hallucinate the keypoints' correspondents. We call the resulting method NeurHal, for Neural Hallucinations. To the best of our knowledge, learning to hallucinate correspondences is a virgin territory, thus we first provide an analysis of the specific features of that novel learning task. This analysis guides us towards employing an appropriate loss function and designing the architecture of the network. After training the network, we experimentally demonstrate that it is indeed able to hallucinate correspondences on unseen pairs of images. We also apply this network to a camera pose estimation problem and find it is significantly more robust than state-of-the-art local feature matching-based competitors.

In this chapter, we will first review related work in Section 5.2 and present an analysis of the problem followed by a solution proposal in Section 5.3. In Section 5.4 we will report quantitative and qualitative experiments to demonstrate both the ability of our model to perform correspondence hallucination but also its effect on absolute camera pose estimation on low-overlap image pairs. Lastly we will discuss the limitations of our approach in Section 5.5.

5.2 Related Work

To the best of our knowledge, aiming at hallucinating visual correspondences has never been done but the related fields of local feature description and matching are immensely vast, and we focus here only on recent learning-based approaches. We refer the reader to Chapter 2 for more details.

Learning-based local feature description. Using deep neural networks to learn to compute local feature descriptors have shown to bring significant improvements in invariance to viewpoint and illumination changes compared to handcrafted methods (Balntas et al., 2017; Csurka & Humenberger, 2018; Gauglitz et al., 2011; Salahat & Qasaimeh, 2017). Most methods learn descriptors locally around pre-computed *covisible* interest regions in both images (Balntas et al., 2016a; Detone et al., 2018; Luo et al., 2019; Yi et al., 2016), using convolutional-based siamese architectures trained with a triplet loss (Balntas et al., 2016b; Gordo et al., 2016b; Schroff et al., 2015), contrastive loss (Radenović et al., 2016) or variants (Mishchuk et al., 2017; Simonyan et al., 2014). To further improve the performances, (Dusmanu et al., 2019; Revaud et al., 2019) propose to jointly learn to detect and describe keypoints in both images, while (Germain et al., 2020) only detects keypoints in one image and matches against dense descriptors in the other image.

Learning-based local feature matching. All the methods described in the previous paragraph establish correspondences by comparing descriptors using a simple operation such as a dot product. Thus the combination of such a simple matching method with a siamese architecture inevitably produces outlier correspondences, especially in non-covisible regions. To reduce the amount of outliers, most approaches employ so-called Mutual Nearest Neighbor (MNN) filtering. However, it is possible to go beyond a simple MNN and learn to match descriptors. Learning-based matching methods (Brachmann & Rother, 2019b; Choy et al., 2020, 2016; Moo yi et al., 2018; Sun et al., 2020; Zhang et al., 2019) take as input local descriptors and/or putative correspondences, and learn

to output correspondences probabilities. However, all these matching methods focus only on predicting correctly covisible correspondences.

Jointly learning local feature description and matching. Several methods have recently proposed to jointly learn to compute and match descriptors (Li et al., 2020a; Rocco et al., 2020, 2018; Sarlin et al., 2020; Sun et al., 2021). All these methods use a siamese Convolutional Neural Network (CNN) to obtain dense local descriptors, but they significantly differ regarding the way they establish matches. They actually fall into two categories. The first category of methods (Li et al., 2020a; Rocco et al., 2020, 2018) computes a 4D correlation tensor that essentially represents the scores of all the possible correspondences. This 4D correlation tensor is then used as input to a second network that learns to modify it using soft-MNN and 4D convolutions. Instead of summarizing all the information into a 4D correlation tensor, the second category of methods(Sarlin et al., 2020; Sun et al., 2021) rely on so-called Transformers (Caron et al., 2021; Cordonnier et al., 2020; Dosovitskiy et al., 2020; Katharopoulos et al., 2020; Ramachandran et al., 2019; Vaswani et al., 2017; Zhao et al., 2020) to let the descriptors of both images communicate and adapt to each other. All these methods again focus on identifying correctly covisible correspondences and consider non-covisible correspondences as noise. While our architecture is closely related to the second category of methods as we also rely on Transformers, the motivation for using it is quite different since it is our goal of hallucinating correspondences that calls for a non-siamese architecture.

Visual content hallucination. Perhaps closest to our goal of hallucinating correspondences given a pair of partially overlapping images, is (Cai et al., 2021) that seeks to estimate a relative rotation between two non-overlapping images by learning to reason about "hidden" cues such as direction of shadows in outdoor scenes, parallel lines or vanishing points. The work of Yang et al. (2019b) proposes to hallucinate the content of RGB-D scans to perform relative pose estimation between two images. More recently (Chen et al., 2021) regresses distributions over relative camera poses for spherical images using joint processing of both images, and manages to recover relative poses despite very limited visual overlap. The work of (Jin et al., 2021a; Qian et al., 2020; Yang et al., 2020) shows that employing a *hallucinate-then-match* paradigm can be a reliable way of recovering 3D geometry or relative pose from sparsely sampled images. In this work, we focus on the problem of *correspondence* hallucination which unlike previously mentioned approaches does not aim at recovering explicit visual content or directly regressing a relative camera pose.

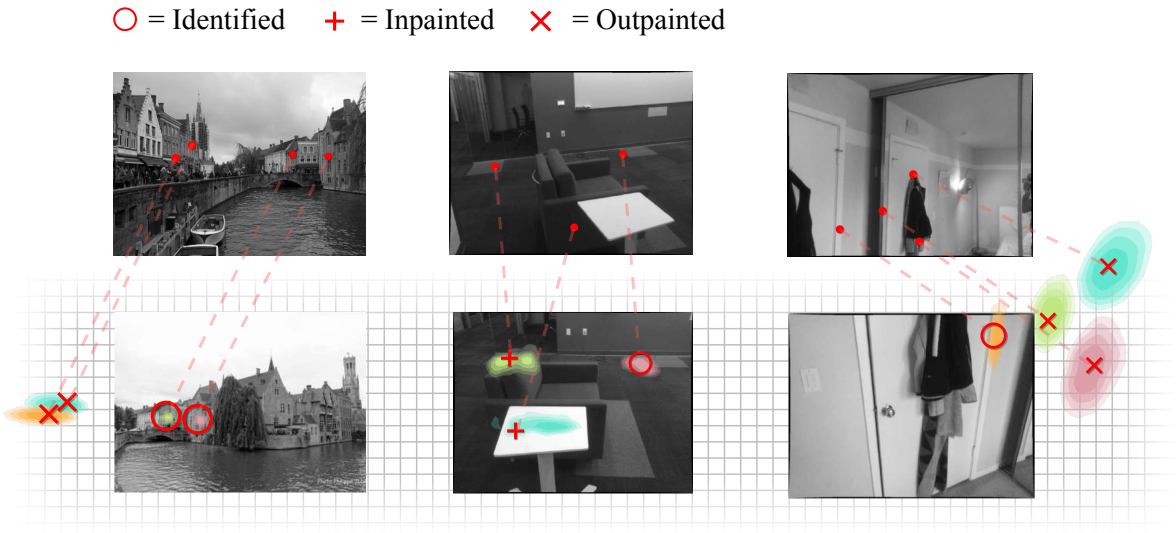


Figure 5.1: **Visual correspondence hallucination.** Our network, called NeurHal, takes as input a pair of partially overlapping source and target images and a set of keypoints detected in the source image, and outputs for each keypoint a probability distribution over its correspondent's location in the target image. When the correspondent is actually visible, its location can be *identified*; when it is not, its location must be *hallucinated*. Two types of hallucination tasks can be distinguished: 1) if the correspondent is occluded, its location has to be *inpainted*; 2) if it is outside the field of view of the target image, its location needs to be *outpainted*. NeurHal generalizes to scenes not seen during training: For each of these three pairs of source/target images coming from the test scenes of ScanNet (Dai et al., 2017) and MegaDepth (Li & Snavely, 2018), we show (top row) the source image with a small subset of keypoints, and (bottom row) the target image with the probability distributions predicted by our network and the ground truth correspondents: ○ for the identified correspondents, + for the inpainted ones, and × for the outpainted correspondents.

5.3 Our approach

Our goal is to train a network that takes as input a pair of partially overlapping source/target images and keypoints in the source image, and outputs for each keypoint a probability distribution over its correspondent's location in the target image plane, regardless of this correspondent being visible, occluded, or outside the field of view. While the problem of learning to find the location of a *visible* correspondent received a lot of attention in the past few years (see Chapter 2), to the best of our knowledge, this work is the first attempt of learning to find the location of a correspondent regardless of this correspondent being visible, occluded, or outside the field of view. Since this learning task is virgin territory, we first analyze its specific features below, before defining a loss function and a network architecture able to handle these features.

5.3.1 Analysis of the problem

The task of finding the location of a correspondent regardless of this correspondent being visible, occluded, or outside the field of view actually leads to three different problems. Before stating those three problems, let us first recall the notion of correspondent from the previous chapters as it is the keystone of our problem.

Correspondent. Given a keypoint $\mathbf{p}_\mathtt{S} \in \mathbb{R}^2$ in the source image $\mathtt{I}_\mathtt{S}$, its depth $d_\mathtt{S} \in \mathbb{R}^+$, and the relative camera pose $\mathtt{R}_\mathtt{TS} \in \mathrm{SO}(3)$, $\mathbf{t}_\mathtt{TS} \in \mathbb{R}^3$ between the coordinate systems of $\mathtt{I}_\mathtt{S}$ and the target image $\mathtt{I}_\mathtt{T}$, the *correspondent* $\mathbf{p}_\mathtt{T} \in \mathbb{R}^2$ of $\mathbf{p}_\mathtt{S}$ in the target image plane is obtained by warping $\mathbf{p}_\mathtt{S}$: $\mathbf{p}_\mathtt{T} := \omega\left(d_\mathtt{S}, \mathbf{p}_\mathtt{S}, \mathtt{R}_\mathtt{TS}, \mathbf{t}_\mathtt{TS}\right) := \mathtt{K}\pi\left(d_\mathtt{S}\mathtt{R}_\mathtt{TS}\mathtt{K}^{-1}\mathbf{p}_\mathtt{S} + \mathbf{t}_\mathtt{TS}\right)$, where $\mathtt{K}$ is the camera calibration matrix[1] and $\pi\left(\mathbf{u}\right) := \left[\mathbf{u}_x/\mathbf{u}_z, \mathbf{u}_y/\mathbf{u}_z, 1\right]^\mathtt{T}$ is the projection function. In a slight abuse of notation, we do not distinguish a homogeneous 2D vector from a non-homogenous 2D vector. Let us highlight that the correspondent $\mathbf{p}_\mathtt{T}$ of $\mathbf{p}_\mathtt{S}$ may not be *visible*, i.e. it may be occluded or outside the field of view.

Identifying the correspondent. In the case where a network has to establish a correspondence between a keypoint $\mathbf{p}_\mathtt{S}$ in $\mathtt{I}_\mathtt{S}$ and its *visible* correspondent $\mathbf{p}_\mathtt{T}$ in $\mathtt{I}_\mathtt{T}$, standard approaches, such as comparing a local descriptor computed at $\mathbf{p}_S$ in $\mathtt{I}_\mathtt{S}$ with descriptors computed at detected keypoints in $\mathtt{I}_\mathtt{T}$, are applicable to *identify* the correspondent $\mathbf{p}_\mathtt{T}$.

Outpainting the correspondent. When $\mathbf{p}_\mathtt{T}$ is outside the field of view of $\mathtt{I}_\mathtt{T}$, there is nothing to identify, i.e. neither can $\mathbf{p}_\mathtt{T}$ be detected as a keypoint nor can a local descriptor be computed at that location. Here the network first needs to identify correspondences in

[1] We assumed *w.l.o.g.* the source image $\mathtt{I}_\mathtt{S}$ and the target image $\mathtt{I}_\mathtt{T}$ are rectified images according to the pinhole camera model.

the region where $\mathtt{I}_\mathtt{T}$ overlaps with $\mathtt{I}_\mathtt{S}$ and realize that the correspondent $\mathbf{p}_\mathtt{T}$ is outside the field of view to eventually *outpaint* it (see Fig. 5.1). We call this operation "outpainting the correspondent" as the network needs to hallucinate the location of $\mathbf{p}_\mathtt{T}$ outside the field of view of $\mathtt{I}_\mathtt{T}$.

Inpainting the correspondent. When $\mathbf{p}_\mathtt{T}$ is occluded in $\mathtt{I}_\mathtt{T}$, the problem is even more difficult since local features can be computed at that location but will not match the local descriptor computed at $\mathbf{p}_\mathtt{S}$ in $\mathtt{I}_\mathtt{S}$. As in the outpainting case, the network needs to identify correspondences in the region where $\mathtt{I}_\mathtt{T}$ overlaps with $\mathtt{I}_\mathtt{S}$ and realize that the correspondent $\mathbf{p}_\mathtt{T}$ is occluded to eventually *inpaint* the correspondent $\mathbf{p}_\mathtt{T}$ (see Fig. 5.1). We call this operation "inpainting the correspondent" as the network needs to hallucinate the location of $\mathbf{p}_\mathtt{T}$ behind the occluding object.

Let us now introduce a loss function and an architecture that are able to unify the identifying, inpainting and outpainting tasks.

5.3.2 Loss function

The distinction we made between the identifying, inpainting and outpainting tasks come from the fact that the source image $\mathtt{I}_\mathtt{S}$ and the target image $\mathtt{I}_\mathtt{T}$ are the projections of the same 3D environment from two different camera poses. In order to integrate this idea and obtain a unified correspondence learning task, we rely on the Neural Reprojection Error. Let us recall its formulation. Given the correspondence map $\mathtt{C}_\mathtt{T}$ of $\mathbf{p}_\mathtt{S}$ in the image plane of $\mathtt{I}_\mathtt{T}$ defined by $\mathtt{C}_\mathtt{T}(\mathbf{p}_\mathtt{T}) := p(\mathbf{p}_\mathtt{T}|\mathbf{p}_\mathtt{S}, \mathtt{I}_\mathtt{S}, \mathtt{I}_\mathtt{T})$, we define the NRE by:

$$\text{NRE}(\mathbf{p}_\mathtt{S}, \mathtt{C}_\mathtt{T}, \mathtt{R}_\mathtt{TS}, \mathbf{t}_\mathtt{TS}, d_\mathtt{S}) := -\ln \mathtt{C}_\mathtt{T}(\mathbf{x}_\mathtt{T}) \text{ where } \mathbf{x}_\mathtt{T} = \mathtt{K}_\mathtt{C}\omega(d_\mathtt{S}, \mathbf{p}_\mathtt{S}, \mathtt{R}_\mathtt{TS}, \mathbf{t}_\mathtt{TS}) \ . \tag{5.1}$$

The likelihood $\mathtt{C}_\mathtt{T}$ of $\mathbf{p}_\mathtt{T}$ being the correspondent of $\mathbf{p}_\mathtt{S}$ can only be evaluated for $\mathbf{p}_\mathtt{T} \in \Omega_{\mathtt{C}_\mathtt{T}}$ where $\Omega_{\mathtt{C}_\mathtt{T}}$ is the set of all the pixel locations in $\mathtt{C}_\mathtt{T}$. Here, we implicitly defined that the likelihood of $\mathbf{p}_\mathtt{T}$ falling outside the boundaries of $\mathtt{C}_\mathtt{T}$ is zero. In practice, a correspondence map $\mathtt{C}_\mathtt{T}$ is implemented as a neural network that takes as input $\mathbf{p}_\mathtt{S}$, $\mathtt{I}_\mathtt{S}$ and $\mathtt{I}_\mathtt{T}$, and outputs a softmaxed 2D tensor. A correspondence map $\mathtt{C}_\mathtt{T}$ may not have the same number of lines and columns than $\mathtt{I}_\mathtt{T}$ especially when the goal is to outpaint a correspondence. Thus, in the general case, to transform a 2D point from the image plane of $\mathtt{I}_\mathtt{T}$ to the correspondence plane of $\mathtt{C}_\mathtt{T}$, we will need another calibration matrix $\mathtt{K}_\mathtt{C}$. Let us highlight that this likelihood is obtained using the visual information of $\mathtt{I}_\mathtt{S}$ and $\mathtt{I}_\mathtt{T}$ only.

The NRE provides us with a framework to learn to identify, inpaint or outpaint the correspondent of $\mathbf{p}_\mathtt{S}$ in $\mathtt{I}_\mathtt{T}$ in a unified manner since Eq. (5.1) is differentiable w.r.t. $\mathtt{C}_\mathtt{T}$ and there is no assumption regarding covisibility. The main difficulty to overcome is the

definition of a network architecture able to output a consistent $\mathtt{C}_\mathtt{T}$ being given only $\mathbf{p}_\mathtt{S}$, $\mathtt{I}_\mathtt{S}$ and $\mathtt{I}_\mathtt{T}$ as inputs, i.e. the network must figure out whether the correspondent of $\mathbf{p}_\mathtt{S}$ in $\mathtt{I}_\mathtt{T}$ can be identified or has to be inpainted or outpainted.

5.3.3 Network architecture

The analysis from Section 5.3.1 and the use of the NRE as a loss (Section 5.3.2) call for:

- a non-siamese architecture to be able to link the information from $\mathtt{I}_\mathtt{S}$ with the information from $\mathtt{I}_\mathtt{T}$ to *outpaint* or *inpaint* the correspondent if needed;

- an architecture that outputs a matching score for all the possible locations in $\mathtt{I}_\mathtt{T}$ as well as locations beyond the field of view of $\mathtt{I}_\mathtt{T}$ as the network could decide to identify, inpaint or outpaint a correspondent at these locations.

To fullfil these requirements, we propose the following: Our network takes as input $\mathtt{I}_\mathtt{S}$ and $\mathtt{I}_\mathtt{T}$ as well as a set of keypoints $\{\mathbf{p}_{\mathtt{S},n}\}_{n=1...N}$ in the source image plane of $\mathtt{I}_\mathtt{S}$ A siamese CNN backbone is applied to $\mathtt{I}_\mathtt{S}$ and $\mathtt{I}_\mathtt{T}$ to produce compact dense local descriptor maps $\mathtt{H}_\mathtt{S}$ and $\mathtt{H}_\mathtt{T}$. In order to be able to *outpaint* correspondents in the target image plane, we pad $\mathtt{H}_\mathtt{T}$ with a learnable fixed vector $\boldsymbol{\lambda}$. This padding step allows to *initialize* descriptors at locations outside the field of view of $\mathtt{I}_\mathtt{T}$. The amount of padding is a hyper-parameter. We note γ the relative output-to-input correspondence map resolution ratio.

The dense descriptor maps $\mathtt{H}_\mathtt{S}$ and $\mathtt{H}_{\mathtt{T},\mathrm{pad}}$, and the keypoints $\{\mathbf{p}_{\mathtt{S},n}\}_{n=1...N}$ are then used as inputs of a cross-attention-based backbone $\mathcal{F}$ with positional encoding. This part of the network outputs a feature vector $\mathbf{h}_{\mathtt{S},n}$ for each keypoint $\mathbf{p}_{\mathtt{S},n}$ and dense feature vectors $\mathtt{D}_{\mathtt{T},\mathrm{pad}}$ of the size of $\mathtt{H}_{\mathtt{T},\mathrm{pad}}$. This cross-attention-based backbone allows the local descriptors $\mathtt{H}_\mathtt{S}$ and $\mathtt{H}_{\mathtt{T},\mathrm{pad}}$ to *communicate* with each other. Thus, during training, the network will be able to leverage this ability to communicate to learn to perform *geometric reasoning* and produce peaked *inpainted* and *outpainted* correspondence maps.

The correspondence map $\mathtt{C}_{\mathtt{T},n}$ of $\mathbf{p}_{\mathtt{S},n}$ in the image plane of $\mathtt{I}_\mathtt{T}$ is computed by applying a 1×1 convolution to $\mathtt{D}_{\mathtt{T},\mathrm{pad}}$ using $\mathbf{h}_{\mathtt{S},n}$ as filter, followed by a 2D softmax. The correspondence map $\mathtt{C}_{\mathtt{T},n}$ of $\mathbf{p}_{\mathtt{S},n}$ in the image plane of $\mathtt{I}_\mathtt{T}$ is computed by applying a 1x1 convolution to $\mathtt{D}_{\mathtt{T},\mathrm{pad}}$ using $\mathbf{h}_{\mathtt{S},n}$ as filter, followed by a 2D softmax.

An overview of our architecture, that we call NeurHal, is presented in Fig. 5.2. In practice, in order to keep the required amount of memory and the computational time reasonably low, the correspondence maps $\{\mathtt{C}_{\mathtt{T},n}\}_{n=1...N}$ have a low resolution, i.e. for a target image of size 640×480, we use a CNN with an effective stride of $s = 8$ and consequently the resulting correspondence maps (with $\gamma = 50\%$) are of size 160×120. Producing low resolution correspondence maps prevents NeurHal from predicting accurate

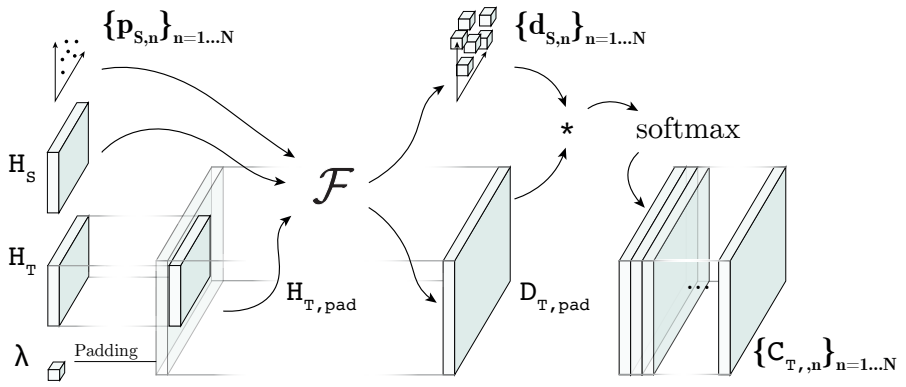


Figure 5.2: **Overview of NeurHal:** See text for details.

correspondences. But as we show in the experiments, this low resolution is sufficient to hallucinate correspondences and have an *affirmative answer* to both questions: (i) can we derive a network architecture able to learn to hallucinate correspondences? and (ii) is correspondence hallucination beneficial for absolute pose estimation? Thus, we leave the question of the accuracy of hallucinated correspondences for future research.

Additional details concerning the architecture are provided in the appendix.

5.3.4 Training-time

Given a pair of partially overlapping images $(\mathtt{I}_\mathtt{S}, \mathtt{I}_\mathtt{T})$, a set of keypoints with ground truth depths $\{\mathbf{p}_{\mathtt{S},n}, d_{\mathtt{S},n}\}_{n=1\ldots N}$ as well as the ground truth relative camera pose $(\mathtt{R}_\mathtt{TS}, \mathbf{t}_\mathtt{TS})$, the corresponding sum of NRE terms (Eq. 5.1) can be minimized w.r.t. the parameters of the network that produces the correspondence maps. Thus, we train our network using stochastic gradient descent and early stopping by providing pairs of overlapping images along with the aforementioned ground truth information. Let us also highlight that there is no distinction in the training process between the identifying, inpainting and outpainting tasks since the only thing our network outputs are correspondence maps. Moreover there is no need for labeling keypoints with ground truth labels such as "identify/visible", "inpaint/occluded" or "outpaint/outside the field of view". Additional information concerning the training are provided in the appendix.

5.3.5 Test-time

At test-time, our network only requires a pair of partially overlapping images $(\mathtt{I}_\mathtt{S}, \mathtt{I}_\mathtt{T})$ as well as keypoints $\{\mathbf{p}_{\mathtt{S},n}\}_{n=1\ldots N}$ in $\mathtt{I}_\mathtt{S}$, and outputs a correspondence map $\mathtt{C}_{\mathtt{T},n}$ in the image plane of $\mathtt{I}_\mathtt{T}$ for each keypoint, regardless of its correspondent being visible, occluded or outside the field of view.

5.4 Experiments

In these experiments, we seek to answer two questions: 1) "Is the proposed NeurHal approach presented in Section 5.3 indeed capable of hallucinating correspondences?" and 2) "In the context of camera pose estimation, does the ability to hallucinate correspondences bring further robustness?".

5.4.1 Evaluation of the ability to hallucinate correspondences

We evaluate the ability of our network to hallucinate correspondences on four datasets: the indoor datasets ScanNet (Dai et al., 2017) and NYU (Nathan Silberman & Fergus, 2012), and the outdoor datasets MegaDepth (Li & Snavely, 2018) and ETH-3D (Schöps et al., 2017).

For the indoor setting (outdoor setting, respectively), we train NeurHal on ScanNet (Megadepth, respectively) on the training scenes as described in Sec. 5.3.4, and evaluate it on the *disjoint* set of validation scenes. Thus, all the qualitative and quantitative results presented in this section cannot be ascribed to scene memorization.

For each dataset, we run predictions over $2,500$ source and target image pairs sampled from the test set, with overlaps between 2% and 80%. For every image pair, we also feed as input to NeurHal keypoints in the source image. These keypoints have known ground truth correspondents in the target image and labels (visible, occluded, outside the field of view) that we use to evaluate the ability of our network to hallucinate correspondences. For more details on the settings of our experiment see Sec. C.3.2 of the appendix. For this experiment, we use $\gamma = 50\%$.

We report in Fig. 5.3 two histograms computed over more than one million keypoints for each task we seek to validate: identification, inpainting, and outpainting. The first histogram Fig. 5.3 (left) is obtained by evaluating for each correspondence map the NRE cost (Eq. 5.1) at the ground truth correspondent's location. In order to draw conclusions, we also report the negative log-likelihood of a uniform correspondence map ($\ln |\Omega_{\mathtt{C_T}}|$). We find that for each task and for both datasets, the predicted probability mass lies significantly below $\ln |\Omega_{\mathtt{C_T}}|$, which demonstrates NeurHal's ability to perform identification, inpainting and outpainting. On ScanNet, we also observe that identification is a simpler task than outpainting while inpainting is the hardest task: On average, the NRE cost of inpainted correspondents is higher than the average NRE cost of outpainted correspondents, which indicates the predicted correspondence maps are less peaked for inpainting than they are for outpainting. This corroborates what we empirically observed on qualitative results in Fig. 5.1, and supports our analysis in Sec. 5.3.1. On Megadepth, outpainting and inpainting histograms have a similar shape which does not reflect the previous statement, but we believe this is due to the fact that inpainting labels are noisy for this

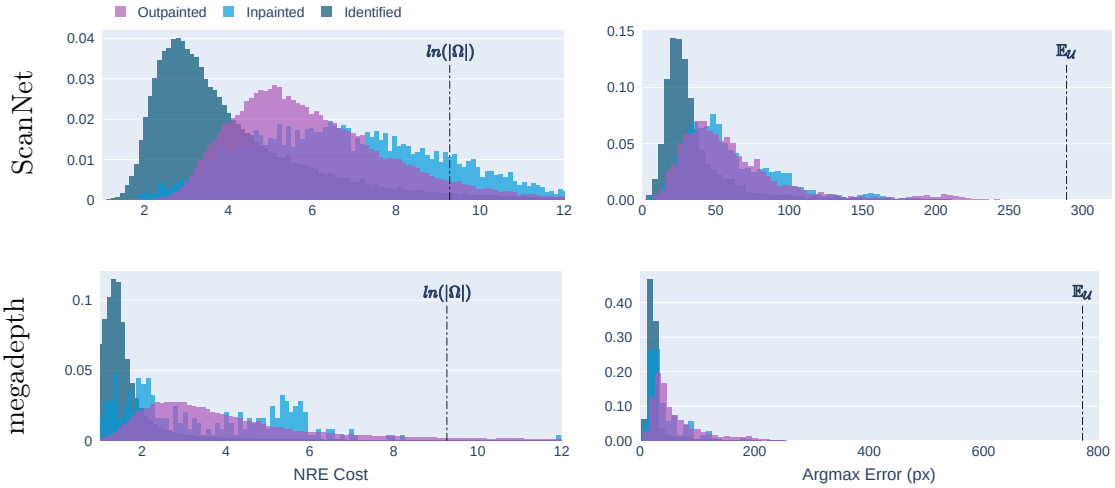


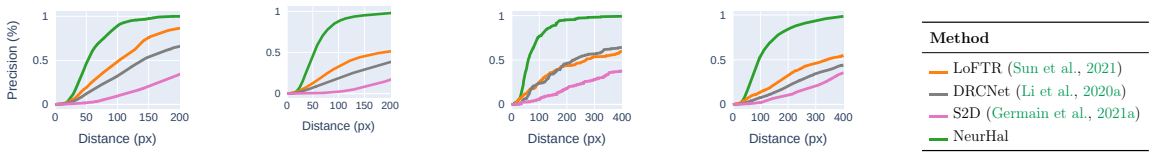
Figure 5.3: **Evaluation of the ability of our network to hallucinate correspondences on the test sets of ScanNet (Dai et al., 2017) and MegaDepth (Li & Snavely, 2018).** (left) Histograms of the NRE (see Eq. 5.1) for each task (identifying, outpainting, inpainting), computed on correspondence maps produced by NeurHal. The value $\ln|\Omega_{\mathtt{C_T}}|$ is the NRE of a uniform correspondence map. (right) Histograms of the errors between the argmax (mode) of a correspondence map and the ground truth correspondent's location, for each task. The value $\mathbb{E}_{\mathcal{U}}$ is the average error of a random prediction.

dataset, as explained in Sec. C.3.2 of the appendix.

On the right histogram of Fig. 5.3, we report the distribution of the distance between the argmax of a correspondence map and the ground truth correspondent's location. We also report the average error of a random prediction. We find the histogram mass lies significantly to the left of the random prediction average error, indicating our model is able to place modes correctly in the correspondence maps, regardless of the task at hand. On ScanNet, we observe that the inpainting and outpainting histograms are very similar, indicating the predicted argmax is equally good for both tasks. As mentioned above, the correspondence maps produced by NeurHal have a low resolution (see Sec. 5.3.3) which explains why the "argmax error" is not closer to zero pixel.

In Fig. 5.4, we compare the hallucination performances of NeurHal against state-of-the-art local feature matching methods. We call S2D the NRE features from the previous chapter. Since all these local feature matching methods were designed and trained on pairs of images with significant overlap to perform only identification, they obtain poor inpainting results. Concerning the outpainting task, these methods seek to find a correspondent within the image boundaries, consequently they cannot outpaint correspondences and obtain very poor results.

In Fig. 5.5 we show several qualitative inpainting/outpainting results on ScanNet and



(a) Inpainting - S (b) Outpainting - S (c) Inpainting - M (d) Outpainting - M

Figure 5.4: **Ability to hallucinate - comparison against state-of-the-art local feature matching methods on ScanNet (S) and Megadepth (M).** For each method, we report the percentage of keypoint's correspondents whose distance w.r.t. the ground truth location is lower than x pixels, as a function of x, for (a-c) the inpainting task and (b-d) the outpainting task.

MegaDepth datasets. In the appendix, we also report qualitative results obtained on the NYU Depth dataset (Fig. 5.13) and on the ETH-3D dataset (Fig. 5.12).

These results allow us to conclude that NeurHal is able to hallucinate correspondences with a strong generalization capacity. Technical details regarding the evaluation protocol in Sec. C.3.3.

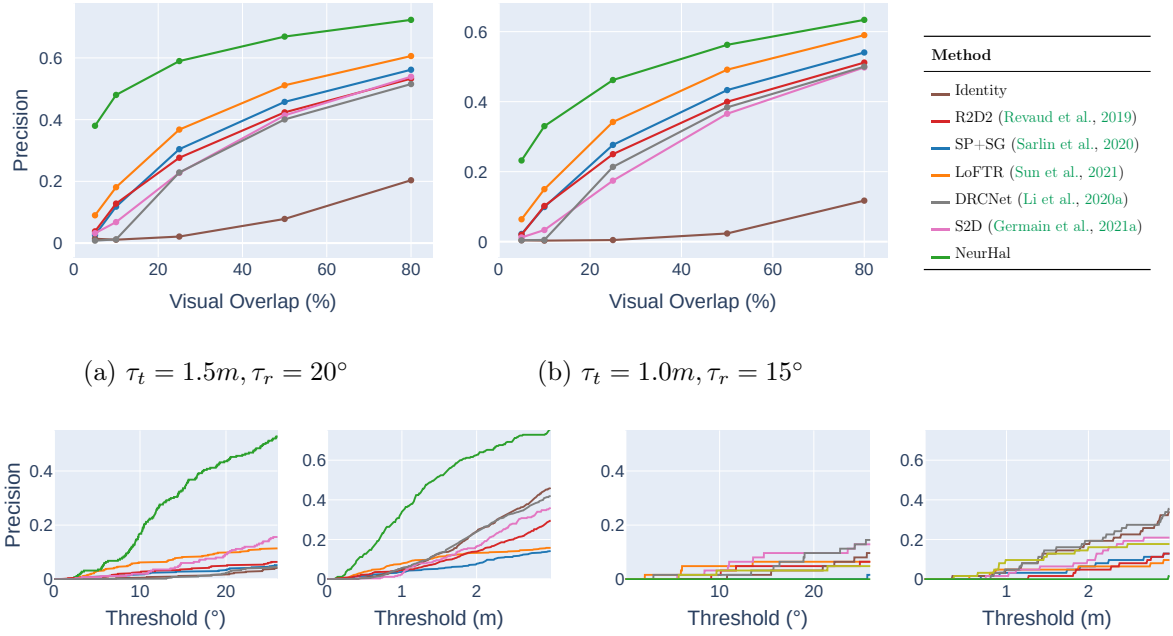
5.4.2 Application to camera pose estimation

In the previous experiment, we showed that our network is able to hallucinate correspondences. We now evaluate whether this ability helps improving the robustness of an absolute camera pose estimator. We run this evaluation on the test set of ScanNet over 2,500 source and target image pairs captured in scenes that were not used at training time. For each source/target image pair, we employ NeurHal to produce correspondence maps. As in the previous experiment, we use $\gamma = 50\%$. Given these correspondence maps and the depth map of the source image, we estimate the absolute camera pose between the target image and the source image using the NRE from the previous chapter.

In Fig. 5.7, we show the results of an ablation study conducted on ScanNet. In this study, we focus on the robustness of the camera pose estimate, i.e. we consider a pose is "correct" if the rotation error is lower than 20 degrees and the translation error is below 1.5 meters. We find that training our network to perform the three tasks (identification, inpainting, and outpainting) produces the best results. In particular, we find that adding outpainting plays a critical role in improving localization of low-overlap image pairs. We also find that learning to inpaint does not bring much improvement to the absolute camera pose estimation.

In Fig. 5.6, we compare the results of our approach against state-of-the-art local feature matching methods. Fig. 5.6 (a) & (b) show that NeurHal is able to estimate the camera pose correctly significantly more often than any other method. Fig. 5.6 (c) shows that NeurHal is much more robust than state-of-the-art local feature matching methods for

Figure 5.5: **Qualitative inpainting/outpainting results.** To illustrate the ability of NeurHal to perform visual correspondence hallucination, we display correspondence maps output by NeurHal on validation image pairs: (top row) outpainting examples, (bottom row) inpainting examples. In the source image, the red dot is a keypoint. In the target image and in the (negative log) correspondence map, the red dot represents the ground truth keypoint's correspondent. The dashed rectangles represent the borders of the target images.



(a) $\tau_t = 1.5m, \tau_r = 20°$ (b) $\tau_t = 1.0m, \tau_r = 15°$

(c) 2%-5% visual overlap (d) Top-25% worst NeurHal predictions

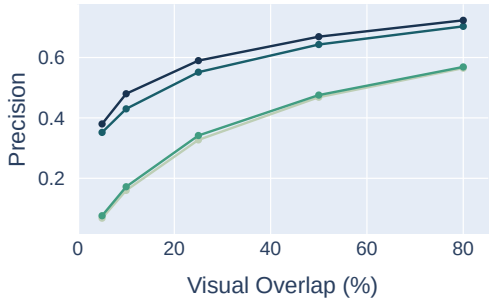
Figure 5.6: **Visual correspondence hallucination for camera pose estimation.** We compare the performance of NeurHal against state-of-the-art local feature matching methods on ScanNet (Dai et al., 2017). The "identity" method consists in systematically predicting the identity pose. In (a) & (b) we report the percentage of camera poses being correctly estimated for pairs of images that have an overlap between 2% and x%, as a function of x, for two rotation and translation error thresholds. In (c) we focus on image pairs with less than 5% of visual overlap. In (d) we focus on the 25% of images pairs where NeurHal has the worst camera pose estimates. See discussion at the end of Sec. 5.4.2.

pairs of images with a low overlap. Fig. 5.6 (d) highlights the fact that when NeurHal fails to correctly estimate the camera pose, all the competitors also fail since all the methods perform similarly to the "identity" method, i.e. the method that consists in systematically predicting the identity pose.

5.4.3 Impact of learning to inpaint and outpaint

To supplement our previous experiments, we now aim at evaluating the impact of learning to inpaint and outpaint specifically. To do so, we isolate keypoints with the *identified*, *inpainted* and *outpainted* labels in our ScanNet (Dai et al., 2017) evaluation set.

In Fig. 5.8, we show the results of an ablation study on NeurHal's training setup. We report for the identification, inpainting and outpainting tasks two sets of cumulative histograms: 1) the NRE costs at ground truth keypoint correspondents' locations, and 2)



Method	Training correspondences		
	Identified	Inpainted	Outpainted
	✓		
	✓	✓	
	✓		✓
	✓	✓	✓

Figure 5.7: **Ablation study - Impact of learning to hallucinate for absolute camera pose estimation.** We compare the influence of adding inpainting and outpainting ($\gamma = 50\%$) tasks when training NeurHal. We report the percentage of camera poses being correctly estimated for pairs of images that have an overlap between 2% and x%, as a function of x, on ScanNet (Dai et al., 2017), with thresholds for translation error and rotation error of $\tau_t = 1.5m$ and $\tau_r = 20.0°$. Learning to hallucinate correspondences significantly improves the percentage of correctly estimated camera poses, with the outpainting learning task bringing most of the improvement.

distances between the argmax of the correspondence map and the ground truth location. On NRE cost cumulative histograms, we also report the results from the uniform distribution, for models trained both with and without outpainting ($\gamma = 0\%$ and $\gamma = 50\%$ respectively).

For the identification task (Fig. 5.8 (a)) we find that all methods yield a consistent performance. The left figure reveals that NeurHal predictions are significantly above the uniform distribution, indicating peaky maps and thus confident predictions. The right figure shows that the distance of the argmax location w.r.t. the ground truth is also robust (NeurHal predicts at 1/8th of the original resolution but the histogram is computed at full resolution).

For the inpainting task (Fig. 5.8 (b)) we can draw similar conclusions. We find however that correspondence maps are overall less peaky and closer to their respective uniform distribution, which indicates that predictions are less confident. We also find that even though it was not trained to inpaint, the identification baseline is surprisingly able to inpaint correspondences as its performance is not far from the identification+inpainting model.

Lastly for the outpainting task (Fig. 5.8(c)), we find that learning to outpaint gives a significant boost in performance on both the NRE distribution and correspondents locations. We also find that jointly learning to inpaint and outpaint is beneficial to the quality of the outpainted cost maps, which implies that both objectives are complementary.

5.4.4 Influence of the pose estimator: NRE vs. RE

The NRE provides a pose estimation framework which leverages dense keypoint matching uncertainties to predict more accurate and robust camera poses. Compared to the standard pose estimator presented in (Chum et al., 2003) which relies on sparse 2D-to-3D correspondences, the NRE preserves rich information in the form of dense loss maps that is particularly suited for ambiguous matches. For the problem of correspondence hallucination we find the loss maps of both outpainted and inpainted correspondences are usually unimodal but quite diffuse, and are thus particularly suited for this pose estimator.

To study the influence of the pose estimator, we report in Fig. 5.9 the performance of NeurHal + NRE (Germain et al., 2021a) vs. NeurHal + RE (Chum et al., 2003). To estimate the camera pose using the method presented in (Chum et al., 2003), we simply take the argmax of each correspondence map and treat it as a sparse 2D correspondent in the query image. We also include the performance of NeurHal when trained without visual correspondence hallucination (i.e. trained using only identified ground truth correspondences.)

We find that the two methods trained without hallucination have poor performances for very low-overlap image pairs which underlines the importance of correspondence hallucination in such cases.

Concerning NeurHal trained with hallucination and using the pose estimator (Chum et al., 2003), taking the argmax of a very coarse correspondence map prevents the pose estimator from achieving good results.

NeurHal trained with hallucination and coupled with the NRE-based pose estimator achieves the best results which shows that to obtain robust absolute camera estimates it is important to *combine* the ability to hallucinate correspondences of NeurHal with the NRE-based pose estimator.

5.4.5 Impact of the value of γ

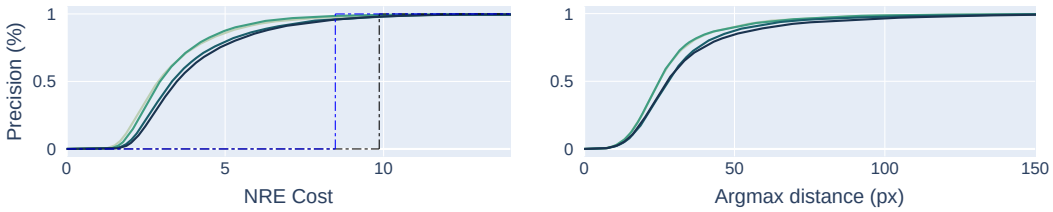
We report in Fig. 5.10 the absolute camera pose estimation performance for varying values of γ. We compute the percentage of camera poses being correctly estimated for ScanNet (Dai et al., 2017) test images pairs that have an overlap between 2% and x% (as a function of x) for a translation threshold of $1.5m$ and a rotation threshold of $20.0°$.

We find that using only a small percentage of outpainting such as $\gamma = 10\%$ does not improve the performance which is most likely due to the small amount of added training keypoints. For higher γ values however significant gains are visible, especially at small visual overlaps. This experiment demonstrates the benefit of learning to outpaint correspondences beyond image borders, and broaden the extent of usable source keypoints to perform camera pose estimation.

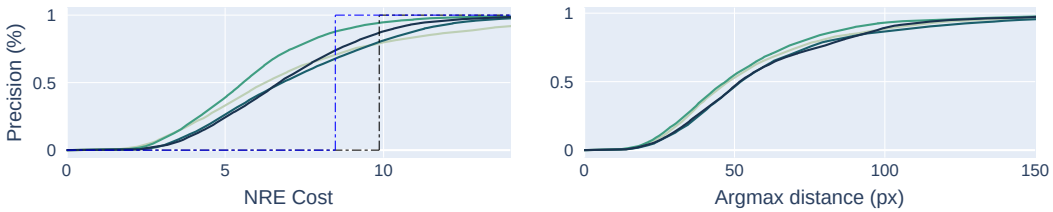
We report in Fig. 5.11 the camera field-of-view as a function of the padding parameter. We find that $\gamma = 50\%$ provides 130° and 71° of field-of-view on average on ScanNet and Megadepth respectively, which is significantly wider than $\gamma = 0\%$.

5.4.6 Generalization to new datasets

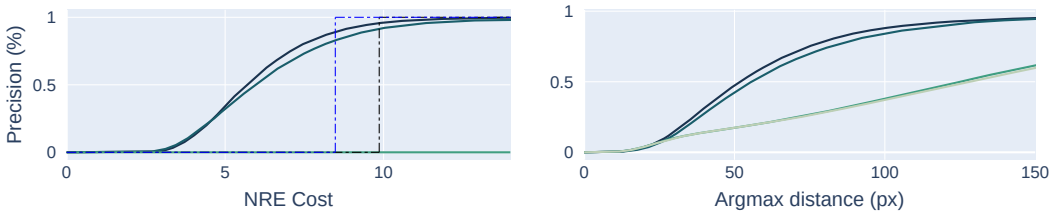
So far we have demonstrated the ability of NeurHal to hallucinate correspondences on unseen validation scenes from both ScanNet (Dai et al., 2017) and Megadepth (Li & Snavely, 2018). In order to further demonstrate the generalization capacity of NeurHal, we report qualitative results obtained on the NYU Depth Dataset (Nathan Silberman & Fergus, 2012) in Fig. 5.13 and on the ETH-3D (Schöps et al., 2017) dataset in Fig. 5.12. We use the set of indoor weights for NYU (i.e. NeurHal trained on ScanNet) and outdoor weights for ETH-3D (i.e. NeurHal trained on MegaDepth). We report the overlayed and upsampled coarse truncated loss map computed following Germain et al. (2021a) on low-overlap image pairs. We find that NeurHal is able to robustly outpaint correspondences despite little visual overlaps and strong relative camera motions. These visuals demonstrate the strong generalization ability of NeurHal.



(a) Identified



(b) Inpainted

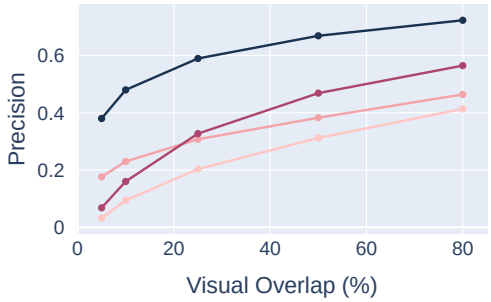


(c) Outpainted

Method	Training correspondences		
	Identified	Inpainted	Outpainted
	✓		
	✓	✓	
	✓		✓
	✓	✓	✓

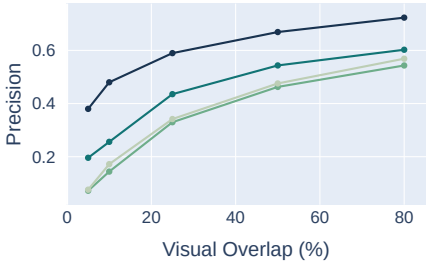
$ln(|\Omega|)_{\gamma=0\%}$ $ln(|\Omega|)_{\gamma=50\%}$

Figure 5.8: **Ability to hallucinate - Ablation study on ScanNet.** We compare the influence of adding inpainting and outpainting when training NeurHal. **(left column)** We report the percentage of keypoint's correspondents whose NRE cost is lower than x, as a function of x, for (a) identified (b) inpainted and (c) outpainted keypoints. **(right column)** We report the percentage of keypoint's correspondents whose distance w.r.t. the ground truth is lower than x pixels, as a function of x, for the same categories.



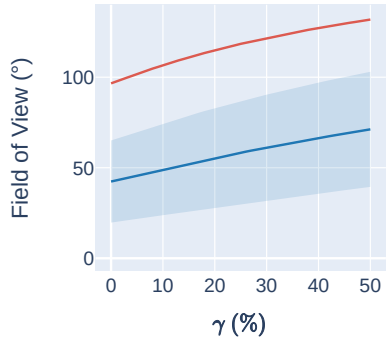
Method	Hallucination	Pose estimator
		(Chum et al., 2003)
	✓	(Chum et al., 2003)
		(Germain et al., 2021a)
	✓	(Germain et al., 2021a)

Figure 5.9: **Influence of the pose estimator: NRE (Germain et al., 2021a) vs. RE (Chum et al., 2003):** To study the influence of using the NRE-based pose estimator compared to using the pose estimator from (Chum et al., 2003), we report the performance of NeurHal with both estimators. We also include, for both estimators, the performance of NeurHal trained with identified correspondences only (i.e. without hallucination). We report the percentage of camera poses being correctly estimated for pairs of ScanNet (Dai et al., 2017) images that have an overlap between 2% and x% (as a function of x).



Method	γ
	$\gamma = 50\%$
	$\gamma = 25\%$
	$\gamma = 10\%$
	$\gamma = 0\%$

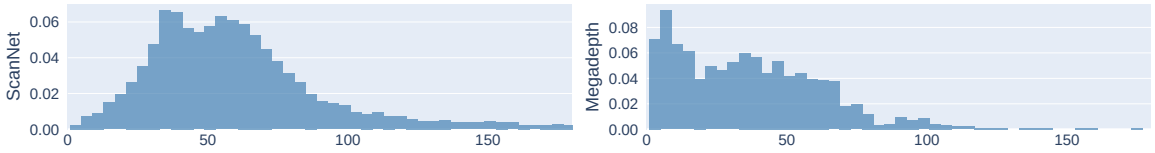
Figure 5.10: **Impact of the value of γ:** For increasing values of γ, we report the percentage of camera poses being correctly estimated for pairs of ScanNet images that have an overlap between 2% and x% (as a function of x), for $\tau_t = 1.5m$ and $\tau_r = 20.0°$. We find that a small value of $\gamma = 10\%$ yields no benefit and even damages performance, while values of $\gamma = 25\%$ and $\gamma = 50\%$ bring significant improvements, especially at small visual overlaps.



(a) Field-of-view w.r.t. γ

Dataset	$\lvert\Delta r_x\rvert$	$\lvert\Delta r_y\rvert$	$\lvert\Delta r_z\rvert$	$\lvert\Delta\theta\rvert$	$\lvert\Delta f\rvert$
ScanNet	29.21°	38.72°	25.68°	55.20°	0.00mm
Megadepth	4.73°	6.91°	1.64°	20.25°	376.69mm

(b) Relative viewpoint statistics



(c) Histogram of absolute relative angle norm $\lvert\Delta\theta\rvert$

Figure 5.11: **Field-of-view as a function of γ and relative viewpoint statistics:** We report in (a) the average camera field-of-view as a function of γ on ScanNet (Dai et al., 2017) and Megadepth (Li & Snavely, 2018) images. We find that $\gamma = 50\%$ enables a significant amount of additional visual content to reproject within the image boundaries. We report in (b) the median absolute difference in rotation along the x, y and z axis, norm of the relative rotation, along with the difference in focal length on low-overlap image pairs for ScanNet (Dai et al., 2017) and Megadepth (Li & Snavely, 2018). We report in (c) the histogram of absolute relative angle norm on both datasets. We find ScanNet image pairs exhibit strong relative angular motion while Megadepth image pairs display predominantly zoom-ins and zoom-outs.

Figure 5.12: **Qualitative results on the ETH3D dataset:** We evaluate NeurHal on outdoor image pairs from the ETH-3D (Schöps et al., 2017) dataset and find it is able to outpaint correspondences despite low visual overlaps. We report pairs of source and target images and overlay the upsampled coarse loss map corresponding to the source detection (in red) on the target image.

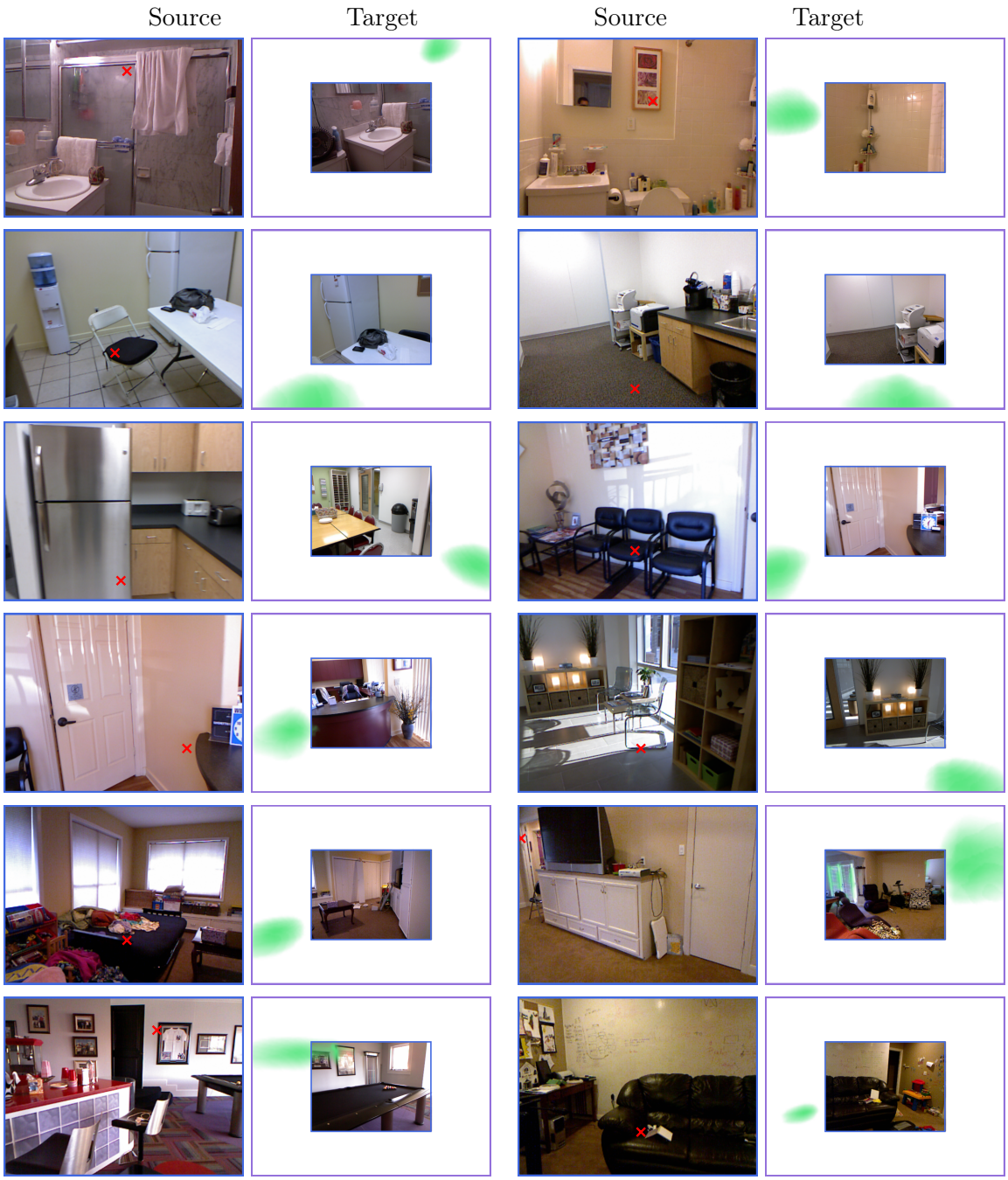


Figure 5.13: **Qualitative results on the NYU dataset:** We evaluate NeurHal on indoor images from the NYU (Nathan Silberman & Fergus, 2012) dataset and find it is able to outpaint correspondences despite low visual overlaps. We report pairs of source and target images and overlay the upsampled coarse loss map corresponding to the source detection (in red) on the target image.

5.5 Limitations

We identified the following limitations for our approach:

1. The previous experiments showed that NeurHal is able to inpaint correspondences but the inpainted correspondence maps are much less peaked compared to the outpainted correspondence maps. This is likely due to the fact that inpainting correspondences is much more difficult than outpainting correspondences (see Sec 5.3.1).

2. The proposed architecture outputs low resolution correspondence maps (see Sec. 5.3.3), e.g. 160×120 for input images of size 640×480 and an amount of padding $\gamma = 50\%$. This is essentially due to the quadratic complexity of attention layers we use (see Sec. C.3.1 of the appendix).

3. Our approach is able to outpaint correspondences but our correspondence maps have a finite size. Thus, in the case where a keypoint's correspondent falls outside the correspondence map, the resulting correspondence map would be erroneous.

We believe these three limitations are interesting future research directions.

5.6 Conclusion

To the best of our knowledge, this paper is the first attempt to learn to inpaint and outpaint correspondences. We proposed an analysis of this novel learning task, which has guided us towards employing an appropriate loss function and designing the architecture of our network. We experimentally demonstrated that our network is indeed able to inpaint and outpaint correspondences on pairs of images captured in scenes that were not seen at training-time, in both indoor (ScanNet) and outdoor (Megadepth) settings. We also tested our network on other datasets (ETH3D and NYU) and discovered that our model has strong generalization ability. We then tried to experimentally illustrate that hallucinating correspondences is not just a fundamental AI problem but is also interesting from a practical point of view. We applied our network to an absolute camera pose estimation problem and found that hallucinating correspondences, especially outpainting correspondences, allowed to significantly outperform the state-of-the-art feature matching methods in terms of robustness of the resulting pose estimate. Beyond this absolute pose estimation application, this work points to new research directions such as integrating correspondence hallucination into Structure-from-Motion pipelines to make them more robust when few images are available.

Chapter 6

Conclusion

In this chapter we will give a brief summary of the contributions presented in this manuscript, as well as describe several paths for future research building on our methods. We will review our contributions in Section 6.1, discuss the potential impact of our work in Section 6.2 and lastly present paths for future work Section 6.3.

6.1 Contributions

In this manuscript, we made several contributions:

- In Chapter 3, we introduced the sparse-to-dense matching paradigm, which consists in performing keypoint detection in one image and to exhaustively search for its correspondent in the other. We derived a weakly and a strongly supervised training procedure to learn robust and accurate correlation maps. We found this paradigm to yield significant improvement in image matching, with a direct impact on a downstream task such as structure-based visual localization.
- In Chapter 4, we proposed the Neural Reprojection Error. The NRE builds on the sparse-to-dense paradigm to act as a learning-based camera pose estimator. Our formulation of the NRE makes it possible to easily backpropagate through image features, which results in a simple deep learning-based problem. In our experiments, we demonstrated our proposed estimator significantly outperforms state-of-the-art RE-based pose estimators, while remaining computationally and memory efficient.
- In Chapter 5, we turned to the problem of visual correspondence hallucination. We introduced NeurHal, a deep learning architecture tailored to leverage both covisible and non-covisible training data. In our experiments, we demonstrated that (*i*) NeurHal is indeed capable of leveraging that data to hallucinate correspondences and (*ii*) NeurHal can be leveraged to increase the robustness of NRE-based camera pose estimation on very low-overlap image pairs.

6.2 Impact

The contributions presented in this thesis have a direct impact on both keypoint matching and visual localization methods.

The initial idea of sparse-to-dense matching breaks with the very popular sparse-to-sparse paradigm, and paves the way for research in asymmetric dense correspondence search across images. Both the NRE and NeurHal reuse this framework by relying on dense correspondence maps which describe keypoint matching likelihoods. The work

of (Zhou et al., 2020a) reuse the idea of densely searching for correspondences in local patches around prior correspondences obtained in a sparse-to-sparse fashion. Lastly the recent work of (Lindenberger et al., 2021) reuse S2DNet to perform refinement of SfM models through multi-view featuremetric error optimization.

The NRE is a novel way of performing learning-based camera pose estimation, and our experiments exhibit great potential for future research in that area. Compared to RE-based pose estimation, we can see that resorting to a dense modeling of local matching uncertainties is key to make a leap in performance compared to previous research. In contrast to learning-based featuremetric optimization methods such as (Sarlin et al., 2021; Stumberg et al., 2020; von Stumberg et al., 2020), the NRE avoids the cumbersome unrolling of direct optimizers steps, and derives a much simpler training procedure based on pixel-wise classification of correspondences.

Lastly NeurHal tackles the problem of visual correspondence hallucination, which was up to now a virgin territory. We believe that while this initial work on the problem of correspondence hallucination displays promising results, there is room for improvement. We also believe that NeurHal can have a strong impact in improving the performance of relocalization algorithms in low-overlap cases, which are often very difficult even for a human. With the recent breakthrough of Transformer architectures we hope to see improved versions of NeurHal that can perform genuine geometric reasoning.

6.3 Future Work

In this last section, we propose several paths for future research building upon our contributions.

6.3.1 Multi-view sparse-to-dense matching

The formulation of sparse-to-dense matching in this manuscript only considers pairs of images: usually a query image and a reference image (e.g. a nearest neighbour from the reference set). While in the case of visual localization we can easily aggregate correspondence maps coming from multiple reference images w.r.t. the query image, we are still in an N-to-1 matching scenario. In other computer vision problems such as Structure-from-Motion however, we often need to consider N-to-M matching problems. In such cases any image can be both considered to be a *source* and a *target*, which drives the number of possible asymmetric matches exponentially. Coping with this high computational cost and deriving a method that leverages dense correspondence maps bi-directionally remains

an open problem.

6.3.2 Correspondence-free localization

The NRE-based pose estimator leverages dense loss maps to both estimate an initial camera pose (through an P3P solver inside an MSAC loop), followed by a refinement step (using Graduated Non-Convexity). During the P3P stage however we sample triplets of correspondences at correspondence maps argmax locations, which thus implies not only reducing the sampled maps to a single value but also defining explicit 2D-to-3D correspondences, much like our previous work in S2DHM and S2DNet. In contrast once this pose initialization step is done, the GNC-based camera pose refinement stage does not require defining explicit 2D-to-3D keypoint correspondences. An interesting path for future work could be to aim at designing a feature extractor that ensures convergence for GNC optimizations that start from much large values of σ_{max}. In other words by taking very large Gaussian kernels and initializing the camera at the reference camera pose, the GNC would no longer be a simple *refinement* process but rather a pose *estimation* method. The advantage of such an approach would be twofold: (*i*) we would not have to rely on any hyperparameter, such as the number of MSAC iterations and (*ii*) the efficiency of the GNC algorithm (which relies on sparse loss maps) would make for a very fast pose estimator. Despite our attempts at running a purely GNC-based pose estimator, we found that our loss maps always ended up guiding the pose to local minima, giving much worse performance than when using the MSAC-based pose initialization.

6.3.3 Geometric Reasoning

In NeurHal we derived an attempt at performing *implicit* geometric reasoning, by forcing our model to make use of both covisible and non-covisible training data. We hypothesize that our model has thus learned internal representation of depth and geometry, in order to perform the complex task of inpainting and outpainting. Verifying that hypothesis practically however seems very hard, as diving inside the neural network provides little explanation to its reasoning and our failure cases exhibit a lack of generalization to some extreme cases. Designing a method that performs a provable geometric reasoning remains an open problem that could however surpass any existing image matching method, especially in very challenging low-overlap cases.

Appendix A

Additional results on sparse-to-dense matching

In this chapter, we present additional qualitative results and details for our sparse-to-dense matching methods S2DHM and S2DNet.

A.1 Additional S2DHM qualitative results

In this section, we show additional qualitative results. In Fig. A.1, we display examples of local correspondences obtained using sparse-to-sparse SuperPoint detection and descriptors against S2DHM. Fig. A.2 shows correlation maps obtained for a sparse 3D descriptor, on both datasets. Lastly, Fig. A.3 displays inlier correspondences after running PnP + RANSAC, using our sparse-to-dense approach.

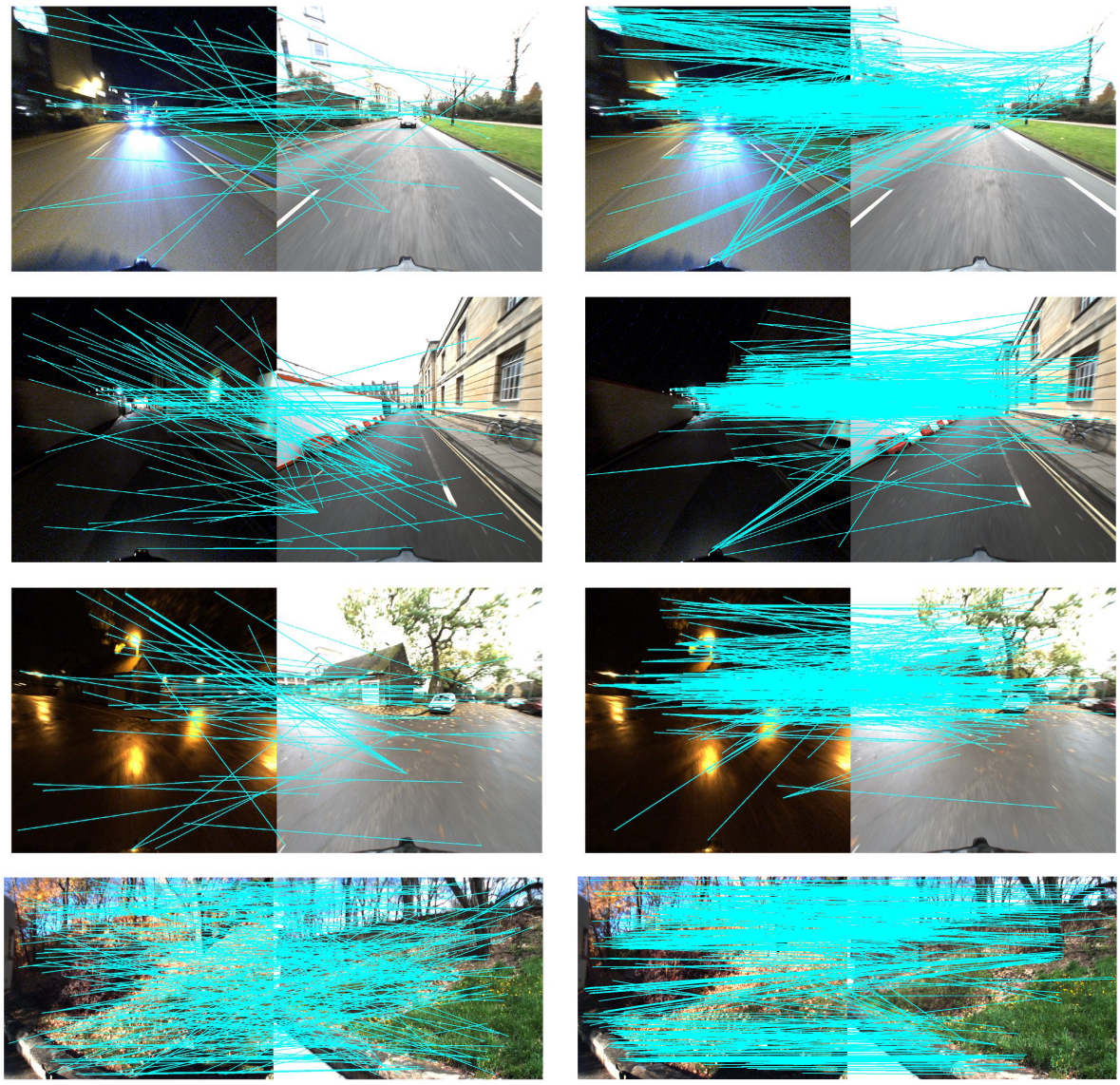


Figure A.1: **Sparse-to-sparse and sparse-to-dense local feature matching.** Left column: Sparse-to-sparse feature matching obtained using SuperPoint detections and SuperPoint descriptors. Right column: Sparse-to-dense feature matching using hypercolumns. Both correspondences are displayed after applying a ratio test. Our approach tends to provide a lot more matches, which are overall more robust thanks to the hypercolumn descriptors.

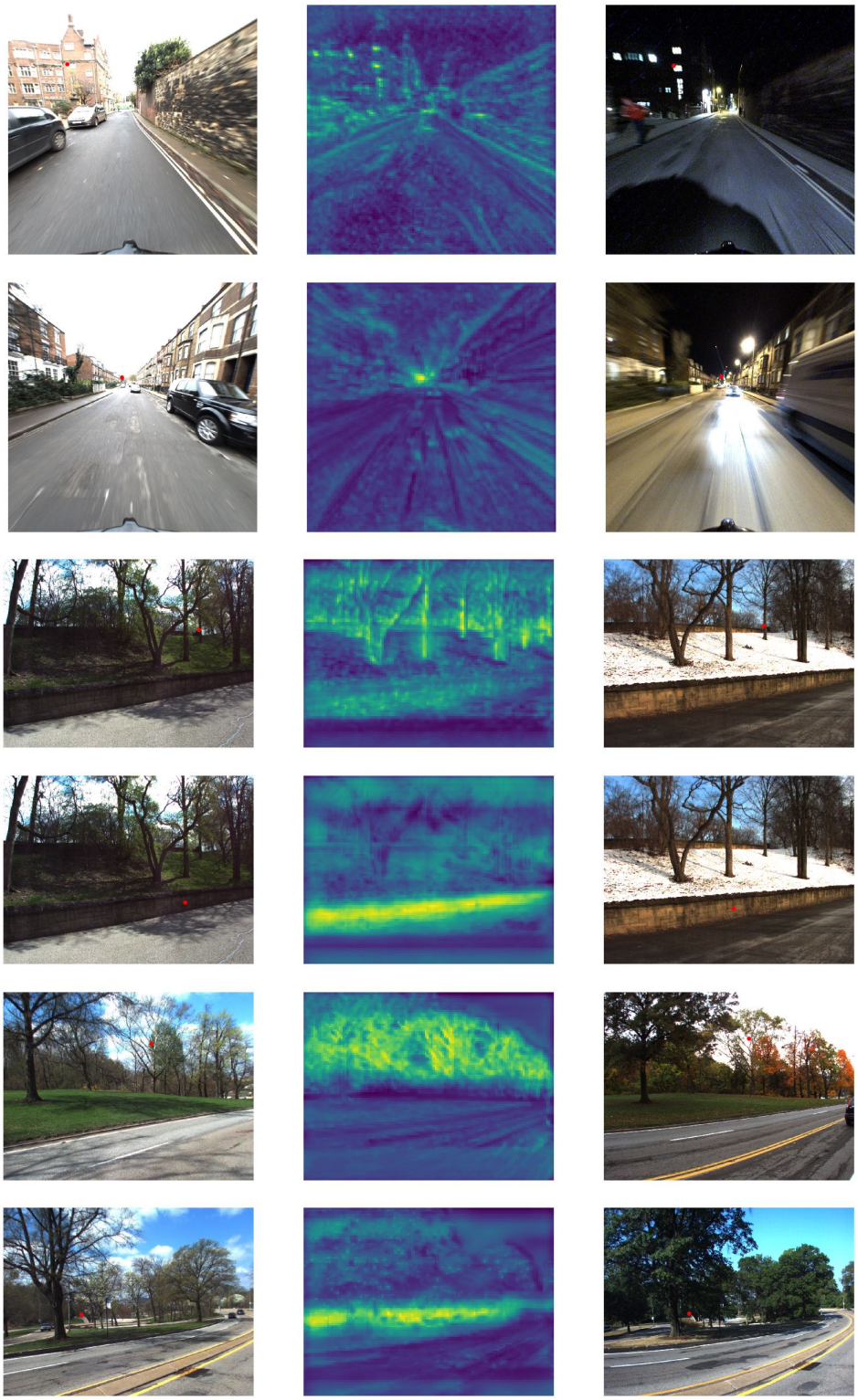


Figure A.2: **Correlation maps visualization.** Left column: Retrieved database image with a reprojected 3D point from the local point cloud. Middle column: Correlation map obtained for the sparse descriptor. Right column: Query image. The bottom-three triplets show cases where matching is ambiguous. Such cases are usually dismissed thanks to the ratio test.

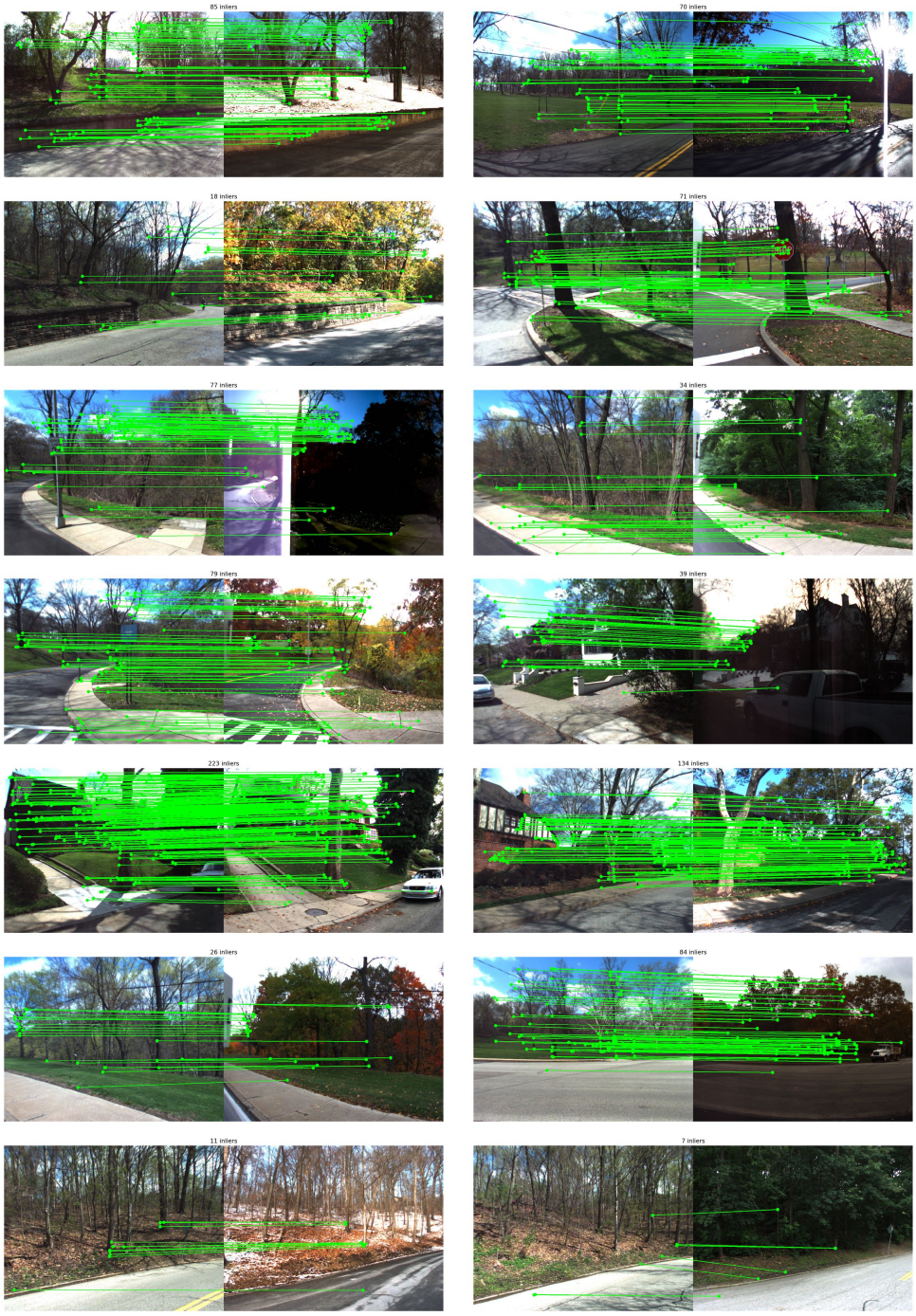


Figure A.3: **Inlier correspondences obtained using RANSAC+PnP.** We show correspondences obtained with our 'Sparse-to-Dense Hypercolumn Matching' method, on difficult vegetation scenes from Extended CMU Seasons. The bottow row shows failure cases due to a failed global image retrieval.

A.2 Additional S2DNet results

In this section, we provide additional results on S2DNet. In Section A.2.1 we report experiment details that were used to run our evaluations and show qualitative results in Section A.2.2.

A.2.1 Experiment details

A.2.1.1 Cyclic Verification

As said in Section 3.6.2.2, we not only filter out correspondences using Eq. (3.5) but we also remove correspondences which do not pass the cyclic check of matching back on their source pixel. This is equivalent to performing a mutual nearest-neighbor verification as it is done with D2-Net (Dusmanu et al., 2019) and R2D2 (Revaud et al., 2019). To perform this verification, we measure the distance between a source keypoint $\mathbf{p}_{\mathrm{A}}^{n}$ and its cyclic correspondent after running the sparse-to-dense matching both ways and remove the correspondence if the following condition is not satisfied:

$$d_{\text{cyclic}} = \|\mathbf{p}_{\mathrm{A}}^{n} - \mathbf{p}_{\mathrm{A}}^{n\,*}\| < \nu\,, \tag{A.1}$$

where

$$\mathbf{p}_{\mathrm{A}}^{n\,*} = \underset{\mathbf{p}\in\Omega}{\operatorname{argmax}}\ \mathtt{C}_{\mathbf{p}_{\mathrm{B}}^{n\,*}}^{\mathrm{B}\to\mathrm{A}}[\mathbf{p}] \tag{A.2}$$

and

$$\mathbf{p}_{\mathrm{B}}^{n\,*} = \underset{\mathbf{p}\in\Omega}{\operatorname{argmax}}\ \mathtt{C}_{\mathbf{p}_{\mathrm{A}}^{n}}^{\mathrm{A}\to\mathrm{B}}[\mathbf{p}]\,. \tag{A.3}$$

In our all experiments, we use a cyclic distance threshold of $\nu = 1$ pixel. In Table A.1, we report the impact of this threshold on the mean matching accuracy.

A.2.1.2 Local Features Evaluation

The local features benchmark (Sattler et al., 2018) couples the localization task with a multiview 3D reconstruction task. As discussed in Section 3.5, the nature of sparse-to-dense matching in S2DNet does not guarantee the uniqueness of detections across multiple images. Thus, performing 3D reconstruction with S2DNet would result in a very high number of triangulated landmarks with low track lengths. Therefore, we perform the preliminary 3D reconstruction step with an off-the-shelf feature detector in a sparse-to-sparse fashion instead. Since we are dealing with daytime image pairs which are easier to match, we find that this is sufficient to obtain an accurate triangulation. We use the SURF (Bay et al., 2006) detector as we found it provided the best results. Indeed, SuperPoint (Detone et al., 2018) detects fewer keypoints which harms the performance

ν	**MMA@1**	**MMA@2**	**MMA@3**	**MMA@10**
1.0	0.563	0.747	0.805	0.911
2.0	0.548	0.749	0.814	0.915
5.0	0.537	0.743	0.808	0.916
10.0	0.532	0.738	0.802	0.916

Table A.1: **Cyclic Verification.** We report the MMA on HPatches (Balntas et al., 2017) for several cyclic distance thresholds ν, using SuperPoint (Detone et al., 2018) detections and $\tau = 0.2$. We find that stricter thresholds improve the MMA at 1 pixel, while slightly damaging the coarser correspondences. In all our localization experiments, we use $\nu = 1.0$

under strong changes of scale. We then relocalize query images using S2DNet adopting this time a sparse-to-dense approach, and using triangulated keypoints as source detections.

A.2.1.3 Hierarchical Localization

In the day-night visual localization benchmark (Sattler et al., 2018), we use S2DNet to perform hierarchical localization. We first perform image retrieval using DenseVLAD (Torii et al., 2015) global image descriptors to fetch the top-20 nearest neighbours of both daytime and nighttime queries. Similar to (Sarlin et al., 2019), we compute a covisibility graph on the retrieved database images to cluster 3D points, leading to a reduced set of places. For each landmark, we pre-compute sparse descriptors using S2DNet and perform sparse-to-dense matching on the query image to find its correspondent. The subsequent 3D-2D correspondences are then fed to a Perspective-n-Point (PnP) solver (Kneip et al., 2011) inside a RANSAC (Fischler & Bolles, 1981) loop.

We compare our method to several baselines provided by the benchmark authors. Active Search (AS) (Sattler et al., 2017a) and City Scale Localization (CSL) (Svärm et al., 2017) are both 2D-3D direct matching methods representing the current state-of-the-art in terms of accuracy. Semantic Match Consistency (SMC) (Toft et al., 2018) applies a semantic segmentation-based match rejection to improve the predicted poses. We report the results of pure image retrieval-based approaches using DenseVLAD (Torii et al., 2015) and NetVLAD (Arandjelovic et al., 2016). For these methods, the query pose is approximated by the pose of the top-1 retrieved database image.

For hierarchical approaches, S2DHM is the closest to ours. This method also performs sparse-to-dense matching, but is trained with weak supervision and computes downsampled correspondence maps. One main advantage of this method is that it is pre-trained on RobotCar daytime and nighttime images for the task of image retrieval. These training images are separate from the reference and evaluation set but still very similar visually

to RobotCar evaluation images. We report the results of HF-Net (Sarlin et al., 2019), which performs hierarchical localization with NetVLAD and SuperPoint (Detone et al., 2018). Lastly, we report the performance of D2-Net (Dusmanu et al., 2019) provided by the authors.

A.2.1.4 InLoc evaluation

Since S2DNet was trained on outdoor images, we found that the confidence scores are overall lower when applied indoors, and confidence thresholdings $\tau > 0$ result in very few correspondences and damages the overall localization results. Thus, for the InLoc experiments, we use $\tau = 0$. InLoc query images are also of very high resolution (3024×4032 pixels). To speed up the matching process, we downscale all images to a maximum width or length of 1200 pixels.

A.2.2 Qualitative Results

We report in Figure A.4 example correspondence maps for InLoc (Taira et al., 2018) and RobotCar nighttime query images. We show the intermediate correspondence maps, as well as the final aggregated map and the retrieved correspondent. We report in Figure A.5 inliers for the Sparse PE pipeline of InLoc (Taira et al., 2018), and show in Figure A.6 two failure cases.

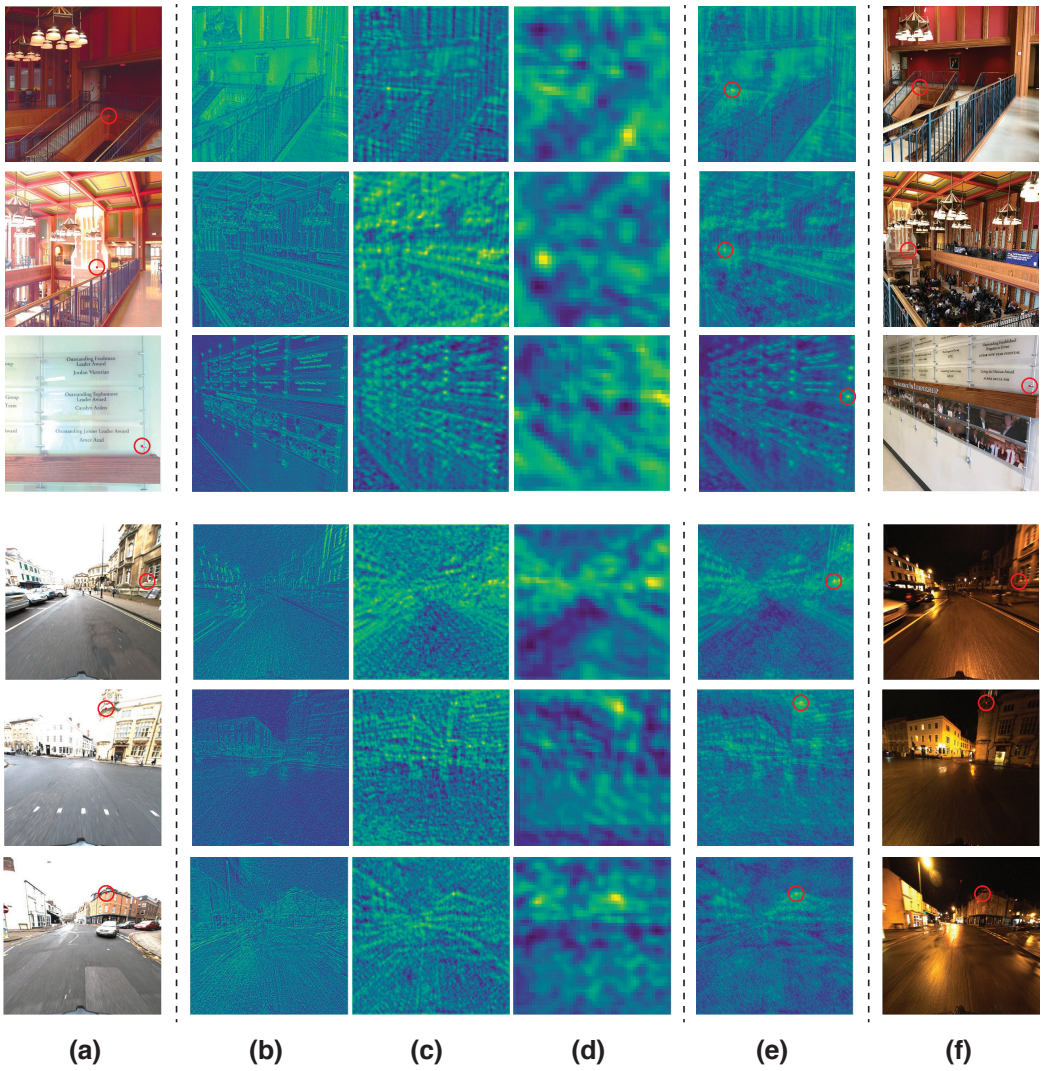


Figure A.4: **Correspondence maps examples.** From left to right: Reference image with a keypoint detection (a), intermediate correspondence maps predicted by S2DNet (b, c, d), aggregated pre-softmax correspondence map (e) and retrieved correspondent in the query image (f). The top three images are from InLoc (Taira et al., 2018) and the bottom three from RobotCar Seasons (Maddern et al., 2017).

Figure A.5: **Inlier Correspondences on InLoc (Taira et al., 2018).** Despite strong changes in scale, illumination and the large scale of the database, S2DNet manages to build robust and accurate correspondences.

Figure A.6: **Failure Cases Examples on InLoc (Taira et al., 2018).** Due to the repetitive structures present in the dataset, we find failure cases where such structures are matched despite being from two different places. In InLoc however, such cases are typically discarded when performing the dense pose verification (Dense PV) step.

Appendix B

Additional results on the Neural Reprojection Error

In the following pages, we present experimental details about the Neural Reprojection Error.

B.1 Derivation of Equation 4.9

In this section, we show how Eq. (4.9) is obtained.

The robust dense loss map $\mathtt{L}_{\mathtt{Q},n,\sigma}$ can be smoothed using an isotropic Gaussian kernel as follows:

$$
\begin{aligned}
&\breve{\mathtt{L}}_{\mathtt{Q},n,\sigma}(\mathbf{p}) := \sum_{\mathbf{r}} k_\sigma(\|\mathbf{r}\|)\, \mathtt{L}_{\mathtt{Q},n}(\mathbf{p}+\mathbf{r}) \\
&= \mathtt{L}_{\mathtt{Q},n}(\mathbf{out}) \sum_{\mathbf{r}} k_\sigma(\|\mathbf{r}\|) \\
&\quad + \sum_{\mathbf{r}} k_\sigma(\|\mathbf{r}\|)\left(\mathtt{L}_{\mathtt{Q},n}(\mathbf{p}+\mathbf{r}) - \mathtt{L}_{\mathtt{Q},n}(\mathbf{out})\right) && \text{(B.1)} \\
&= \sum_{\mathbf{r}} k_\sigma(\|\mathbf{r}\|)\left(\mathtt{L}_{\mathtt{Q},n}(\mathbf{p}+\mathbf{r}) - \mathtt{L}_{\mathtt{Q},n}(\mathbf{out})\right) + \mathrm{cst}_{\mathbf{p}} && \text{(B.2)} \\
&= \sum_{\mathbf{q}\in\Omega_{\mathtt{Q}}} k_\sigma(\|\mathbf{q}-\mathbf{p}\|)\left(\mathtt{L}_{\mathtt{Q},n}(\mathbf{q}) - \mathtt{L}_{\mathtt{Q},n}(\mathbf{out})\right) + \mathrm{cst}_{\mathbf{p}} && \text{(B.3)}
\end{aligned}
$$

$$= \sum_{\mathbf{q} \in \Omega_{\mathtt{Q}}} k_\sigma \left(\|\mathbf{q} - \mathbf{p}\| \right) \left(\mathtt{L}_{\mathtt{Q},n} \left(\mathbf{q} \right) - \ln |\mathring{\Omega}_{\mathtt{Q}}| \right) + \mathrm{cst}_{\mathbf{p}} \tag{B.4}$$

$$= \sum_{\mathbf{q} \in \Gamma_{\mathtt{Q},n}} k_\sigma \left(\|\mathbf{q} - \mathbf{p}\| \right) \left(\mathtt{L}_{\mathtt{Q},n} \left(\mathbf{q} \right) - \ln |\mathring{\Omega}_{\mathtt{Q}}| \right) + \mathrm{cst}_{\mathbf{p}} \tag{B.5}$$

where $k_\sigma \left(\|\mathbf{r}\| \right) := \frac{1}{2\pi\sigma^2} e^{-\frac{\|\mathbf{r}\|^2}{2\sigma^2}}$ is an isotropic Gaussian kernel with standard variation σ and $\Gamma_{\mathtt{Q},n}$ is the set of pixel locations whose corresponding values in $\mathtt{L}_{\mathtt{Q},n}$ are lower than $\ln |\mathring{\Omega}_{\mathtt{Q}}|$. Equation B.5 leads to the smoothed cost function:

$$\breve{\mathcal{L}}_\sigma \left(\mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}} \right) :=$$
$$\sum_{n=1}^{N} \sum_{\mathbf{q} \in \Gamma_{\mathtt{Q},n}} \text{-} \Big(\ln |\mathring{\Omega}_{\mathtt{Q}}| \text{-} \mathtt{L}_{\mathtt{Q},n} \left(\mathbf{q} \right) \Big) k_\sigma \Big(\big\| \mathbf{q} \text{-} \omega \left(\mathbf{u}_n^{\mathtt{G}}, \mathtt{R}_{\mathtt{QG}}, \mathbf{t}_{\mathtt{QG}} \right) \big\| \Big) \,, \tag{B.6}$$

which is a robust non-linear least squares problem and therefore can be minimized using the IRLS algorithm.

B.2 Technical details

B.2.1 Network Architectures (Sec. 4.5)

Coarse network architecture. The purpose of the coarse network $\mathcal{F}_{\text{coarse}}$ is to provide robust descriptors that are used to obtain a coarse pose estimate. To deal with ambiguous cases, it should leverage image context. This motivates a deep architecture with a wide receptive field and a large descriptor size. On the other hand, the network should output dense descriptors of sufficient resolution to reliably estimate a coarse camera pose. We experimentally found that an effective stride of 16 is sufficient. To satisfy these specifications, we opted for an Inception-v3 (Szegedy et al., 2016a) backbone and modified it accordingly. We changed some kernel sizes and truncated the network at the layer Mixed-6e. In the end our final architecture has a receptive field of 927 pixels and produces dense descriptors of size $H/16 \times W/16 \times 1280$.

Fine network architecture. The purpose of the fine network $\mathcal{F}_{\text{fine}}$ is to provide discriminative high-resolution descriptors that are used to refine the coarse pose estimate. However, producing high-resolution descriptors takes a lot of memory. This motivates a deep architecture with a small receptive field and a small descriptor size. We experimentally found that an effective stride of 2 is a good balance between accuracy and memory consumption. To satisfy these specifications, we opted again for a modified Inception-

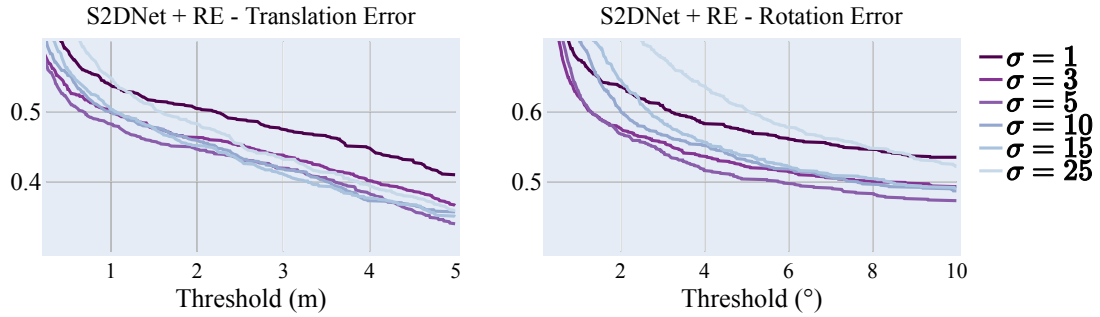


Figure B.1: **Tuning the hyperparameter of an RE-based pose estimator:** We report the cumulative error curves in pose estimation (lower is better), on the hardest category of our Megadepth study, for the RE-based pose estimator that consists in minimize Eq. 10. We find that a careful hyperparameter tuning is very important. On the contrary, our novel formalism leads to a loss that does not possess any hyperparameter.

v3 (Szegedy et al., 2016a) backbone. We only keep the stride of 2 at the first layer and remove any Max-Pooling layer, and we truncate the model at the Mixed-5d layer. Our final architecture has a receptive field of 43 pixels and produces dense descriptors of size $H/2 \times W/2 \times 288$.

Implementation details. The coarse network $\mathcal{F}_{\text{coarse}}$ and the fine network $\mathcal{F}_{\text{fine}}$ are trained independently. Both networks use the same training data which comes from the MegaDepth dataset (Li & Snavely, 2018). As D2-Net (Dusmanu et al., 2019), we remove scenes which overlap with the PhotoTourism (Thomee et al., 2016; Trulls et al., 2019) test set. We train our networks on image pairs ($\mathtt{I}_{\mathtt{S}}$ and $\mathtt{I}_{\mathtt{T}}$) with an arbitrary overlap.

To train $\mathcal{F}_{\text{fine}}$, we extract random crops of size 800×800 and randomly sample a maximum of 64 3D points visible in both $\mathtt{I}_{\mathtt{S}}$ and $\mathtt{I}_{\mathtt{T}}$. Using such large crops may seem an overkill since $\mathcal{F}_{\text{fine}}$ has a small receptive field. Let us highlight that using $C \times C$ crops allows to produce correspondence maps of size $C/2 \times C/2$ which essentially consists in comparing each source patch against C^2 target patches. Thus, even if $\mathcal{F}_{\text{fine}}$ has a small receptive field, the larger the crops during training the better the descriptors, and 800×800 is the maximum size that could fit in memory.

To train $\mathcal{F}_{\text{coarse}}$, we use entire images as inputs since the network has a very large receptive field and randomly sample a maximum of 64 3D points visible in both $\mathtt{I}_{\mathtt{S}}$ and $\mathtt{I}_{\mathtt{T}}$. Each network is trained using early stopping on the MegaDepth validation set. We use Adam (Kingma & Ba, 2015) with an initial learning rate of 10^{-3} and apply a multiplicative decaying factor of $e^{-0.1}$ at every epoch.

B.2.2 Timing

We run all our training and experiments on a machine equipped with an Intel(R) Xeon(R) E5-2630 CPU at 2.20GHz, and an NVIDIA GeForce GTX 1080Ti GPU. The timing results reported in Tab. 4.4 where obtained using a Python implementation of the previously described algorithms. Source code will be made available.

B.2.3 Implementation details about the RE-based vs. NRE-based pose estimators study

- In our RE-based vs. NRE-based pose estimators study, we used LO-RANSAC (Chum et al., 2003), GC-RANSAC (Baráth & Matas, 2018) and MAGSAC++ (Baráth et al., 2020) implementations provided in OpenCV 4.5.0 [1].
- We show in Fig. B.1 the cumulative errors curves for several σ values when minimizing Eq. 4.1 on the hardest category of our Megadepth (Li & Snavely, 2018) study. These results stress how important a careful hyperparameter tuning is in standard RE pose estimators.
- Throughout Chapter 4 we run the coarse GNC with decreasing σ values ranging from 2.0 to 0.6. For the fine GNC, we use values between 8.0 and 0.6.

[1] https://docs.opencv.org/master/d9/d0c/group__calib3d.html

Appendix C

Additional results on NeurHal

In the following pages, we present additional qualitative results, experiments and technical details about our visual correspondence hallucination method NeurHal.

C.1 Additional qualitative results

C.1.1 Qualitative correspondence hallucination results and failure cases

To further demonstrate the ability of NeurHal to perform visual correspondence hallucination, we report in Fig. C.1 and Fig. C.2 qualitative results on ScanNet (Dai et al., 2017) and Megadepth (Li & Snavely, 2018) respectively on scenes that were not seen at training-time. In the target image and in the (negative log) correspondence map, the red dot represents the ground truth keypoint's correspondent. The dashed rectangles represent the borders of the target images.

Let us recall that NeurHal outputs probability distributions (a.k.a correspondence maps) *assuming the two input images are partially overlapping*. It is essential to keep this assumption in mind when looking at these qualitative results. For instance, concerning the example Fig. C.1 (b) (middle), it is very difficult for our human visual system to be sure that the two images are actually overlapping, and consequently the network prediction seems to good to be true. However, if we *assume* that there is an overlap, we realize that it is actually possible to perform correspondence hallucination, by drawing out the two skirting boards, to correctly outpaint the correspondent.

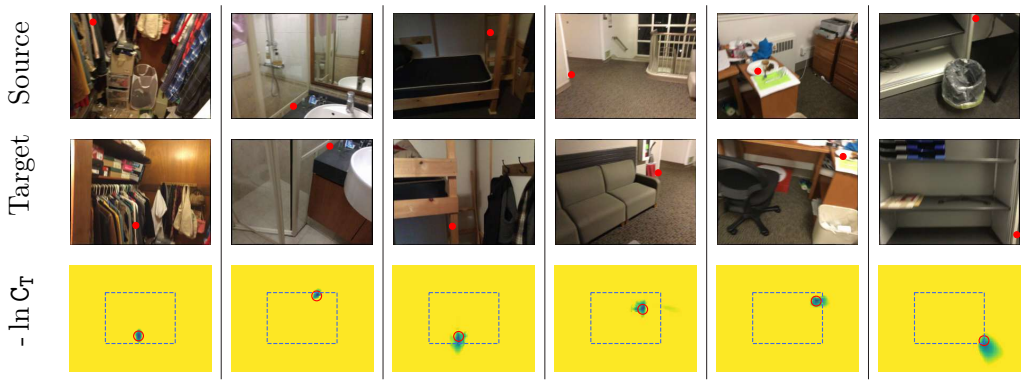
In fact, this overlapping assumption has a regularization effect in cases where the covisible image areas show no distinctive regions, and one image could be at an infinite translation of the other, e.g. Fig. C.1 (b) (second to last).

In Fig. C.1 (d) and Fig. C.2 (d) we show failure cases where the correspondence maps

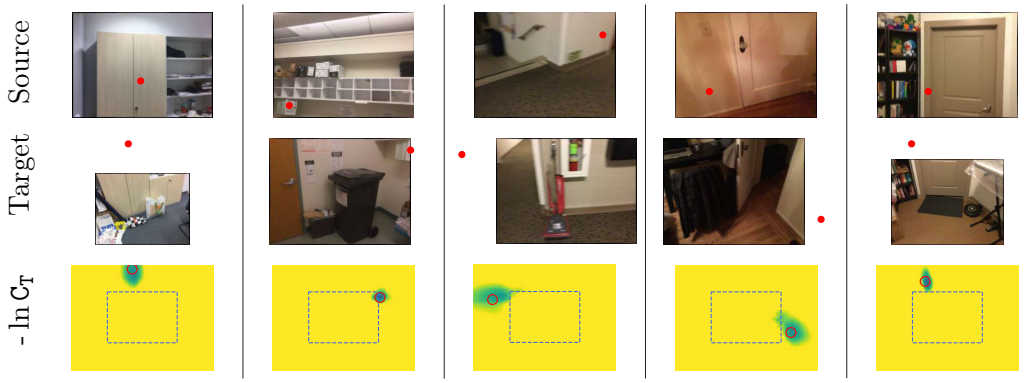
modes predicted by NeurHal are either partially or completely off. We find that failure cases often correlate with strongly ambiguous image pairs, or images that have extremely limited visual overlap.

C.1.2 Qualitative camera pose estimation results

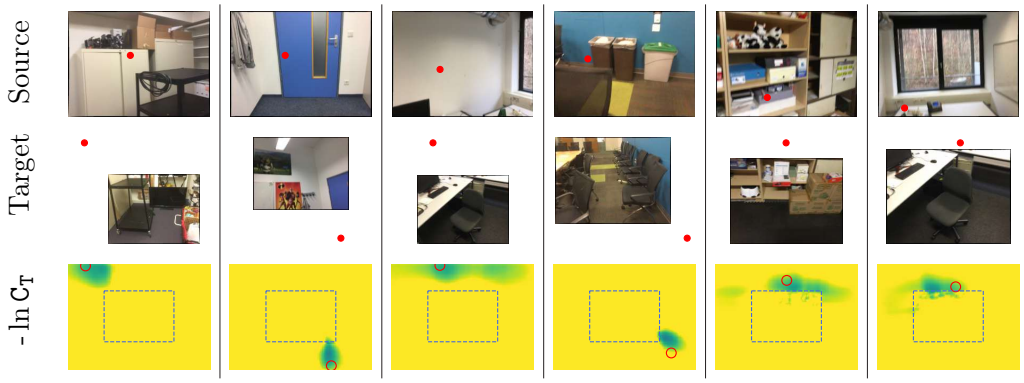
We show in Fig. C.3 qualitative results in camera pose estimation on low-overlap images from ScanNet (Dai et al., 2017), for NeurHal and its three best-performing competitors. For every method we display the keypoints used as input to the camera pose estimator in the source image, along with their reprojection at the estimated camera pose in the target image. For methods using the pose estimator from (Chum et al., 2003), the keypoints are those that have been successfully matched. When using the pose estimator of Germain et al. (2021a), the keypoints are those involved in the prediction of the dense NRE maps. We color in keypoints based on their spatial 2D position in the source image. We find that NeurHal strongly benefits from its outpainting ability, in comparison with all other competitors which struggle to find both sufficient and reliable correspondences. We also report in Fig. C.4 failure cases for NeurHal. We find that such cases correspond to image pairs exhibiting extremely limited visual overlap, strong camera pose rotations and overall significant ambiguities.



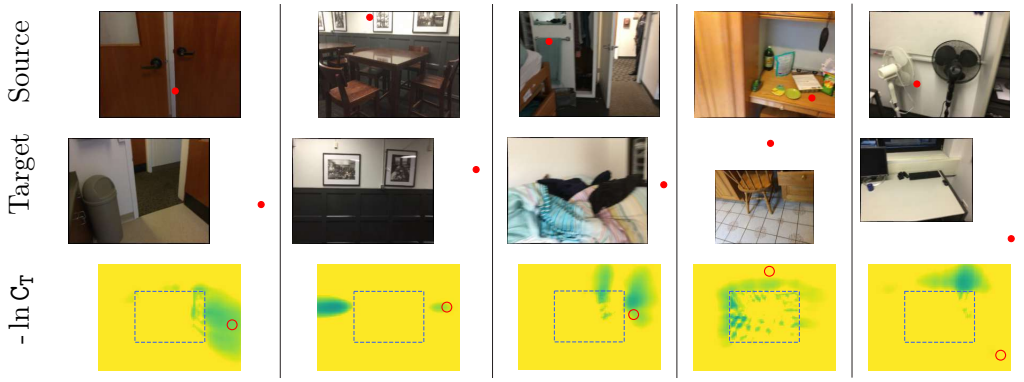
(a) Identification / Inpainting Examples



(b) Outpainting Examples

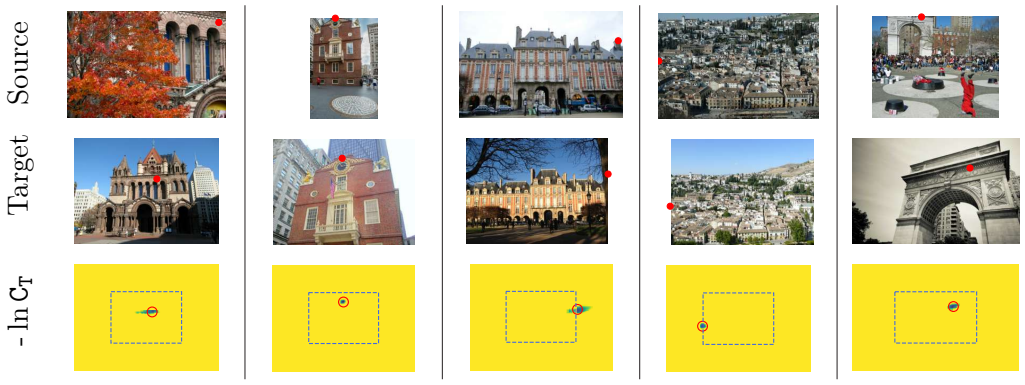


(c) Challenging Examples

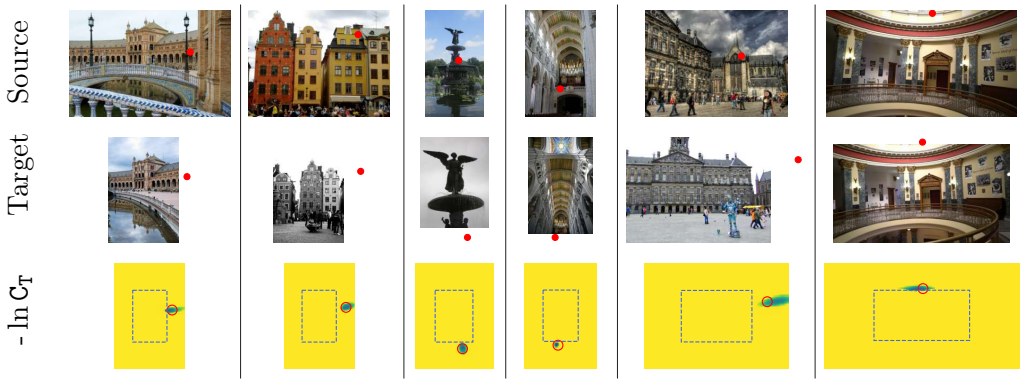


(d) Borderline / Failure Cases

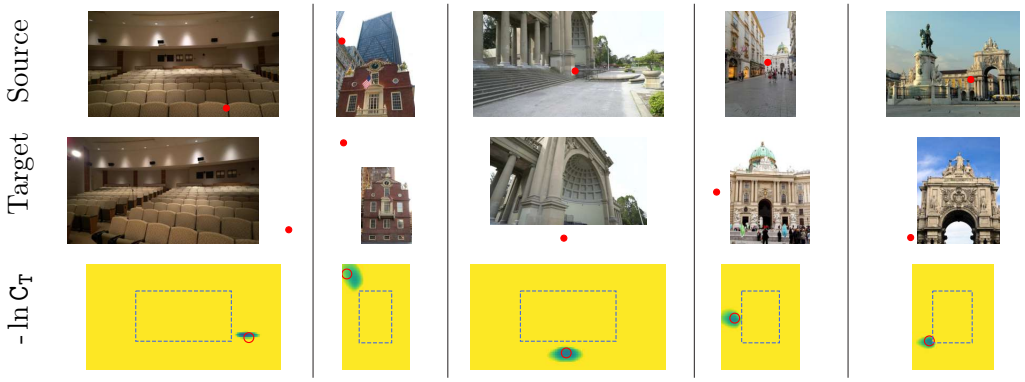
Figure C.1: **Additional qualitative ScanNet (Dai et al., 2017) examples.** See text for details.



(a) Identification / Inpainting Examples



(b) Outpainting Examples



(c) Challenging Examples

(d) Borderline / Failure Cases

Figure C.2: **Additional qualitative Megadepth (Li & Snavely, 2018) examples.** See text for details.

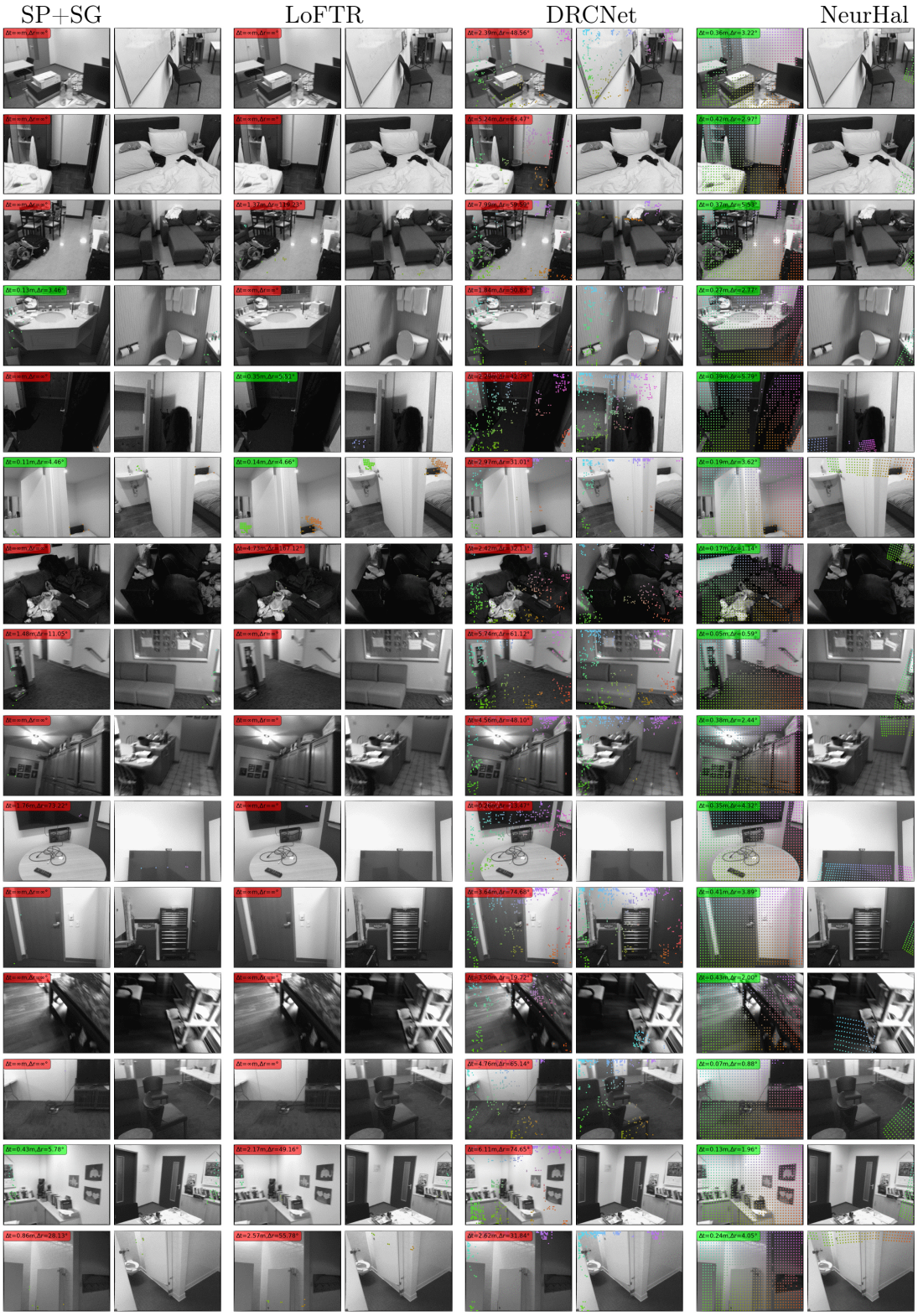


Figure C.3: **Qualitative camera pose estimation results on low-overlap images from ScanNet (Dai et al., 2017):** We show for every method keypoints used as input for the camera pose estimator in the source image (left image), along with their predicted reprojection in the target image (right image). We color-code keypoints based 2D spatial position in the source image. We also report for every pair and every method the camera pose estimation error in translation and rotation, colored in green when the pose is less than $\tau_t = 0.5m$ and $\tau_r = 10.0°$, and in red otherwise.

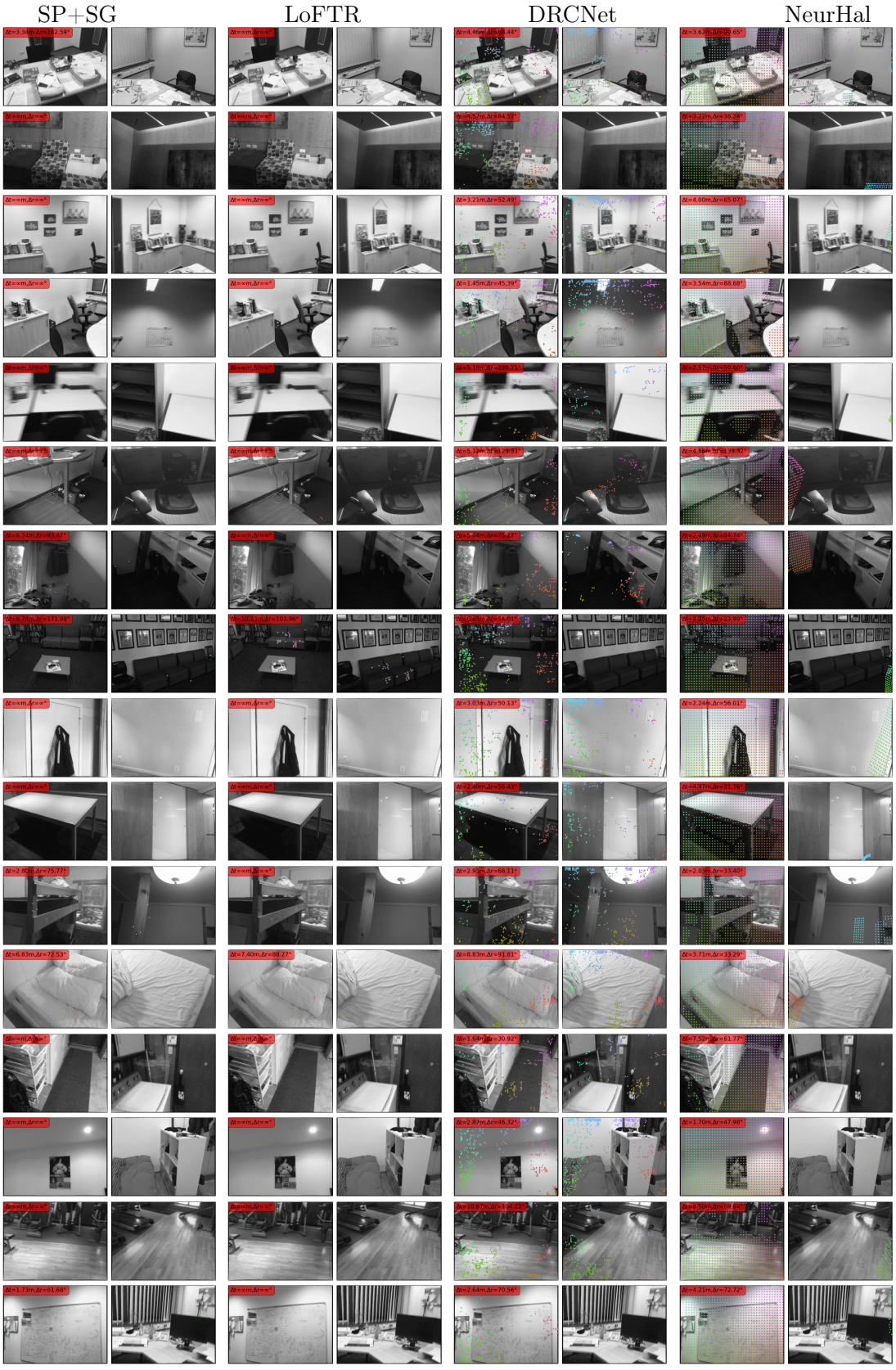


Figure C.4: **NeurHal failure cases on low-overlap images from ScanNet (Dai et al., 2017):** We report cases where NeurHal fails to estimate a camera pose with an error less than $\tau_t = 0.5m$ and $\tau_r = 10.0°$. We find these cases often correlate with extremely low covisibility coupled with strong camera rotations.

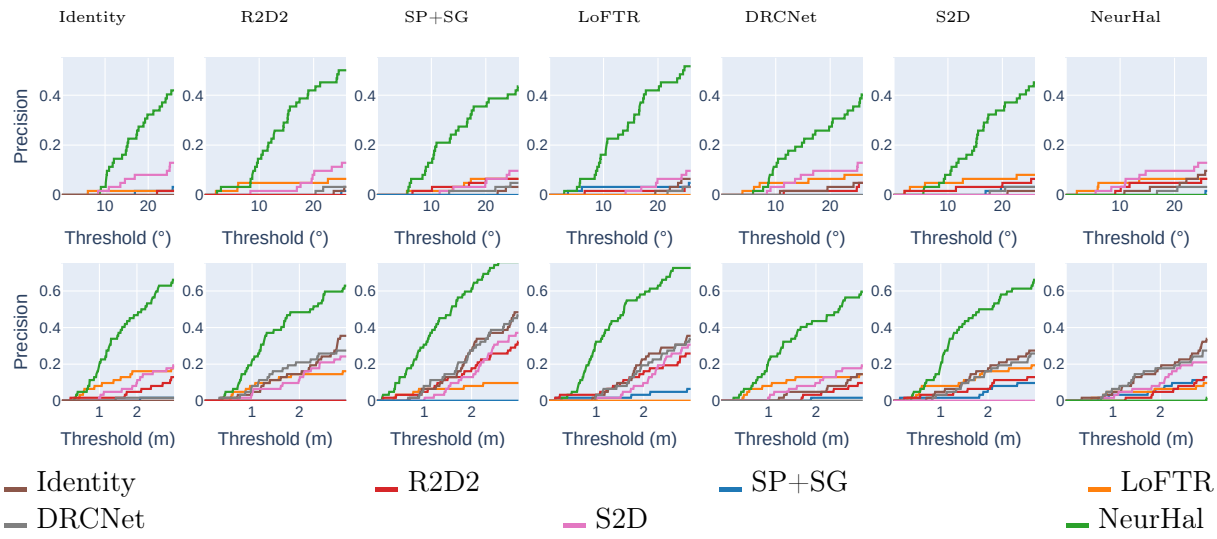


Figure C.5: **Camera pose estimation experiment - Worst cases:** We report the performance of NeurHal and state-of-the-art feature matching methods on ScanNet (Dai et al., 2017) image pairs with visual overlaps between 2% and 5%. For every column, we subselect the 25% of images pairs with the worst predictions for a given method. We find that in all cases, NeurHal strongly outperforms its competitors. On the contrary, on the worst NeurHal predictions state-of-the-art methods achieve a much lower performance, on par or lower than the Identity predictions.

C.2 Additional indoor pose estimation results

In addition to the results presented in Fig. 5.6, we report in Fig. C.5 the performance of NeurHal and state-of-the-art feature matching methods on ScanNet (Dai et al., 2017) image pairs with visual overlaps between 2% and 5%. For every method, we subselect the 25% of images pairs with the worst predictions, and compare it with the performance of its competitors. We find that in all cases, NeurHal strongly outperforms its competitors. On the worst NeurHal predictions, state-of-the-art methods achieve a much lower performance. For this category we can observe that all NeurHal competitors are either on par or achieve a lower performance than the Identity predictions.

This figure highlights the fact that when NeurHal fails to correctly estimate the camera pose, all the competitors also fail since all the methods perform similarly to the "identity" method, i.e. the method that consists in systematically predicting the identity pose.

C.3 Technical details

C.3.1 Architecture details

NeurHal architecture can be separated in two building blocks: the convolutional backbone and the multi-head attention block.

Convolutional backbone. The convolutional backbone consists of a truncated Inceptionv3 (Szegedy et al., 2016a) model (up to Mixed-6a, 768-dimensional descriptors), modified following the NRE to provide, in the case of ScanNet (Dai et al., 2017), a 1/8 output-to-input resolution ratio. To help with memory consumption we apply a simple 2D convolutional layer to compress the descriptor size to 384. In the case where $\gamma > 0$, we subsequently pad $\mathtt{H}_\mathtt{T}$ with the learned vector $\boldsymbol{\lambda}$, producing $\mathtt{H}_{\mathtt{T},\mathrm{pad}}$.

Positional encoding. After computing $\mathtt{H}_\mathtt{S}$ and $\mathtt{H}_{\mathtt{T},\mathrm{pad}}$ with the convolutional backbone, positional encoding is applied to both dense feature maps. Similarly to SuperGlue (Sarlin et al., 2020), we use a 6-layer MLP of size (32, 64, 128, 256, 384), mapping a positional meshgrid between $(-1, 1)$ (centered around the image center) to higher dimensionalities. BatchNorm and ReLU layers are placed between every module. In our experiments, we tried adding more positional encoding layers but found it did not make a difference in performance. After applying the positional encoding, sparse descriptors $\{\mathbf{h}_{\mathtt{S},n}\}_{n=1\ldots N}$ are bilinearly interpolated at $\{\mathbf{p}_{\mathtt{S},n}\}_{n=1\ldots N}$ in $\mathtt{H}_\mathtt{S}$.

Self-attention. Following the positional encoding, a single multi-head attention layer is applied on $\mathtt{H}_{\mathtt{T},\mathrm{pad}}$, with 4 heads. It consists of a standard dot-product attention (Vaswani

Layer	# of parameters
CNN	2.4 M
Positional Encoding	142 K
Self-Attention	1.9 M
Cross-Attention	7.2 M
Total	11.7 M

Table C.1: **Number of parameters in NeurHal**

et al., 2017), coupled with a gating mechanism. For a given query Q, key K and value V, we compute the attention as $\text{Attention}(Q, K, V) = \text{softmax}(g * QK^T)V$ where $g = \sigma(max(QK))$. To mitigate the quadratic cost of the dot-product attention, we also apply a max-pooling operator on keys and values with a stride of 2, as we empirically found it had very little impact on performance. We also tried using a Linear Transformer (*e.g.* LinFormer (Katharopoulos et al., 2020)) architecture, but despite trying numerous variants we found it consistently damaged the convergence of the model.

Cross-attention. Using the same attention-layer design, we subsequently apply it once between $\{\mathbf{h}_{\mathtt{S},n}\}_{n=1...N}$ and $\mathtt{H}_{\mathtt{S}}$. This layer allows for communication between the interpolated source descriptors which will be used to produce the final correspondence maps, and the original dense source image content. Then, we apply k cross-attention layers between $\{\mathbf{h}_{\mathtt{S},n}\}_{n=1...N}$ and $\mathtt{H}_{\mathtt{T},\text{pad}}$. We empirically found these layers to be most important, as they allow for direct communication between the sparse source descriptors and the dense target feature maps, prior to the correspondence maps computation. After trying different values for k and with memory consumption in mind, we settled for $k = 4$ in all our experiments.

Implementation. The model is implemented in PyTorch (Paszke et al., 2017). For an indoor sample with 2000 keypoints it has an average throughput of 8.84 image/s on an NVIDIA RTX 3070 GPU. We report the number of parameters in our model in Table C.1.

C.3.2 Datasets and Training details

ScanNet. The ScanNet (Dai et al., 2017) dataset is a large-scale indoor dataset containing monocular RGB videos and dense depth images, along with ground truth absolute camera poses. As SuperGlue (Sarlin et al., 2020) and LoFTR (Sun et al., 2021), we precompute visual overlaps between all image pairs for both training and test scenes. For the training set we sample images with a visual overlap between 2% and 50% from the

ScanNet training scenes, which provides us with challenging images to handle. We assemble $6M$ image pairs, and randomly subsample $200k$ pairs at every training epoch. For testing images, we sample $2,500$ image pairs with overlaps between 2% and 80% from the ScanNet testing scenes, using several bins to ensure sampling is close to being uniform. For both training and testing images, we sample keypoints in the source image along a regular grid with cell sizes of 16 pixels. We remove keypoints with invalid depth, as well as those where the local depth gradient is too hight, as the depth information might not be reliable. We mark keypoints falling outside the target image plane as being outpainted, and we automatically detect keypoints to inpaint through a cyclic projection of the source keypoints to the target image and back. The remaining keypoints are labeled as identifiable. For all ScanNet experiments, NeurHal uses a 1/8 output-to-input resolution ratio, with a target correspondence map maximum edge size of 80 pixels (when $\gamma = 0\%$).

Megadepth. We use Megadepth (Li & Snavely, 2018) to train and evaluate NeurHal on outdoor images. This dataset contains over one million images captured in touristic places, and split in 196 scenes.

To train NeurHal and following the NRE, we use the provided SIFT (Lowe, 2004)-based 3D reconstruction which was made with COLMAP (Schönberger & Frahm, 2016). As for ScanNet (Dai et al., 2017) for a given image pair we consider all keypoints falling outside the target field of view as potentially outpaintable. Because the sparse 3D point cloud comes from SfM, we find however that very little keypoints can be marked as inpainted. Indeed, no 3D reconstruction is applied to objects or people occluding the scene. To allow for a wide variety of image pairs we use the sparse reconstruction to estimate the visual overlap and sample pairs with an overlap between 20% and 100%. We however find this overlap estimation to be quite unreliable, as only part of the scene is usually reconstructed.

Since Megadepth (Li & Snavely, 2018) images are of much higher resolution than ScanNet (Dai et al., 2017), we configure NeurHal to use a 1/16 output-to-input resolution (with a simple max-pooling layer in the CNN). We set the target correspondence map maximum edge size of 60 pixels (when $\gamma = 0\%$), to allow for space in memory when $\gamma = 50\%$.

Overlap estimation. For a given pair of images, we approximate the visual overlap by computing the covisibility ratio of keypoints for every image pair. For a given source and target image pair, we first compute the source-to-target and target-to-source covisibility ratios using ground truth depth data and camera poses. We then define the visual overlap as the minimum between both ratios. On Megadepth we find this overlap estimation to be fairly noisy, as depth is only partially known.

Optimizers and scheduling. On both datasets NeurHal is trained for a maximum of 40 epochs. We use an initial learning rate of 10^{-3}, with a linear learning rate warm-up in 3 epochs from 0.1 of the initial learning rate. As (Sun et al., 2021), we decay the learning rate by 0.5 every 8 epochs starting from the 8th epoch. We apply the linear scaling rule and use a batch size of 8 over 8 NVIDIA V100 GPUs. We use the AdamW (Loshchilov & Hutter, 2019) optimizer, with a weight decay of 0.1. In all training procedures, we randomly initialize the model weights.

C.3.3 Evaluation Details

Evaluation protocol. All baselines follow the same standard protocol in which we: 1) Compute 2D-2D correspondences between the reference image and the query image, 2) Lift these 2D-2D correspondences to 2D-3D correspondences using the available 3D information for the reference image, 3) Estimate the camera pose given these 2D-3D correspondences by minimizing the Reprojection Error (RE), i.e. applying LO-RANSAC+PnP (Chum et al., 2003) followed by a non-linear iterative refinement. This approach is widely used and leads to state-of-the-art results in visual localization benchmarks. We also include results for the NRE model *which we call S2D*. For the evaluation of Fig. 5.4, we find the inpainted and outpainted correspondents for LoFTR (Sun et al., 2021) and DRCNet (Li et al., 2020a) by fetching the argmax 2D coordinates in the 4D matching confidence volume. For S2D and NeurHal, we simply take the argmax in correspondence maps for the same set of keypoints.

RE-based methods. For all RE (Chum et al., 2003)-based methods, we estimate the camera pose using the pycolmap python binding. We tune the RANSAC threshold for optimal performance, and mark all cases where less than 3 valid correspondences (*i.e.* with a valid depth value) as failure cases (infinite pose error). The remaining parameters are left as default. We follow the evaluation instructions provided by each method, and use indoor weights for SP+SG (Sarlin et al., 2020) and the dual-softmax indoor weights for LoFTR (Sun et al., 2021). In the case of NeurHal +RE (Chum et al., 2003), we simply read the argmax of the predicted correspondence maps to obtain explicit 2D-to-3D correspondences.

NRE-based pose estimator. For both S2D (Germain et al., 2021a) and NeurHal we only use coarse models, which operate at either 1/8th or 1/16th of the original input resolution. We first retrain the S2D coarse model (fully-convolutional Inceptionv3 (Szegedy et al., 2016a), up to Mixed-6e) on the same training set as our method, with the same target resolution of 80 pixels. We refer to this model as S2D. Given correspondence maps and the depth map of the source image, we estimate the camera pose between the target

image and the source image using the NRE. For both S2D and NeurHal we use the same set of regularly sampled source keypoints, and we perform camera pose estimation first using P3P inside an MSAC (Torr & Zisserman, 2000) loop. We run P3P for a maximum of $5,000$ iterations over the top-20% correspondences. We then apply a coarse GNC (Blake & Zisserman, 1987) over all source keypoints with $\sigma_{max} = 2.0$ and $\sigma_{min} = 0.6$. Let us highlight that in all the camera pose experiments, the performances of NeurHal are obtained by predicting *only* low resolution correspondence maps.

Bibliography

Anoosheh, A., Sattler, T., Timofte, R., Pollefeys, M., & Gool, L. (2019). Night-to-day image translation for retrieval-based localization. *2019 International Conference on Robotics and Automation (ICRA)*, (pp. 5958–5964).

Arandjelovic, R., Gronát, P., Torii, A., Pajdla, T., & Sivic, J. (2016). NetVLAD: CNN Architecture for Weakly Supervised Place Recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Arandjelović, R. & Zisserman, A. (2012). Three things everyone should know to improve object retrieval. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 2911–2918).

Arandjelovic, R. & Zisserman, A. (2013). All About VLAD. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Arandjelović, R. & Zisserman, A. (2014). Visual vocabulary with a semantic twist. In *ACCV*.

Babenko, A. & Lempitsky, V. (2015). Aggregating local deep features for image retrieval. *2015 IEEE International Conference on Computer Vision (ICCV)*, (pp. 1269–1277).

Badino, H., Huber, D., & Kanade, T. (2011). The CMU Visual Localization Data Set. http://3dvis.ri.cmu.edu/data-sets/localization.

Bailey, T. & Durrant-Whyte, H. (2006). Simultaneous localization and mapping (slam): part ii. *IEEE Robotics & Automation Magazine*, 13, 108–117.

Baker, S. & Matthews, I. (2004). Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56, 221–255.

Balntas, V., Johns, E., Tang, L., & Mikolajczyk, K. (2016a). PN-Net: Conjoined Triple Deep Network for Learning Local Image Descriptors. In *ArXiv*.

Balntas, V., Lenc, K., Vedaldi, A., & Mikolajczyk, K. (2017). : (pp. 3852–3861).

Balntas, V., Li, S., & Prisacariu, V. (2018). Relocnet: Continuous metric learning relocalisation using neural nets. In *ECCV*.

Balntas, V., Riba, E., Ponsa, D., & Mikolajczyk, K. (2016b). Learning local feature descriptors with triplets and shallow convolutional neural networks. In *BMVC*.

Baráth, D. & Matas, J. (2018). Graph-cut ransac. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (pp. 6733–6741).

Barath, D., Matas, J., & Noskova, J. (2019). MAGSAC: Marginalizing Sample Consensus. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Baráth, D., Noskova, J., Ivashechkin, M., & Matas, J. (2020). Magsac++, a fast, reliable and accurate robust estimator. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 1301–1309).

Barron, J. T. (2019). A general and adaptive robust loss function. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 4326–4334).

Bay, H., Tuytelaars, T., & Van gool, L. (2006). SURF: Speeded Up Robust Features. In *European Conference on Computer Vision (ECCV)*.

Beaudet, P. (1978). Rotationally invariant image operators.

Bentoutou, Y., Taleb, N., Kpalma, K., & Ronsin, J. (2005). An automatic image registration for applications in remote sensing. *IEEE Transactions on Geoscience and Remote Sensing*, 43, 2127–2137.

Bian, J., Lin, W.-Y., Matsushita, Y., Yeung, S., Nguyen, T., & Cheng, M.-M. (2017). Gms: Grid-based motion statistics for fast, ultra-robust feature correspondence. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 2828–2837).

Bianco, S., Ciocca, G., & Marelli, D. (2018). Evaluating the performance of structure from motion pipelines. *J. Imaging*, 4, 98.

Blake, A. & Zisserman, A. (1987). *Visual Reconstruction*. MIT press.

Brachmann, E., Krull, A., Nowozin, S., Shotton, J., Michel, F., Gumhold, S., & Rother, C. (2017). Dsac — differentiable ransac for camera localization. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 2492–2500).

Brachmann, E. & Rother, C. (2018). Learning less is more - 6d camera localization via 3d surface regression. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (pp. 4654–4662).

Brachmann, E. & Rother, C. (2019a). Expert sample consensus applied to camera re-localization. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, (pp. 7524–7533).

Brachmann, E. & Rother, C. (2019b). Neural- Guided RANSAC: Learning Where to Sample Model Hypotheses. In *ICCV*.

Brachmann, E. & Rother, C. (2021). Visual camera re-localization from rgb and rgb-d images using dsac. *IEEE transactions on pattern analysis and machine intelligence*, PP.

Bretzner, L. & Lindeberg, T. (1998). Feature tracking with automatic selection of spatial scales. *Comput. Vis. Image Underst.*, 71, 385–392.

Bui, M., Albarqouni, S., Ilic, S., & Navab, N. (2018). Scene coordinate and correspondence learning for image-based localization. In *BMVC*.

Bujnak, M., Kukelova, Z., & Pajdla, T. (2010). New efficient solution to the absolute pose problem for camera with unknown focal length and radial distortion. In *ACCV*.

Cai, R., Hariharan, B., Snavely, N., & Averbuch-Elor, H. (2021). Extreme Rotation Estimation using Dense Correlation Volumes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Calonder, M., Lepetit, V., Strecha, C., & Fua, P. (2010). Brief: Binary robust independent elementary features. In *ECCV*.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-end object detection with transformers. *ArXiv*, abs/2005.12872.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., & Joulin, A. (2021). Emerging Properties in Self-Supervised Vision Transformers. In *ArXiv*.

Cavallari, T., Bertinetto, L., Mukhoti, J., Torr, P. H. S., & Golodetz, S. (2019). Let's take this online: Adapting scene coordinate regression network predictions for online rgb-d camera relocalisation. *2019 International Conference on 3D Vision (3DV)*, (pp. 564–573).

Chapelle, O. & Wu, M. (2009). Gradient descent optimization of smoothed information retrieval metrics. *Information Retrieval*, 13, 216–235.

Chen, D. M., Baatz, G., Köser, K., Tsai, S. S., Vedantham, R., Pylvänäinen, T., Roimela, K., Chen, X., Bach, J., Pollefeys, M., Girod, B., & Grzeszczuk, R. (2011). City-scale landmark identification on mobile devices. *CVPR 2011*, (pp. 737–744).

Chen, K., Snavely, N., & Makadia, A. (2021). Wide-baseline relative camera pose estimation with directional learning. In *CVPR*.

Choudhary, S. & Narayanan, P. (2012). Visibility probability structure from sfm datasets and applications. In *ECCV*.

Choy, C., Lee, J., Ranftl, R., Park, J., & Koltun, V. (2020). High-dimensional convolutional networks for geometric pattern recognition. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 11224–11233).

Choy, C. B., Gwak, J., Savarese, S., & Chandraker, M. (2016). Universal correspondence network. In *NIPS*.

Chum, O., Chin, T.-J., Ranftl, R., Mishkin, D., Matas, J., & Barath, D. (2020). RANSAC in 2020: A CVPR Tutorial.

Chum, O. & Matas, J. (2005). Matching with prosac - progressive sample consensus. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 1, 220–226 vol. 1.

Chum, O., Matas, J., & Kittler, J. (2003). Locally Optimized RANSAC. In *DAGM-Symposium*.

Chum, O., Werner, T., & Matas, J. (2005). Two-view geometry estimation unaffected by a dominant plane. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 1, 772–779 vol. 1.

Clark, R., Bloesch, M., Czarnowski, J., Leutenegger, S., & Davison, A. (2018). Ls-net: Learning to solve nonlinear least squares for monocular stereo. *ArXiv*, abs/1809.02966.

Cohen, A. & Zach, C. (2015). The likelihood-ratio test and efficient robust estimation. *2015 IEEE International Conference on Computer Vision (ICCV)*, (pp. 2282–2290).

Cordonnier, J.-B., Loukas, A., & Jaggi, M. (2020). On the Relationship Between Self-Attention and Convolutional Layers. In *ArXiv*.

Csorba, M. (1997). Simultaneous localisation and map building.

Csurka, G. & Humenberger, M. (2018). From Handcrafted to Deep Local Invariant Features. In *Computing Research Repository*.

Cummins, M. & Newman, P. (2008). Fab-map: Probabilistic localization and mapping in the space of appearance. *The International Journal of Robotics Research*, 27, 647 – 665.

Czarnowski, J., Leutenegger, S., & Davison, A. (2017). Semantic texture for robust dense tracking. *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, (pp. 851–859).

Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T., & Nießner, M. (2017). Scannet: Richly-annotated 3d reconstructions of indoor scenes. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 2432–2443).

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *CVPR*.

Detone, D., Malisiewicz, T., & Rabinovich, A. (2018). Superpoint: Self-Supervised Interest Point Detection and Description. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Dias, P., Kassim, A., & Srinivasan, V. (1995). A neural network based corner detection method. *Proceedings of ICNN'95 - International Conference on Neural Networks*, 4, 2116–2120 vol.4.

Ding, M., Wang, Z., Sun, J., Shi, J., & Luo, P. (2019). Camnet: Coarse-to-fine retrieval for camera re-localization. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, (pp. 2871–2880).

Donoser, M. & Schmalstieg, D. (2014). Discriminative feature-to-point matching in image-based localization. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 516–523).

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2020). An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ArXiv*.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929.

Dreyfus, S. (1962). The numerical solution of variational problems. *Journal of Mathematical Analysis and Applications*, 5, 30–45.

Duchi, J. C., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. In *J. Mach. Learn. Res.*

Durrant-Whyte, H. & Bailey, T. (2006). Simultaneous localization and mapping: part i. *IEEE Robotics & Automation Magazine*, 13, 99–110.

Durrant-Whyte, H., Rye, D., & Nebot, E. (1996). Localization of autonomous guided vehicles.

Dusmanu, M., Rocco, I., Pajdla, T., Pollefeys, M., Sivic, J., Torii, A., & Sattler, T. (2019). D2-net: A trainable cnn for joint description and detection of local features. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 8084–8093).

Dusmanu, M., Schönberger, J. L., & Pollefeys, M. (2020). Multi-view optimization of local feature geometry. In *ECCV*.

Engel, J., Koltun, V., & Cremers, D. (2018). Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40, 611–625.

Engel, J., Schöps, T., & Cremers, D. (2014). Lsd-slam: Large-scale direct monocular slam. In *ECCV*.

Fiore, P. (2001). Efficient linear solution of exterior orientation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23, 140–148.

Fischler, M. A. & Bolles, R. C. (1981). Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM*, 24.

Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36, 193–202.

Gauglitz, S., Höllerer, T., & Turk, M. (2011). Evaluation of Interest Point Detectors and Feature Descriptors for Visual Tracking. *IJCV*, 94, 335–360.

Germain, H., Bourmaud, G., & Lepetit, V. (2018). Improving nighttime retrieval-based localization. In *arXiv Preprint*.

Germain, H., Bourmaud, G., & Lepetit, V. (2019). Sparse-To-Dense Hypercolumn Matching for Long-Term Visual Localization. In *International Conference on 3D Vision (3DV)*.

Germain, H., Bourmaud, G., & Lepetit, V. (2020). S2DNet: Learning Image Features for Accurate Sparse-to-Dense Matching. In *European Conference on Computer Vision (ECCV)*.

Germain, H., Lepetit, V., & Bourmaud, G. (2021a). Neural reprojection error: Merging feature learning and camera pose estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 414–423).

Germain, H., Lepetit, V., & Bourmaud, G. (2021b). Visual correspondence hallucination: Towards geometric reasoning. In *arXiv Preprint*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. C., & Bengio, Y. (2014). Generative adversarial nets. In *NIPS*.

Gordo, A., Almazán, J., Revaud, J., & Larlus, D. (2016a). Deep image retrieval: Learning global representations for image search. In *ECCV*.

Gordo, A., Almazán, J., Revaud, J., & Larlus, D. (2016b). Deep Image Retrieval: Learning Global Representations for Image Search. In *European Conference on Computer Vision (ECCV)*.

Gordo, A., Almazán, J., Revaud, J., & Larlus, D. (2017). End-to-end learning of deep visual representations for image retrieval. *International Journal of Computer Vision*, 124, 237–254.

Haralick, R., Lee, C.-N., Ottenburg, K., & Nölle, M. (1991). Analysis and solutions of the three point perspective pose estimation problem. *Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, (pp. 592–598).

Haralick, R. M., Lee, C.-N., Ottenberg, K., & Nölle, M. (1994). Review and Analysis of Solutions of the Three Point Perspective Pose Estimation Problem. *IJCV*, 13.

Hariharan, B., Arbeláez, P. A., Girshick, R. B., & Malik, J. (2014). Hypercolumns for Object Segmentation and Fine-Grained Localization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

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

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 770–778).

He, K., Çakir, F., Bargal, S. A., & Sclaroff, S. (2018). Hashing as tie-aware learning to rank. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (pp. 4023–4032).

Heinly, J., Schonberger, J., Dunn, E., & Frahm, J.-M. (2015). Reconstructing the world* in six days *(as captured by the yahoo 100 million image dataset). In *CVPR 2015*.

Heisterklaus, I., Qian, N., & Miller, A. (2014). Image-based pose estimation using a compact 3d model. *2014 IEEE Fourth International Conference on Consumer Electronics Berlin (ICCE-Berlin)*, (pp. 327–330).

Huber, P. (1964). Robust estimation of a location parameter. *Annals of Mathematical Statistics*, 35, 492–518.

Ioffe, S. & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ArXiv*, abs/1502.03167.

Irschara, A., Zach, C., Frahm, J.-M., & Bischof, H. (2009). From structure-from-motion point clouds to fast location recognition. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 2599–2606).

Ishimura, N., Hashimoto, T., Tsujimoto, S., & Arimoto, S. (1986). Spline approximation of line images by modified dynamic programming. *Systems and Computers in Japan*, 17, 21–29.

Ivakhnenko, A. & Lapa, V. G. (1966). Cybernetic predicting devices.

Jégou, H. & Chum, O. (2012). Negative evidences and co-occurences in image retrieval: The benefit of pca and whitening. In *ECCV*.

Jégou, H., Douze, M., & Schmid, C. (2011). Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33, 117–128.

Jin, L., Qian, S., Owens, A., & Fouhey, D. F. (2021a). Planar surface reconstruction from sparse views. *ArXiv*, abs/2103.14644.

Jin, Y., Mishkin, D., Mishchuk, A., Matas, J., Fua, P., Yi, K. M., & Trulls, E. (2021b). Image matching across wide baselines: From paper to practice. *International Journal of Computer Vision*, 129, 517–547.

Kalantidis, Y., Mellina, C., & Osindero, S. (2016). Cross-dimensional weighting for aggregated deep convolutional features. *ArXiv*, abs/1512.04065.

Katharopoulos, A., Vyas, A., Pappas, N., & Fleuret, F. (2020). Transformers are rnns: Fast autoregressive transformers with linear attention. In *ICML*.

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

Kendall, A., Grimes, M., & Cipolla, R. (2015). Posenet: A convolutional network for real-time 6-dof camera relocalization. *2015 IEEE International Conference on Computer Vision (ICCV)*, (pp. 2938–2946).

Kingma, D. P. & Ba, J. (2015). Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Kitchen, L. & Rosenfeld, A. (1982). Gray-level corner detection. *Pattern Recognit. Lett.*, 1, 95–102.

Kneip, L., Scaramuzza, D., & Siegwart, R. (2011). A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. *CVPR 2011*, (pp. 2969–2976).

Kobyshev, N., Riemenschneider, H., & Gool, L. (2014). Matching features correctly through semantic understanding. *2014 2nd International Conference on 3D Vision*, 1, 472–479.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60, 84 – 90.

Kukelova, Z., Bujnak, M., & Pajdla, T. (2013). Real-time solution to the absolute pose problem with unknown radial distortion and focal length. *2013 IEEE International Conference on Computer Vision*, (pp. 2816–2823).

Larsson, V., Fredriksson, J., Toft, C., & Kahl, F. (2016). Outlier rejection for absolute pose estimation with known orientation. In *BMVC*.

Laskar, Z., Melekhov, I., Kalia, S., & Kannala, J. (2017). Camera relocalization by computing pairwise relative poses using convolutional neural network. *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, (pp. 920–929).

Lebeda, K., Matas, J. E. S., & Chum, O. (2012). Fixing the Locally Optimized RANSAC. In *BMVC*.

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

Lepetit, V., Moreno-Noguer, F., & Fua, P. (2008). Epnp: An accurate o(n) solution to the pnp problem. *International Journal of Computer Vision*, 81, 155–166.

Li, X., Han, K., Li, S., & Prisacariu, V. (2020a). Dual-Resolution Correspondence Networks. *Advances in Neural Information Processing Systems*, 33.

Li, X., Wang, S., Zhao, Y., Verbeek, J., & Kannala, J. (2020b). Hierarchical scene coordinate classification and regression for visual localization. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 11980–11989).

Li, Y., Snavely, N., & Huttenlocher, D. (2010). Location recognition using prioritized feature matching. In *ECCV*.

Li, Y., Snavely, N., Huttenlocher, D., & Fua, P. (2016). Worldwide pose estimation using 3d point clouds. In *Large-Scale Visual Geo-Localization*.

Li, Z. & Snavely, N. (2018). Megadepth: Learning single-view depth prediction from internet photos. *CVPR*, (pp. 2041–2050).

Lim, H., Sinha, S. N., Cohen, M. F., & Uyttendaele, M. (2012). Real-time image-based 6-dof localization in large-scale environments. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 1043–1050).

Lindeberg, T. (1993). Detecting salient blob-like image structures and their scales with a scale-space primal sketch: A method for focus-of-attention. *International Journal of Computer Vision*, 11, 283–318.

Lindeberg, T. (1998). Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30, 79–116.

Lindenberger, P., Sarlin, P.-E., Larsson, V., & Pollefeys, M. (2021). Pixel-Perfect Structure-from-Motion with Featuremetric Refinement. In *ICCV*.

Loshchilov, I. & Hutter, F. (2019). Decoupled weight decay regularization. In *ICLR*.

Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 60(2).

Lucas, B. D. & Kanade, T. (1981). An iterative image registration technique with an application to stereo vision. In *IJCAI*.

Luo, Z., Shen, T., Zhou, L., Zhang, J., Yao, Y., Li, S., Fang, T., & Quan, L. (2019). ContextDesc: Local Descriptor Augmentation with Cross-Modality Context. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2522–2531).

Luong, Q.-T. & Faugeras, O. D. (1996). The Fundamental Matrix: Theory, Algorithms, and Stability Analysis. *IJCV*, 17(1), 43–75.

Lv, Z., Dellaert, F., Rehg, J. M., & Geiger, A. (2019). Taking a Deeper Look at the Inverse Compositional Algorithm. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 4581–4590).

Lynen, S., Sattler, T., Bosse, M., Hesch, J., Pollefeys, M., & Siegwart, R. (2015). Get out of my lab: Large-scale, real-time visual-inertial localization. In *Robotics: Science and Systems*.

Ma, W.-C., Wang, S., Gu, J.-Y., Manivasagam, S., Torralba, A., & Urtasun, R. (2020). Deep feedback inverse problem solver. *ArXiv*, abs/2101.07719.

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

Middelberg, S., Sattler, T., Untzelmann, O., & Kobbelt, L. (2014). Scalable 6-DOF Localization on Mobile Devices. In *European Conference on Computer Vision (ECCV)*.

Mikolajczyk, K. & Schmid, C. (2001). Indexing based on scale invariant interest points. *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, 1, 525–531 vol.1.

Mikolajczyk, K. & Schmid, C. (2002). An affine invariant interest point detector. In *ECCV*.

Mikolajczyk, K. & Schmid, C. (2004). Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 60, 63–86.

Minsky, M. & Papert, S. (1969). Perceptrons - an introduction to computational geometry.

Mishchuk, A., Mishkin, D., Radenović, F., & Matas, J. (2017). Working hard to know your neighbor's margins: Local descriptor learning loss. In *NIPS*.

Mishkin, D., Radenović, F., & Matas, J. (2018). Repeatability is not enough: Learning affine regions via discriminability. In *ECCV*.

Mobahi, H. & Fisher, J. W. (2015). On the Link Between Gaussian Homotopy Continuation and Convex Envelopes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 43–56).

Moo yi, K., Trulls, E., Ono, Y., Lepetit, V., Salzmann, M., & Fua, P. (2018). Learning to Find Good Correspondences. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2666–2674).

Mur-Artal, R., Montiel, J., & Tardós, J. D. (2015). Orb-slam: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31, 1147–1163.

Nathan Silberman, Derek Hoiem, P. K. & Fergus, R. (2012). Indoor segmentation and support inference from rgbd images. In *ECCV*.

Nesterov, Y. (1983). A method for unconstrained convex minimization problem with the rate of convergence $o(1/k^2)$.

Ng, T., Balntas, V., Tian, Y., & Mikolajczyk, K. (2020). Solar: Second-order loss and attention for image retrieval. *ArXiv*, abs/2001.08972.

Noh, H., Araújo, A., Sim, J., Weyand, T., & Han, B. (2017). Large-scale image retrieval with attentive deep local features. *2017 IEEE International Conference on Computer Vision (ICCV)*, (pp. 3476–3485).

Ono, Y., Trulls, E., Fua, P., & Yi, K. M. (2018). Lf-net: Learning local features from images. In *NeurIPS*.

Park, S., Schöps, T., & Pollefeys, M. (2017). Illumination change robustness in direct visual slam. *2017 IEEE International Conference on Robotics and Automation (ICRA)*, (pp. 4523–4530).

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., Devito, Z., Lin, Z., Desmaison, A., Antiga, L., & Lerer, A. (2017). Automatic Differentiation in Pytorch. In *NeurIPS*.

Perdoch, M., Chum, O., & Matas, J. (2009). Efficient Representation of Local Geometry for Large Scale Object Retrieval. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Philbin, J., Chum, O., Isard, M., Sivic, J., & Zisserman, A. (2007). Object Retrieval with Large Vocabularies and Fast Spatial Matching. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Piasco, N., Sidibé, D., Gouet-Brunet, V., & Demonceaux, C. (2019). Learning scene geometry for visual localization in challenging conditions. *2019 International Conference on Robotics and Automation (ICRA)*, (pp. 9094–9100).

Qian, S., Jin, L., & Fouhey, D. F. (2020). Associative3d: Volumetric reconstruction from sparse views. In *ECCV*.

Quan, L. & Lan, Z. (1999). Linear n-point camera pose determination. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21, 774–780.

Radenovic, F., Iscen, A., Tolias, G., Avrithis, Y. S., & Chum, O. (2018a). Revisiting Oxford and Paris: Large-Scale Image Retrieval Benchmarking. *CoRR*, abs/1803.11285.

Radenović, F., Tolias, G., & Chum, O. (2016). CNN Image Retrieval Learns from BoW: Unsupervised Fine-Tuning with Hard Examples. In *European Conference on Computer Vision (ECCV)*.

Radenovic, F., Tolias, G., & Chum, O. (2018b). Fine-Tuning CNN Image Retrieval with No Human Annotation. *IEEE TPAMI*.

Ramachandran, P., Parmar, N., Vaswani, A., Bello, I., Levskaya, A., & Shlens, J. (2019). Stand-Alone Self-Attention In Vision Models. In *NeurIPS*.

Ranftl, R. & Koltun, V. (2018). Deep fundamental matrix estimation. In *ECCV*.

Razavian, A. S., Sullivan, J., Maki, A., & Carlsson, S. (2014). Visual Instance Retrieval with Deep Convolutional Networks. *CoRR*, abs/1412.6574.

Revaud, J., De Souza, C., Humenberger, M., & Weinzaepfel, P. (2019). R2d2: Reliable and repeatable detector and descriptor. In *NeurIPS*, volume 32 (pp. 12405–12415).: Curran Associates, Inc.

Rocco, I., Arandjelović, R., & Sivic, J. (2020). Efficient Neighbourhood Consensus Networks via Submanifold Sparse Convolutions. *IEEE TPAMI*.

Rocco, I., Cimpoi, M., Arandjelović, R., Torii, A., Pajdla, T., & Sivic, J. (2018). Neighbourhood Consensus Networks. In *NeurIPS*.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*.

Rosenblatt, F. (1957). *The perceptron - A perceiving and recognizing automaton*. Technical Report 85-460-1, Cornell Aeronautical Laboratory, Ithaca, New York.

Rosten, E. & Drummond, T. (2006). Machine learning for high-speed corner detection. In *ECCV*.

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. R. (2011). ORB: An Efficient Alternative to SIFT or SURF. In *ICCV*.

Rumelhart, D., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323, 533–536.

Rutkowski, W. S. & Rosenfeld, A. (1978). *A comparison of corner detection techniques for chain coded curves*. Technical Report 623, Maryland University.

Salahat, E. & Qasaimeh, M. (2017). Recent Advances in Features Extraction and Description Algorithms: A Comprehensive Survey. In *2017 IEEE International Conference on Industrial Technology (ICIT)* (pp. 1059–1063).

Sarlin, P.-E., Cadena, C., Siegwart, R., & Dymczyk, M. (2019). From Coarse to Fine: Robust Hierarchical Localization at Large Scale. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 12716–12725).

Sarlin, P.-E., Debraine, F., Dymczyk, M., Siegwart, R., & Cadena, C. (2018). Leveraging deep visual descriptors for hierarchical efficient localization. *ArXiv*, abs/1809.01019.

Sarlin, P.-E., Detone, D., Malisiewicz, T., & Rabinovich, A. (2020). SuperGlue: Learning Feature Matching with Graph Neural Networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sarlin, P.-E., Unagar, A., Larsson, M., Germain, H., Toft, C., Larsson, V., Pollefeys, M., Lepetit, V., Hammarstrand, L., Kahl, F., & Sattler, T. (2021). Back to the Feature: Learning robust camera localization from pixels to pose. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sattler, T., Havlena, M., Radenovic, F., Schindler, K., & Pollefeys, M. (2015). Hyperpoints and Fine Vocabularies for Large-Scale Location Recognition. In *ICCV*.

Sattler, T., Havlena, M., Schindler, K., & Pollefeys, M. (2016). Large-Scale Location Recognition and the Geometric Burstiness Problem. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sattler, T., Leibe, B., & Kobbelt, L. (2011). Fast image-based localization using direct 2d-to-3d matching. *2011 International Conference on Computer Vision*, (pp. 667–674).

Sattler, T., Leibe, B., & Kobbelt, L. (2017a). Efficient & effective prioritized matching for large-scale image-based localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39, 1744–1756.

Sattler, T., Maddern, W. P., Toft, C., Torii, A., Hammarstrand, L., Stenborg, E., Safari, D., Okutomi, M., Pollefeys, M., Sivic, J., Kahl, F., & Pajdla, T. (2018). Benchmarking 6dof outdoor visual localization in changing conditions. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (pp. 8601–8610).

Sattler, T., Sweeney, C., & Pollefeys, M. (2014). On Sampling Focal Length Values to Solve the Absolute Pose Problem. In *European Conference on Computer Vision (ECCV)*.

Sattler, T., Torii, A., Sivic, J., Pollefeys, M., Taira, H., Okutomi, M., & Pajdla, T. (2017b). Are Large-Scale 3D Models Really Necessary for Accurate Visual Localization? In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sattler, T., Weyand, T., Leibe, B., & Kobbelt, L. (2012). Image Retrieval for Image-Based Localization Revisited. In *BMVC*.

Sattler, T., Zhou, Q., Pollefeys, M., & Leal-Taixé, L. (2019). Understanding the limitations of cnn-based absolute camera pose regression. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 3297–3307).

Schönberger, J. L. & Frahm, J.-M. (2016). Structure-From-Motion Revisited. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Schönberger, J. L., Pollefeys, M., Geiger, A., & Sattler, T. (2017). Semantic Visual Localization. *CoRR*, abs/1712.05773.

Schönberger, J. L., Zheng, E., Pollefeys, M., & Frahm, J.-M. (2016). Pixelwise View Selection for Unstructured Multi-View Stereo. In *European Conference on Computer Vision (ECCV)*.

Schöps, T., Schönberger, J. L., Galliani, S., Sattler, T., Schindler, K., Pollefeys, M., & Geiger, A. (2017). A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Schroff, F., Kalenichenko, D., & Philbin, J. (2015). Facenet: A Unified EMbedding for Face Recognition and Clustering. *CoRR*, abs/1503.03832.

Shen, X., Darmon, F., Efros, A. A., & Aubry, M. (2020). Ransac-flow: generic two-stage image alignment. In *ECCV*.

Simo-Serra, E., Trulls, E., Ferraz, L., Kokkinos, I., Fua, P., & Moreno-Noguer, F. (2015). Discriminative Learning of Deep Convolutional Feature Point Descriptors. In *ICCV*.

Simonyan, K., Vedaldi, A., & Zisserman, A. (2014). Learning Local Feature Descriptors Using Convex Optimisation. *IEEE TPAMI*, 36.

Simonyan, K. & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR*, abs/1409.1556.

Singh, G. & Kosecka, J. (2016). *Semantically Guided Geo-Location and Modeling in Urban Environments*.

Smith, L. N. (2017). Cyclical learning rates for training neural networks. *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, (pp. 464–472).

Sochman, J. & Matas, J. (2007). Learning a fast emulator of a binary decision process. In *ACCV*.

Stumberg, L., Wenzel, P., Yang, N., & Cremers, D. (2020). Lm-reloc: Levenberg-marquardt based direct visual relocalization. *2020 International Conference on 3D Vision (3DV)*, (pp. 968–977).

Sun, J., Shen, Z., Wang, Y., Bao, H., & Zhou, X. (2021). Loftr: Detector-free local feature matching with transformers. *ArXiv*, abs/2104.00680.

Sun, W., Jiang, W., Trulls, E., Tagliasacchi, A., & Yi, K. M. (2020). ACNe: Attentive Context Normalization for Robust Permutation-Equivariant Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 11286–11295).

Svärm, L., Enqvist, O., Kahl, F., & Oskarsson, M. (2017). City-scale localization for cameras with known vertical direction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39, 1455–1461.

Svärm, L., Enqvist, O., Oskarsson, M., & Kahl, F. (2014). Accurate Localization and Pose Estimation for Large 3D Models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sweeney, C., Fragoso, V., Höllerer, T., & Turk, M. (2016). Large Scale SfM with the Distributed Camera Model. In *International Conference on 3D Vision (3DV)*.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 1–9).

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016a). Rethinking the inception architecture for computer vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 2818–2826).

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016b). Rethinking the inception architecture for computer vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 2818–2826).

Szeliski, R. (2004). Image alignment and stitching : A tutorial 1.

Taira, H., Okutomi, M., Sattler, T., Cimpoi, M., Pollefeys, M., Sivic, J., Pajdla, T., & Torii, A. (2018). Inloc: Indoor Visual Localization with Dense Matching and View Synthesis. *CoRR*, abs/1803.10368.

Taira, H., Rocco, I., Sedlár, J., Okutomi, M., Sivic, J., Pajdla, T., Sattler, T., & Torii, A. (2019). Is this the right place? geometric-semantic pose verification for indoor visual localization. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, (pp. 4372–4382).

Tang, C. & Tan, P. (2019). BA-Net: Dense Bundle Adjustment Network. In *ICLR*.

Tareen, S. A. K. & Saleem, Z. (2018). A comparative analysis of sift, surf, kaze, akaze, orb, and brisk. *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, (pp. 1–10).

Thomee, B., Shamma, D. A., Friedland, G., Elizalde, B., Ni, K., Poland, D., Borth, D., & Li, L. (2016). YFCC100M: The New Data in Multimedia Research. *Commun. ACM*, 59.

Tian, Y., Balntas, V., Ng, T., Laguna, A. B., Demiris, Y., & Mikolajczyk, K. (2020a). D2d: Keypoint extraction with describe to detect approach. In *ACCV*.

Tian, Y., Fan, B., & Wu, F. (2017). L2-net: Deep learning of discriminative patch descriptor in euclidean space. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 6128–6136).

Tian, Y., Laguna, A. B., Ng, T., Balntas, V., & Mikolajczyk, K. (2020b). Hynet: Learning local descriptor with hybrid similarity measure and triplet loss. *arXiv: Computer Vision and Pattern Recognition*.

Tian, Y., Yu, X., Fan, B., Wu, F., Heijnen, H., & Balntas, V. (2019). SOSNet: Second Order Similarity Regularization for Local Descriptor Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Tieleman, T. & Hinton, G. (2012). Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning.

Toft, C., Stenborg, E., Hammarstrand, L., Brynte, L., Pollefeys, M., Sattler, T., & Kahl, F. (2018). Semantic Match Consistency for Long-Term Visual Localization. In *European Conference on Computer Vision (ECCV)*.

Tolias, G., Sicre, R., & Jégou, H. (2015). Particular Object Retrieval with Integral Max-Pooling of CNN Activations. *CoRR*, abs/1511.05879.

Torii, A., Arandjelović, R., Sivic, J., Okutomi, M., & Pajdla, T. (2015). 24/7 place recognition by view synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40, 257–271.

Torr, P. H. & Zisserman, A. (2000). MLESAC: A New Robust Estimator with Application to Estimating Image Geometry. *Computer Vision and Image Understanding*, 78(1), 138–156.

Tran, N.-T., Tan, D.-K. L., Doan, A.-D., Do, T.-T., Bui, T.-A., Tan, M., & Cheung, N.-M. (2019). On-device scalable image-based localization via prioritized cascade search and fast one-many ransac. *IEEE Transactions on Image Processing*, 28, 1675–1690.

Triggs, B., McLauchlan, P., Hartley, R., & Fitzgibbon, A. (1999). Bundle adjustment - a modern synthesis. In *Workshop on Vision Algorithms*.

Trulls, E., Jin, Y., Yi, K., Mishking, D., Matas, J., & Fua, P. (2019). Phototourism Challenge, CVPR 2019 Image Matching Workshop.

Truong, P., Danelljan, M., Gool, L., & Timofte, R. (2020a). Gocor: Bringing globally optimized correspondence volumes into your neural network. *ArXiv*, abs/2009.07823.

Truong, P., Danelljan, M., Gool, L., & Timofte, R. (2021). Learning accurate dense correspondences and when to trust them. *ArXiv*, abs/2101.01710.

Truong, P., Danelljan, M., & Timofte, R. (2020b). Glu-net: Global-local universal network for dense flow and correspondences. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 6257–6267).

Tuytelaars, T. & Mikolajczyk, K. (2008).

Tyszkiewicz, M., Fua, P., & Trulls, E. (2020). DISK: Learning Local Features with Policy Gradient. In *NeurIPS*.

Vaswani, A., Shazeer, N. M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *ArXiv*, abs/1706.03762.

Verdie, Y., Yi, K. M., Fua, P., & Lepetit, V. (2015). Tilde: A temporally invariant learned detector. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 5279–5288).

von Stumberg, L., Wenzel, P., Khan, Q., & Cremers, D. (2020). Gn-net: The gauss-newton loss for multi-weather relocalization. *IEEE Robotics and Automation Letters*, 5, 890–897.

Von Stumberg, L., Wenzel, P., Khan, Q., & Cremers, D. (2020). GN-Net: The Gauss-Newton Loss for Multi-Weather Relocalization. *IEEE Robotics and Automation Letters*, 5(2), 890–897.

Walch, F., Hazirbas, C., Leal-Taixé, L., Sattler, T., Hilsenbeck, S., & Cremers, D. (2017). Image-Based Localization Using LSTMs for Structured Feature Correlation. In *ICCV*.

Wang, C., Galoogahi, H. K., Lin, C.-H., & Lucey, S. (2018). Deep-lk for efficient adaptive object tracking. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, (pp. 627–634).

Wijmans, E. & Furukawa, Y. (2016). Exploiting 2D Floorplan for Building-Scale Panorama RGBD Alignment. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wrobel, B. (2001). Minimum solutions for orientation.

Xu, B., Davison, A., & Leutenegger, S. (2021). Deep probabilistic feature-metric tracking. *IEEE Robotics and Automation Letters*, 6, 223–230.

Yang, L., Bai, Z., Tang, C., Li, H., Furukawa, Y., & Tan, P. (2019a). Sanet: Scene agnostic network for camera localization. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, (pp. 42–51).

Yang, Z., Dan, T., & Yang, Y. (2018). Multi-temporal remote sensing image registration using deep convolutional features. *IEEE Access*, 6, 38544–38555.

Yang, Z., Pan, J. Z., Luo, L., Zhou, X., Grauman, K., & Huang, Q. (2019b). Extreme relative pose estimation for rgb-d scans via scene completion. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 4526–4535).

Yang, Z., Yan, S., & Huang, Q.-X. (2020). Extreme relative pose network under hybrid representations. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 2452–2461).

Yi, K. M., Trulls, E., Lepetit, V., & Fua, P. (2016). LIFT: Learned Invariant Feature Transform. In *European Conference on Computer Vision (ECCV)*.

Zach, C. & Bourmaud, G. (2017). Iterated Lifting for Robust Cost Optimization. In *BMVC*.

Zach, C. & Bourmaud, G. (2018). Descending, Lifting or Smoothing: Secrets of Robust Cost Optimization. In *European Conference on Computer Vision (ECCV)* (pp. 547–562).

Zamir, A. R. & Shah, M. (2010). Accurate Image Localization Based on Google Maps Street View. In *European Conference on Computer Vision (ECCV)*.

Zeisl, B., Sattler, T., & Pollefeys, M. (2015). Camera Pose Voting for Large-Scale Image-Based Localization. In *ICCV*.

Zhang, J., Kaess, M., & Singh, S. (2014). Real-time depth enhanced monocular odometry. *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, (pp. 4973–4980).

Zhang, J. & Singh, S. (2014). Loam: Lidar odometry and mapping in real-time. In *Robotics: Science and Systems*.

Zhang, J., Sun, D., Luo, Z., Yao, A., Zhou, L., Shen, T., Chen, Y., Quan, L., & Liao, H. (2019). Learning Two-View Correspondences and Geometry Using Order-Aware Network. In *ICCV*.

Zhang, W. & Kosecka, J. (2006). Image Based Localization in Urban Environments. *International Symposium on 3D Data Processing, Visualization, and Transmission*.

Zhang, Z., Sattler, T., & Scaramuzza, D. (2021). Reference pose generation for long-term visual localization via learned features and view synthesis. *Int. J. Comput. Vis.*, 129, 821–844.

Zhao, H., Jia, J., & Koltun, V. (2020). Exploring Self-Attention for Image Recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 10073–10082).

Zhou, Q., Sattler, T., & Leal-Taixé, L. (2020a). Patch2pix: Epipolar-guided pixel-level correspondences. *ArXiv*, abs/2012.01909.

Zhou, Q., Sattler, T., Pollefeys, M., & Leal-Taixé, L. (2020b). To learn or not to learn: Visual localization from essential matrices. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, (pp. 3319–3326).

Zitová, B. & Flusser, J. (2003). Image registration methods: a survey. *Image Vis. Comput.*, 21, 977–1000.