Neural architectures for estimating correspondences between images

Ignacio Rocco

PSL UNIVERSITÉ PARIS

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à l'École Normale Supérieure

Neural architectures for estimating correspondences between images

Soutenue par

Ignacio Rocco

Le 27 octobre 2020

École doctorale n°386

Sciences Mathématiques de Paris Centre

Spécialité

Informatique

Composition du jury :

Vincent Lepetit École des Ponts ParisTech	*Président du jury*
Andrea Vedaldi University of Oxford	*Rapporteur*
Diane Larlus NAVER LABS Europe	*Examinateur*
Patrick Pérez Valeo.ai	*Examinateur*
Josef Sivic École Normale Supérieure, Inria	*Directeur de thèse*
Relja Arandjelović DeepMind	*Codirecteur de thèse*

ENS | PSL

Résumé

L'objectif de cette thèse est de développer des méthodes pour la mise en correspondance entre de paires d'images dans des situations difficiles, telles que (i) des changements extrêmes d'éclairage, comme pour le scénario d'appariement jour-nuit, (ii) l'appariement de scènes peu texturées ou avec des structures répétitives, comme c'est souvent le cas pour les scènes d'intérieur, (iii) l'appariement sur de longues échelles de temps (plusieurs mois ou années), où certains éléments structurels peuvent avoir été modifiés et (iv) l'appariement entre des parties d'objets appartenant à la même classe, mais qui peuvent présenter de grandes différences d'apparence intra-classe. Les principales contributions de cette thèse sont les suivantes. Tout d'abord, nous développons une approche entrainable pour l'alignement paramétrique d'images en utilisant un modèle de réseau siamois. Ce modèle prend deux images en entrée et peut estimer les paramètres d'une transformation géométrique telle qu'une transformation affine, une homographie ou une transformation en spline. Le modèle proposé contient trois modules distincts pour l'extraction de caractéristiques, l'appariement et la régression des paramètres. Ces modules sont composés d'opérations différentiables, permettant un entraînement de bout-en-bout. Deuxièmement, afin de permettre l'entraînement à partir de paires d'images réelles sans annotation de correspondance, nous développons un module inspiré de la mesure utilisée par l'algorithme RANSAC, mais implémentée de manière différentiable afin que le modèle soit entraîné en utilisant la rétropropagation standard. Nous montrons que notre approche faiblement supervisée peut fournir un gain de performance significatif par rapport à l'entraînement uniquement avec des images déformées synthétiquement. Troisièmement, nous développons les Réseaux de Consensus de Voisinage qui peuvent être utilisés pour estimer de manière

robuste les correspondances pour des tâches où des correspondances discrètes sont nécessaires. Ces modèles opèrent dans l'espace 4D des correspondances d'images et peuvent être entrainés pour identifier des groupes cohérents de correspondances afin de lever l'ambiguïté de correspondances difficiles. Enfin, comme la formulation dense des Réseaux de Consensus de Voisinage est gourmande en mémoire et en calcul, limitant leur mise en application pratique, nous développons une variante plus efficace qui peut réduire considérablement les besoins en mémoire et le temps d'exécution. Cette formulation efficace utilise un tenseur de corrélation parcimonieux pour stocker les correspondances provisoires, qui est traité par un réseau de neurones convolutifs parcimonieux 4D pouvant filtrer les correspondances incorrectes. Nos méthodes obtiennent des résultats état de l'art pour l'appariement d'images au niveau de catégories dans les benchmarks PF-WILLOW, PF-PASCAL, TSS et Caltech-101, ainsi que pour l'appariement au niveau de l'instance dans les benchmarks HPatches-Sequences, InLoc et Aachen Day-Night.

Abstract

The goal of this thesis is to develop methods for establishing correspondences between pairs of images in challenging situations, such as (i) extreme illumination changes, as in the day-night matching scenario, (ii) matching scenes with little texture or with repetitive structures, as is frequently the case for indoor scenes, (iii) handling matching of scenes across long time-scales (*e.g.* years), where some structural elements may have been modified and (iv) matching parts of objects which belong to the same class, but which may have large intra-class appearance differences. The key contributions of this thesis are the following. First, we propose a trainable approach for parametric image alignment employing a Siamese network model. This model processes two input images and can estimate the parameters of a geometric transformation such as affine, homography or thin-plate spline. The proposed model contains three distinct modules for feature extraction, feature matching and parameter regression. These modules are implemented using differentiable operations, which results in an end-to-end trainable architecture. Second, to allow training from real image pairs lacking correspondence annotation we develop a *soft-inlier count* module. This soft-inlier count module is inspired by the *inlier count* measure used in RANSAC, but implemented in a differentiable way to allow training using standard backpropagation. We show that the proposed weakly-supervised approach can provide a significant performance gain compared to training solely with synthetically warped images. Third, we develop Neighbourhood Consensus Networks which robustly estimate correspondences in tasks where discrete correspondences are required. These models operate on the 4D space of image matches and can be trained to identify coherent patterns of correspondences enabling to disambiguate difficult matches. Finally, because the dense formulation

of the Neighbourhood Consensus Network is memory and computationally intensive, limiting its applicability, we develop a more efficient variant that can significantly reduce the memory requirements and execution time. This efficient formulation uses a sparse-correlation tensor for storing the tentative correspondences, which is processed by a sparse submanifold 4D CNN that can filter out the incorrect correspondences. Our methods obtain state-of-the-art results in the PF-WILLOW, PF-PASCAL, TSS and Caltech-101 category-level matching benchmarks, as well as in the HPatches-Sequences, InLoc and Aachen Day-Night instance-level matching benchmarks.

Acknowledgements

I would like to thank Josef and Relja for having believed in me and having decided to take me as a PhD student in the first place. I really enjoyed working with you and I have learned a lot from you in these four years.

I would like to thank all the members of the Willow team, with whom I hope to keep in touch in the coming years as colleagues, and, most especially, friends.

I would like to thank all the researchers I have collaborated with: Mircea Cimpoi, Torsten Sattler, Jiri Sedlar, Tomas Pajdla, Akihiko Torii, Hajime Taira and Mihai Dusmanu. It was a great pleasure to work with you.

I would like to thank my family for their support both before and during my PhD, particularly my parents, grand-parents and my brother.

I would like to thanks my friends Marto, José, Cristian, Nico and Calle for our chats and calls, despite of the distance.

Finally, I would like to very specially thank Vero for joining me on this journey and for being by my side during all these years in France. This thesis is dedicated to our daughter Clara.

Contents

Chapter 1

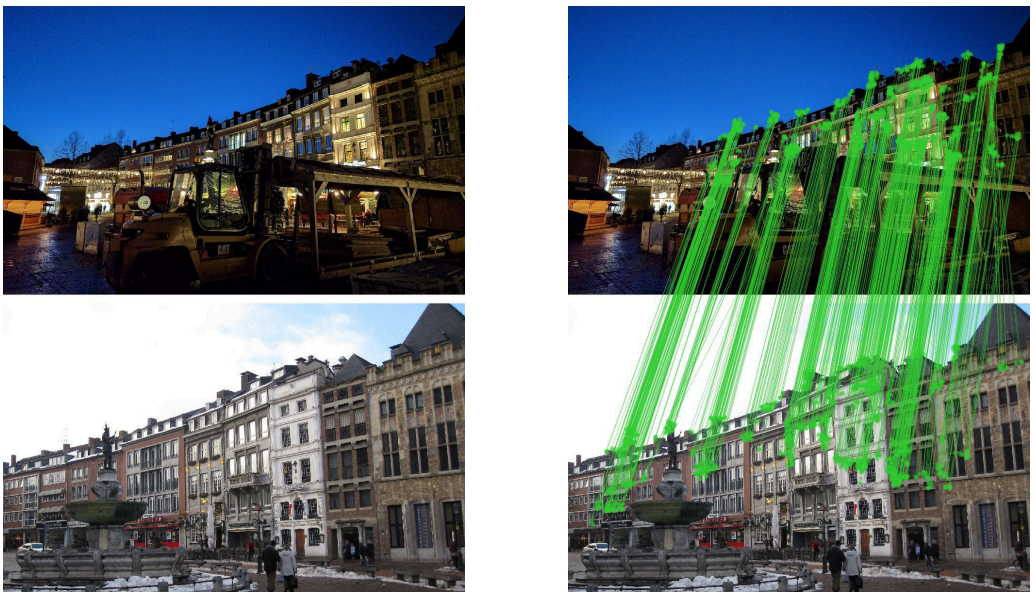
Introduction

1.1 Goal

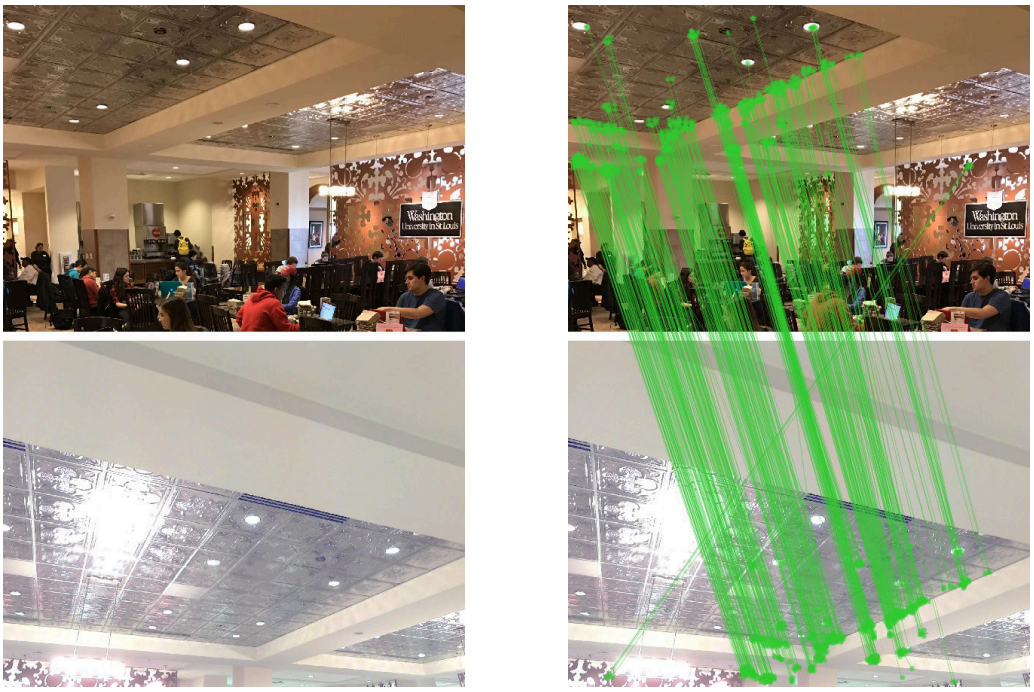
The goal of this thesis is to develop methods for establishing correspondences between pairs of images in challenging situations, such as (i) extreme illumination changes, as in the day-night matching scenario, (ii) matching scenes with little texture or with repetitive structures, as is frequently the case for indoor scenes, (iii) handling matching of scenes across long time-scales (*e.g.* years), where some structural elements may have been modified and (iv) matching parts of objects which belong to the same class, but which may have large intra-class appearance differences. An illustration of the goal of this thesis is presented in Fig. 1-1.

Although existing methods based on local invariant features have been used to successfully establish correspondences in many challenging situations, their performance is inherently limited by the underlying assumptions in their manually engineered designs. While hand-crafted feature descriptors can be designed to be invariant to affine illumination transformations, they are not invariant to strong illumination changes, such as in day-night matching, or to large appearance changes, such as in the case of category-level matching.

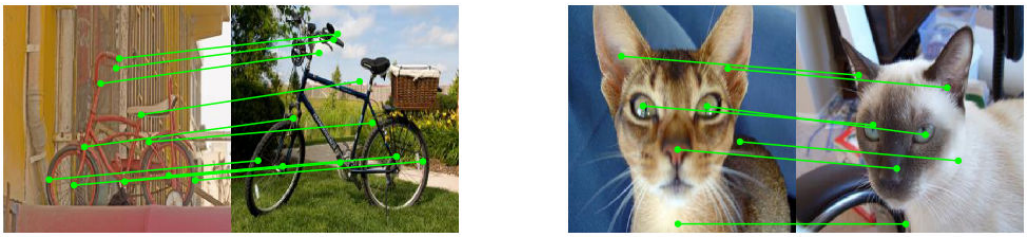
To overcome the limitations of manually engineered methods, our goal in this thesis is to develop *trainable* methods for finding correspondences between images, in which the image descriptors can acquire the necessary invariances for solving these



(a) Matching (shown as green links) in outdoor scenes with day-night illumination changes.



(b) Matching in indoor scenes with repetitive structures and surfaces with little texture.



(c) Matching different objects of the same object category across large changes in appearance.

Figure 1-1: **Goal**. We seek to solve the image correspondence problem in challenging situations, such as (a) day-night matching, (b) dealing with scenes with repetitive structures and little texture, or (c) matching parts of objects which belong to the same class.

challenging matching tasks by learning from the data itself.

Solving the difficult image correspondence problems that we target in this thesis in a trainable manner presents several challenges, which are discussed next.

1.2 Challenges

When seeking to develop trainable models to solve the image correspondence problem, we must address several challenges. These can be divided into three groups, as illustrated in Fig. 1-2: (i) challenges inherent to the difficult matching problems that we address, (ii) challenges related to the formulation of suitable end-to-end trainable models for solving the correspondence problem, and (iii) challenges related to finding a suitable training scheme for training such models.

1.2.1 Challenging matching problems

Large appearance variation. Matching images with large appearance variation, such as the strong illumination changes in day-night matching, or the large intra-class variations in category-level matching, is difficult. In order to be able to find correspondences under these large appearance variations, we require intermediate feature representations that are invariant to these factors of variation. Moreover, these features should be descriptive and discriminative enough so that they can be unambiguously matched. In the past, feature point detectors have been used to select salient image regions, with the hope that these would be more robustly described and unambiguously matched. However, the usage of detectors presents additional difficulties such as achieving high repeatability under these strong appearance variations. An alternative option consists of densely extracting descriptors along a regular grid over the images. While dense descriptors circumvent the issue of missing detections due to low repeatability, they introduce additional challenges in terms of computational complexity. Therefore, finding trainable intermediate feature representations that are robust to large appearance variations while being computationally efficient represents a challenge.

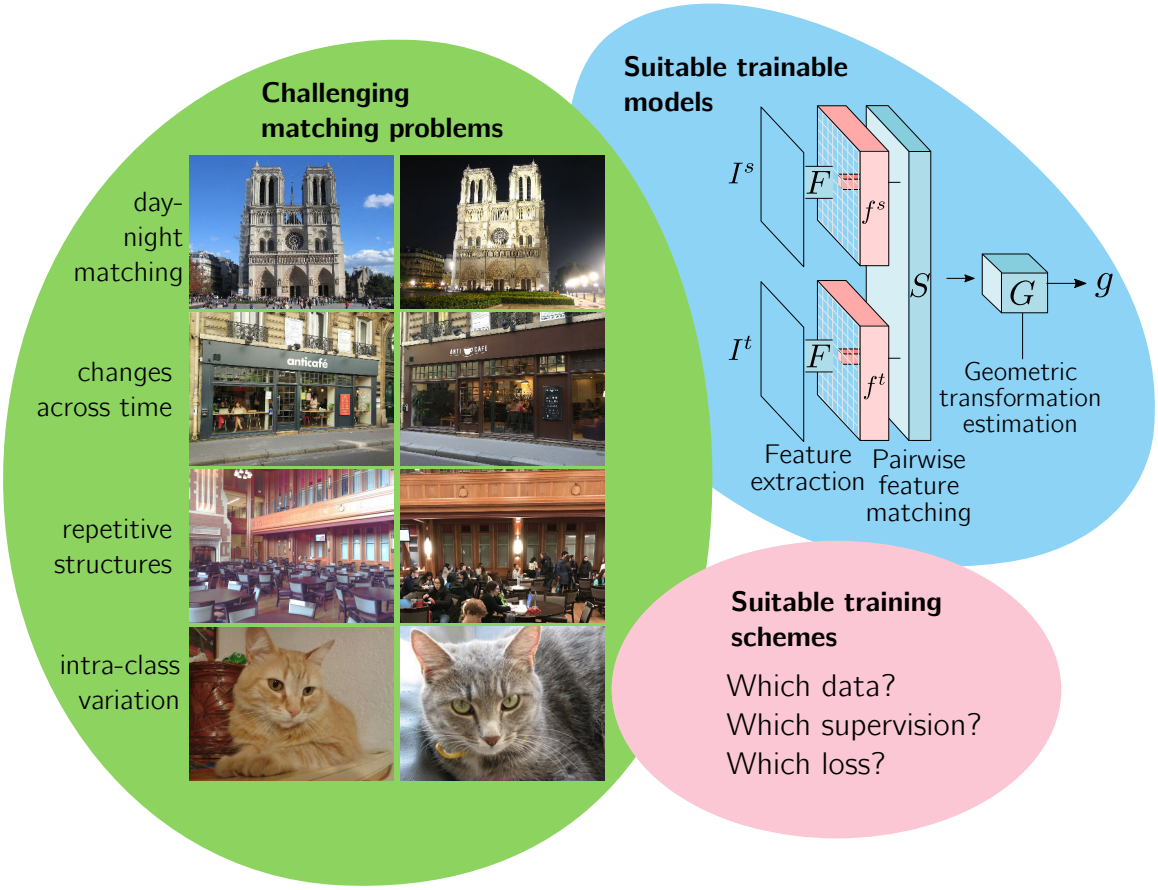


Figure 1-2: **Challenges**. We identify three main types of challenges: (i) those inherent to solving the difficult matching problems we address, such as day-night matching or category-level matching, (ii) those related to the formulation of suitable end-to-end trainable models for solving the correspondence problem, and (iii) finding a suitable training scheme for training such models.

Large viewpoint variation. Achieving robustness to large variations in viewpoint in trainable features is challenging. While the classic approach of using covariant feature detectors to normalize the image regions used for description can be used, the detection stage typically involves several hard-decision steps (*i.e.* which points are considered detections, which is the correct scale and rotation, etc.) which are not differentiable. On the other hand, trainable models can gain invariance to viewpoint changes by training with diverse data or by performing data augmentation during the training process. However, the level of invariance that can be learned is limited by the observed diversity in the data and the augmentations used, and the generalization of these invariance properties to unseen data is not guaranteed. Therefore, designing

trainable feature representations with good viewpoint invariance properties constitutes an additional challenge.

Repetitive structures and low-textured areas. In this thesis, we also would like to obtain correspondences in indoor scenes, which typically have repetitive structures (such as doors, windows or columns) or low-textured areas (such as walls or ceilings). In both of these cases, establishing matches is challenging due to the ambiguities that arise, as one particular image feature may have several good matching candidates in the other image.

1.2.2 Suitable trainable models

End-to-end trainable matching. While most of the work in trainable methods for correspondence estimation has concentrated on the development of trainable descriptors, we target the combined problem of description and matching. Therefore, we seek to obtain trainable models that can be optimized for the matching task directly, and in an end-to-end trainable way. This poses the challenges of designing a feature extraction module and a matching module which are both amenable to back-propagation.

Accurately localized correspondences. For many downstream image correspondence tasks, such as pose estimation or 3D reconstruction, having accurately localized correspondences is crucial. While classical hand-crafted methods can achieve sub-pixel keypoint localization accuracy, as they employ computationally inexpensive feature detectors, trainable models tend to be more computationally expensive, which may limit the resolution in which images can be processed and, in consequence, limiting the localization accuracy of the output correspondences. Therefore, obtaining trainable models for image correspondence estimation that can obtain accurately localized correspondences while being computationally efficient, is a challenging problem.

1.2.3 Suitable training scheme

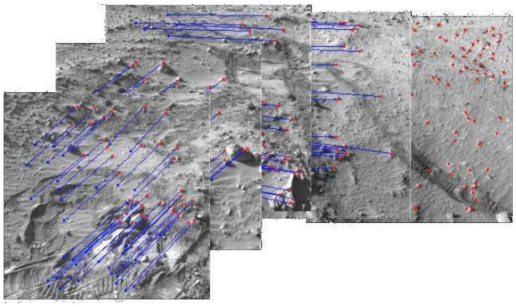
In order to train a model for the matching task, we require training data which contains the rich diversity in appearance and viewpoint variations that we wish to tackle at testing time. Furthermore, unless a fully unsupervised approach is used, this data will require a certain level of manual annotation, in accordance with the type of supervision and loss that will be used to train the model. As annotating image correspondences densely is unfeasible, the fully-supervised setup cannot be directly employed. Therefore, approaches such as self-supervised learning, weakly-supervised learning or unsupervised learning will need to be used. Finding suitable training data and a suitable training scheme is, therefore, an additional challenge.

1.3 Motivation

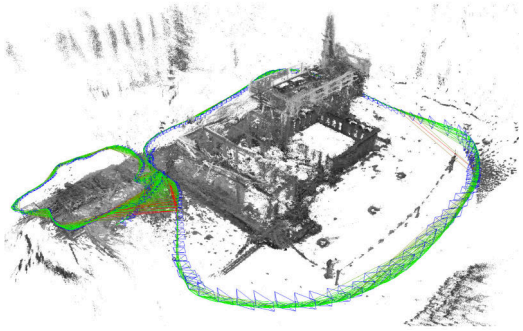
Correspondence estimation is a fundamental problem of computer vision with numerous applications in robotics, augmented reality and 3D reconstruction, some of which are presented below.

Visual odometry. When mobile robots displace, the robot pose can be estimated from the commands given to its control system or by an inertial measurement unit. However, small pose errors can accumulate leading to a large pose error, a phenomenon that is called drifting. The visual information coming from the robot's cameras can be used (possibly jointly with other sensors) in order to estimate the robot position and reduce the drift. For this, the correspondences between images taken at different times need to be robustly established, which is particularly difficult in indoor scenarios with low texture or repetitive structures. An example of visual odometry is presented in Fig. 1-3a.

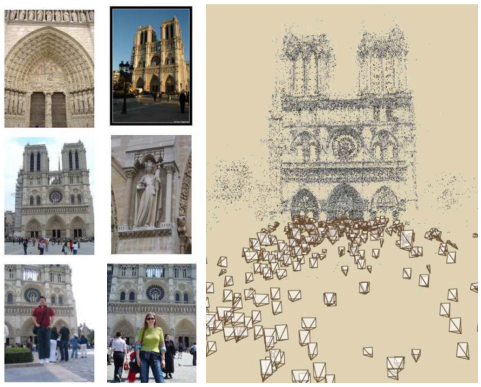
Simultaneous localization and mapping. Related to visual odometry is the problem of simultaneous localization and mapping (SLAM), where both the robot pose and scene structure are estimated simultaneously. In this case, a map of the



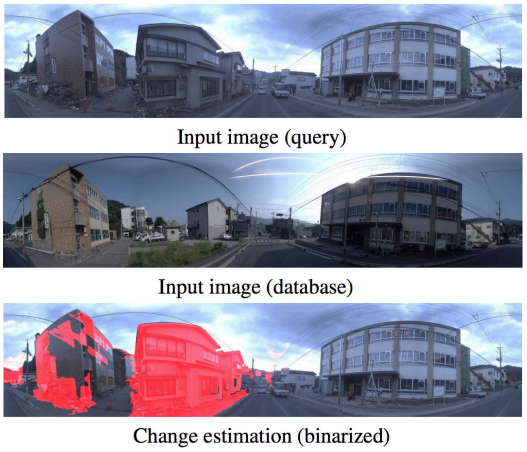
(a) Visual odometry for Mars Rover navigation [Maimone et al., 2007]



(b) Simultaneous localization and mapping [Engel et al., 2014]

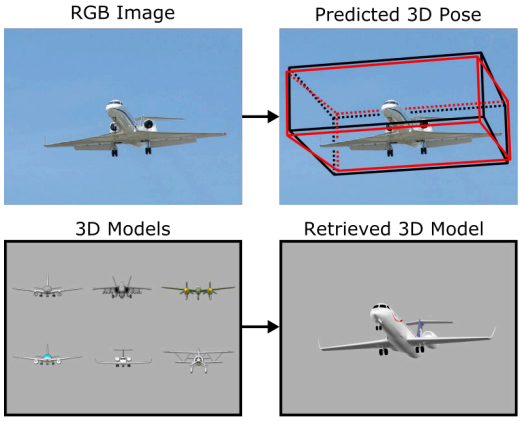


(c) 3D reconstruction from photo collections [Snavely et al., 2006]



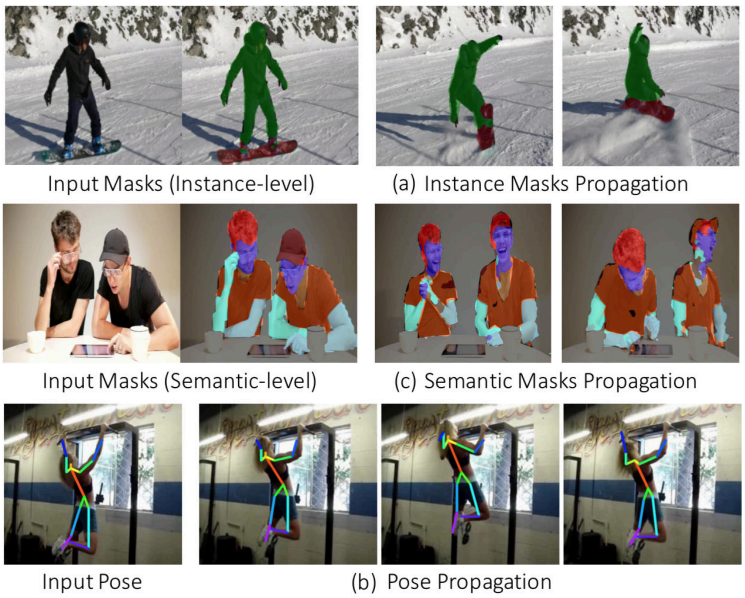
(d) Scene change estimation [Sakurada and Okatani, 2015]

(e) Visual localization [Sattler et al., 2018]

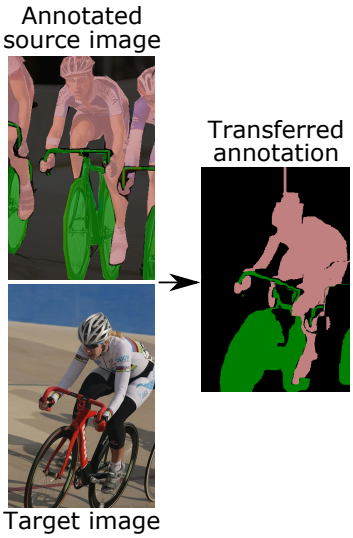


(f) Object pose estimation [Grabner et al., 2018]

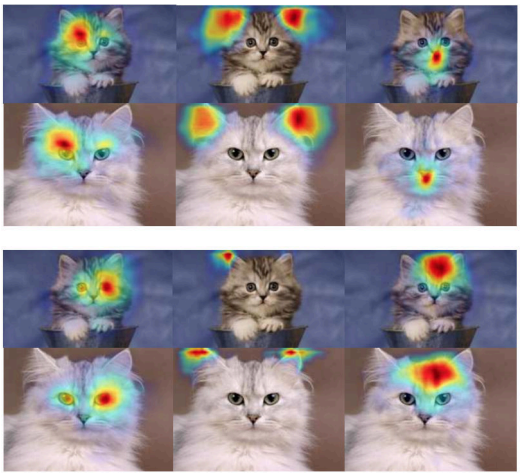
Figure 1-3: **Different applications relying on correspondence estimation**.



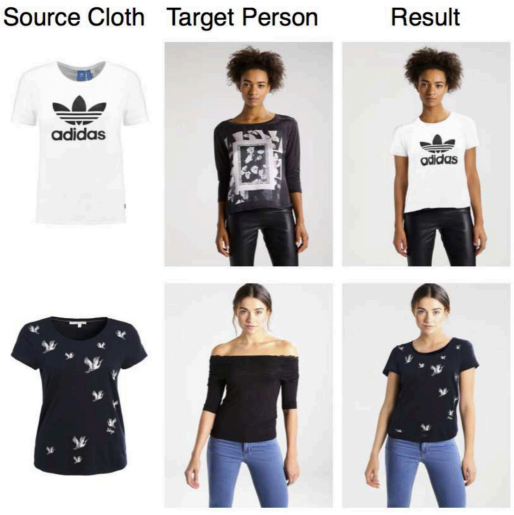
(g) Propagation of annotations [Wang et al., 2019]



(h) Semantic annotation transfer [Kim et al., 2018b]



(i) Object part discovery [Novotny et al., 2017]



(j) Semantic image editing [Han et al., 2019]

Figure 1-3: (Cont.) **Different applications relying on correspondence estimation**.

environment is built as the robot moves, and is used for its localization. Again, the estimation of correspondences constitutes one of the main underlying problems. An illustration of the map and camera poses obtained with SLAM is presented in Fig. 1-3b.

3D reconstruction from photo collections. Furthermore, correspondence estimation can be used to reconstruct the 3D scene structure from a set of images using structure-from-motion approaches, which can be followed by multi-view stereo algorithms. The created 3D models can be used for diverse purposes such as visual localization (discussed next), architecture, archaeology or auditing construction sites. While images are usually acquired by flying drones or driving cars equipped with cameras, they can also come from unstructured photo collections downloaded from the internet, in which case images may contain large viewpoint and illumination variation, as well as scene modifications. An example of a 3D model obtained from an internet photo collection is presented in Fig. 1-3c.

Visual localization. For robotic localization applications, such as required by self-driving cars or indoor robot navigation, visual localization can be used to obtain precise localization estimates. In this scenario, correspondences between a query image from the robot and a previously obtained database image are established, and 3D algorithms (such as Perspective-n-Point) are used to estimate the robot's pose. The problem of visual localization is illustrated in Fig. 1-3e, where a pre-computed 3D model of the environment is shown on top and different possible query images used for localization are shown at the bottom. Note that these images contain large variations in illumination conditions.

Object pose estimation. Object pose estimation is an important problem for robotic manipulation and augmented reality which can be tackled by similar correspondence estimation approaches as camera pose estimation. In this case, correspondences between an object of interest and the rendered images of a similar 3D model can be computed and used to estimate the pose. Note that the *exact* 3D model of the object

might be unavailable. In that case, a *similar* 3D model from the same object category can be used, resulting in a category-level matching problem.

Propagation of annotations in videos. Another application of correspondence estimation is to propagate information from one video frame to the subsequent frames. The information from the initial frame can take several forms, such as an object bounding box (in which case we call the problem *object tracking*), semantic labels, or object keypoints. An example of this task is presented in Fig. 1-3g.

Semantic annotation transfer. Related to the previous task is the problem of annotation transfer across different images of the same category. In this case, the annotations for a target image can be obtained by transferring the annotations of a semantically related source image. These annotations can consist of sparse keypoints or semantic labels, for instance. As these annotations are costly to obtain, this technique can be used to obtain an estimate of the annotations, which can be later refined manually or by other automatic techniques. An illustration of semantic annotation transfer is presented in Fig. 1-3h.

Object part discovery. Correspondence estimation can be also used for unsupervised object part discovery. In this case, a collection of unannotated images from the same category can be used to discover the most characteristic object parts. An example of this task is presented in Fig. 1-3i.

Semantic image editing. Finally, category-level image alignment can be used to perform image-editing, such as replacing the texture of an object with a different texture from the same category. An example of this for the task of garment *virtual try-on* is presented in Fig. 1-3j.

1.4 Contributions

In this thesis, we make contributions in both parametric image alignment and discrete correspondence estimation and develop methods that are able to handle both instance and category-level matching. The key contributions of this thesis are the following:

1. Trainable parametric image alignment. Our first contribution consists of a trainable approach for parametric image alignment employing a Siamese network model. This model processes two input images and can estimate the parameters of a geometric transformation such as affine, homography or thin-plate spline. The proposed model contains three distinct modules for feature extraction, feature matching and parameter regression. These modules are implemented using differentiable operations, resulting in an end-to-end trainable architecture.

2. Correlation layer for Siamese-networks. In particular, for the matching module of the Siamese network, we propose to use a correlation layer instead of the subtraction or concatenation approaches used in other works employing Siamese networks for related tasks. An ablation study shows the superiority of the correlation operation with respect to these other alternatives. We believe that the correlation operation produces superior performance as it solely retains image similarity information, thus obtaining good generalization properties despite a large domain gap between training and evaluation, which is not the case in the subtraction or concatenation operations which retain information about the actual image content and are therefore more sensitive to such domain gap.

3. Transformation-agnostic loss for strong supervision. Furthermore, we propose a transformation agnostic loss that operates on image coordinates instead of the transformation parameters directly. This loss enables the model to be trained with different types of geometric transformations without modifications to the method or its implementation and avoids having to worry about the particular parameterizations of each geometric model.

4. Soft-inlier count module and loss for weak supervision. The previously presented loss requires knowledge of the ground-truth transformation and was used in combination with synthetic imagery. In order to train from real image pairs lacking correspondence annotation, we develop a *soft-inlier count* module. This soft-inlier count module is inspired by the *inlier count* measure used in RANSAC but implemented in a differentiable way to allow training using standard backpropagation. We show that the proposed weakly-supervised approach can provide a significant performance gain with respect training solely with synthetically warped-images.

5. Neighbourhood Consensus Networks. Moreover, we develop the Neighbourhood Consensus Networks which can be used to robustly estimate correspondences in tasks where discrete correspondences are required. These models operate on the 4D space of image matches and can be trained to identify coherent patterns of correspondences enabling to disambiguate difficult matches or correct errors in tentative matches caused by large changes in appearance and background clutter. These models can also be incorporated into other computer vision pipelines as a generic robust matching model. In order to train the Neighbourhood Consensus Network, we propose an image-level loss which operates on pairs of positive images, where correspondences between scenes or objects exist, and of negative images, where scenes or objects do not correspond. This loss allows for training in a weakly supervised way and facilitates data collection as little annotation is required.

6. Indoor venues dataset (IVD). In addition, we collected and released a dataset for indoor localization consisting in 3861 corresponding image pairs from 89 different indoor scenes, originally uploaded by individuals to Google Maps. The dataset includes different types of venues such as restaurants, cafes and museums from six different cities (Amsterdam, Brussels, Copenhagen, Edinburgh, Paris and Prague). The dataset contains challenging image pairs featuring changes in illumination and modifications of the scene (both temporal, due to transient objects, and permanent), providing therefore similar conditions to those observed in real indoor localization situations.

7. Efficient Neighbourhood Consensus Networks. Finally, because the dense formulation of the Neighbourhood Consensus Network is memory and computationally intensive, therefore limiting its applicability, we develop a more efficient variant that can reduce the memory requirements and run-time by more than 10×. This efficient formulation uses a sparse-correlation tensor for storing the tentative correspondences, which is processed by a sparse submanifold 4D CNN that can filter out the incorrect correspondences.

1.5 Outline of the thesis

In Chapter 2, we present a literature review discussing the relevant methods for solving instance and category-level correspondence problems. In particular, we review both hand-crafted and trainable methods for obtaining both discrete and dense image correspondences.

Chapter 3 presents our trainable approach for parametric image alignment (contribution 1), based on a Siamese CNN with a correlation layer for matching (contribution 2), and employing our proposed strongly supervised loss together with synthetic imagery (contribution 3). Results are presented for both instance and category level matching problems.

Chapter 4 presents the soft-inlier count module that allows training the previously presented model in a weakly-supervised manner (contribution 4), obtaining a significant improvement over the baseline model for category-level matching.

Chapter 5 presents the Neighbourhood Consensus Networks for robust estimation of discrete image correspondences (contribution 5), with applications for category-level matching and indoor localization, as well as a suitable weakly supervised training loss for training from an indoor dataset containing only annotation at the level of image pairs (contribution 6).

Chapter 6 addresses the limitations of Neighbourhood Consensus Networks and presents the more efficient Sparse Neighbourhood Consensus Networks, which exploit the sparsification of the correlation tensor and a sparse submanifold CNN to obtain

significant performance gains (contribution 7).

Chapter 7 concludes this thesis presenting the main obtained results and possible future research directions.

1.6 Publications and Software

During the development of this thesis, four papers were presented at major computer vision and machine learning conferences (CVPR'17, CVPR'18, NeurIPS'18 and ECCV'20). In addition, extended versions of two of these conference papers were accepted by the T-PAMI journal (one was published and the other one is in press). The following chapters of this thesis present the material from our publications in the following way:

- Chapter 3 is based on the *Convolutional neural network architecture for geometric matching* conference paper, presented as spotlight at CVPR'17 [Rocco et al., 2017], as well as in its T-PAMI'18 extended journal version [Rocco et al., 2018b].
- Chapter 4 is based on the CVPR'18 paper *End-to-end weakly-supervised semantic alignment* [Rocco et al., 2018a].
- Chapter 5 is based on the NeurIPS'18 conference paper *Neighbourhood Consensus Networks* which was accepted as a spotlight [Rocco et al., 2018c], as well as in an extended journal version to appear in T-PAMI [Rocco et al., Accepted].
- Chapter 6 is based on the ECCV'20 conference paper *Efficient Neighbourhood Consensus Networks via Submanifold Sparse Convolutions* [Rocco et al., 2020].

All the software developed during this thesis is available online at `https://www.di.ens.fr/~iroccosp/` under open-source licences.

Chapter 2

Literature Review

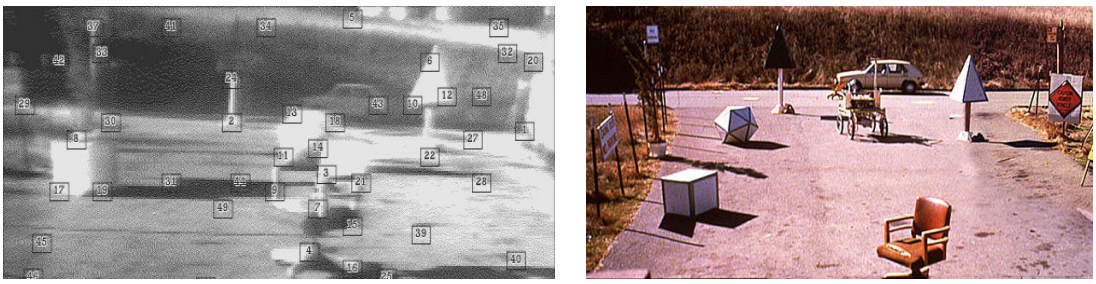
In this chapter, we review related work on correspondence estimation. We begin by reviewing methods for instance-level matching in Sec. 2.1, followed by methods for category-level matching in Sec. 2.2.

2.1 Instance-level matching

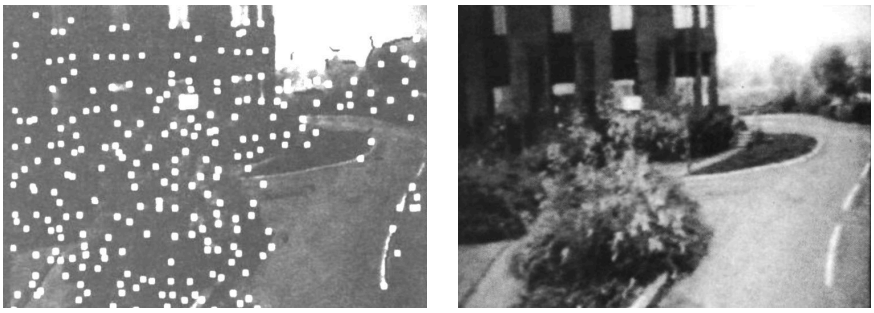
In this section, we review methods for obtaining correspondences between images of the same scene, a problem which we call *instance-level matching*. We begin by reviewing manually engineered methods for local feature detection (Sec. 2.1.1), followed by methods for obtaining invariant feature descriptors (Sec. 2.1.2). Then, in Sec. 2.1.3, Sec. 2.1.4 and Sec. 2.1.5 we present more recent trainable methods for feature description, detection or joint detection and description, respectively. Next, in Sec. 2.1.6 we present the alternative approach of using densely extracted features, thus avoiding the feature detection step. Finally, in Sec. 2.1.7 we review different strategies for filtering the obtained candidate matches.

2.1.1 Hand-crafted local feature detectors

In constrained correspondence problems such as stereo-vision or optical-flow, the search space for a correspondence can be reduced to a local neighbourhood of the



(a) Moravec corner detector



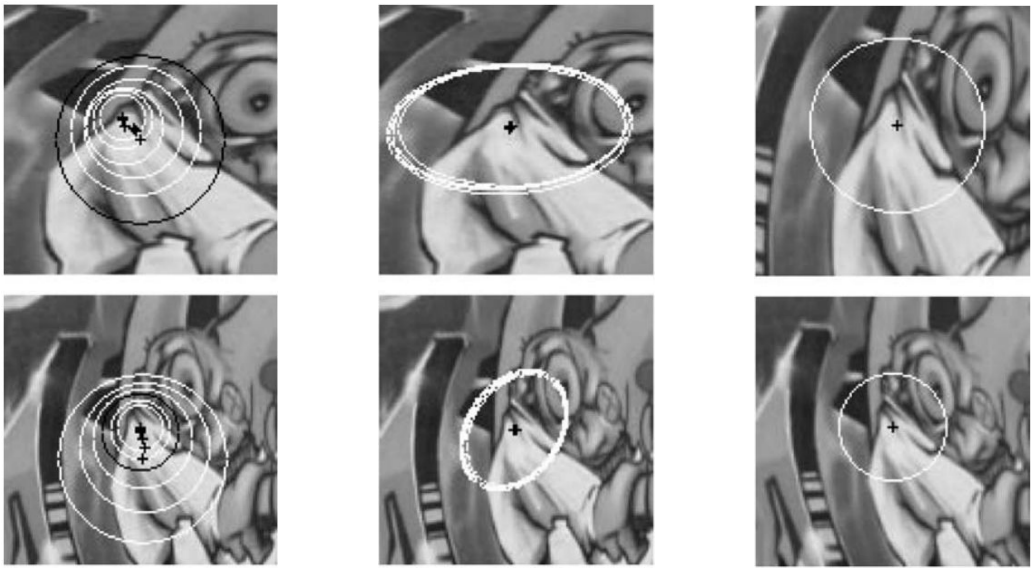
(b) Harris corner detector

Figure 2-1: **Early corner detectors**. We show the output of the early Moravec (a) and Harris (b) corner detectors. The detected corners are shown on the left, and the imaged scenes are shown on the right (note that in (a) we show a photograph of the scene, while (b) corresponds to the actual digital image used for corner detection). Images from [Moravec, 1980] and [Harris and Stephens, 1988].

initial point, significantly simplifying the complexity of the correspondence assignment. In a general setting where there is a large motion between camera poses, however, matches cannot be assumed to lie within a local neighbourhood.

In this more general case, obtaining dense correspondences is extremely challenging. Therefore, in order to simplify match assignment, local feature detectors have been developed. In this scenario, local image features such as corners, blobs or stable-regions are first extracted and in a subsequent stage used for matching.

An early corner detector was proposed by Moravec [1980], where mean local changes in image intensity along different directions are computed, and local maxima are selected. An example of Moravec's early corner detector is presented in Fig. 2-1a. An improvement was later proposed Harris and Stephens [1988], who replaced the computation of image intensity differences with a first-order Taylor expansion, improv-



(a) (b) (c)

Figure 2-2: **Harris-Laplace and Harris-Affine detectors.** (a) Multi-scale Harris detections (white) and selected scale using the LoG (black). (b) Result of the iterative affine adaptation proposed in Harris-Affine for the multi-scale Harris detections of (a). (c) Normalized local image region, according to the estimated affine parameters. [Mikolajczyk and Schmid, 2004].

ing isotropy properties (and therefore improving invariance to rotation). Furthermore, instead of considering the local maxima of the image intensity change function directly, Harris and Stephens [1988] proposed to select the local maxima of a *cornerness* function based on the eigenvalues of the local structure tensor, allowing to eliminate responses on edges. An example of the Harris' corner detector is presented in 2-1b.

Later on, Mikolajczyk and Schmid [2004] proposed Harris-Laplace points which extended the formulation of Harris and Stephens [1988] by adding multi-scale processing (following Lindeberg [1998]) combined with a scale-selection step using the Laplacian-of-Gaussian (LoG) function. In this way, the Harris-Laplace detector achieves scale-covariance. In addition, Mikolajczyk and Schmid [2004] propose an additional variant called Harris-Affine, where multi-scale Harris points are first extracted, and then affine regions are iteratively estimated, thus achieving affine-covariance. A similar method for affine adaptation was proposed by Schaffalitzky and Zisserman [2002b]. An example of Harris-Laplace and Harris-Affine features is presented in Fig. 2-2. In

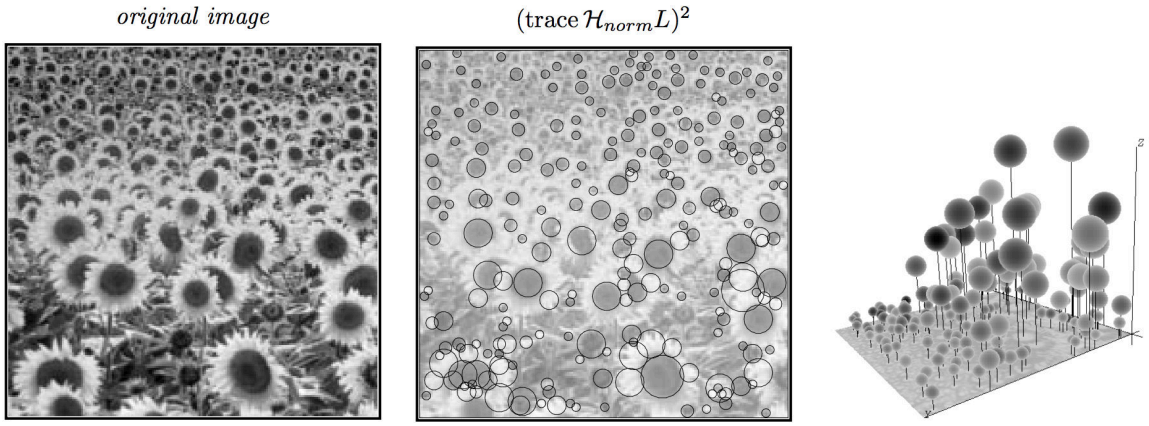


Figure 2-3: **Laplacian-of-Gaussian detector.** (a) Original image. (b) Detected scale-covariant blobs. (c) Detected blobs displayed in the 3D scale-space [Lindeberg, 1998].

Chapter 3 we develop an iterative approach for dealing with large viewpoint changes which is similar in spirit to the iterative estimation in Hessian-Affine features. However, our method attempts to directly register a pair of images without using local image features.

The LoG function can also be used by itself as a scale-covariant blob detector operating in a 3D scale-space representation of the image, as proposed by Lindeberg [1998], which extends the initial method of Beaudet [1978]. In this case, the image is convolved with a family of LoG filters of different scales, and local maxima in the 3D scale-space are selected, as illustrated in Fig. 2-3. A variant of this approach was proposed by Lowe [2004], where the LoG function is approximated by the Difference-of-Gaussians (DoG) function, which allows for a more computationally efficient implementation. Furthermore, as these detection functions are prone to selecting detections on edges, Lowe [2004] proposed an edge-response elimination approach based on an analysis of the eigenvalues of the Hessian matrix, which is similar in spirit to the analysis of the eigenvalues of the structure tensor previously proposed by Harris and Stephens [1988]. Another similar approach for edge-response elimination was proposed by Mikolajczyk [2002], by selecting keypoints that are simultaneously local extrema of both the trace and determinant of the Hessian.

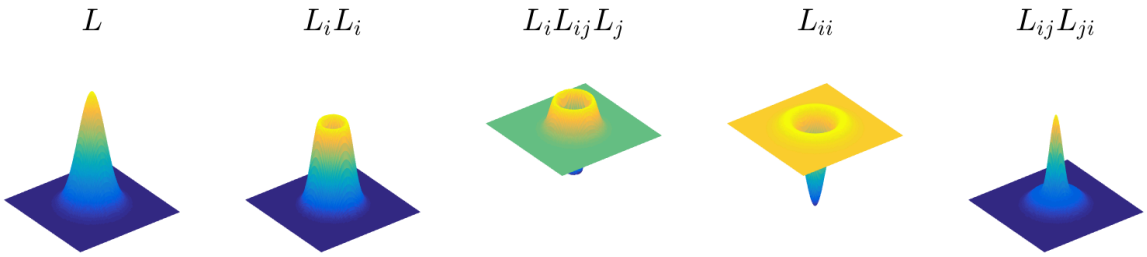
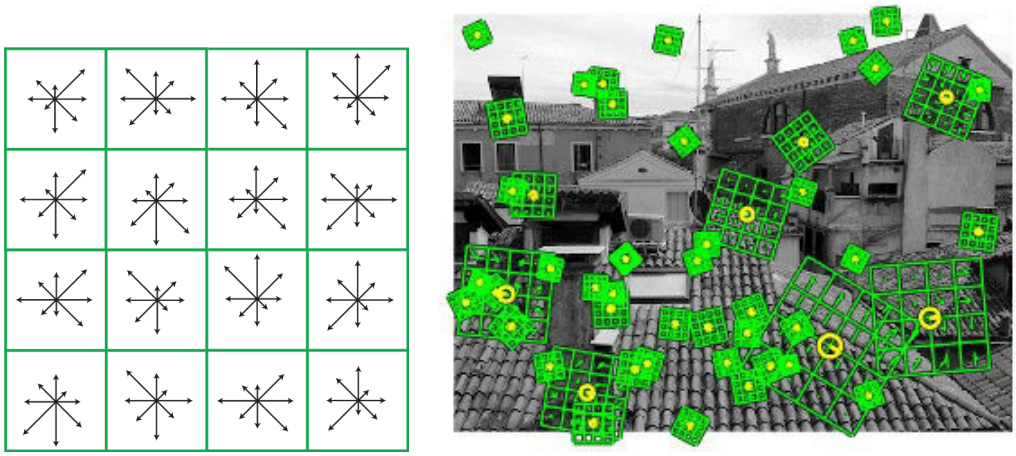


Figure 2-4: **Rotational invariant kernels.** We plot the first 5 rotational invariant kernels (up to second order) used by Schmid and Mohr [1997] to compute rotational invariant descriptors. They consist of a Gaussian kernel (L) and combinations of its derivatives as indicated by the subindices (expressed in Einstein notation for compactness).

2.1.2 Hand-crafted local feature descriptors

In the previous section, we reviewed the development of corner and blob feature detectors. While these methods provide candidate regions for matching, therefore greatly simplifying the matching task by reducing the search space of candidate matches, they do not directly provide any means for comparing two image regions. Early corner detectors such as Moravec or Harris corners were typically matched by comparing local image grayscale patches extracted around the detected corners with a normalized correlation function [Faugeras et al., 1992; Thacker and Courtney, 1992]. However, this approach is very sensitive to misalignments, noise and illumination changes, which lead to the development of more sophisticated *feature descriptors*.

An early feature descriptor was proposed by Schmid and Mohr [1997], where each detected keypoint would be associated with a descriptor vector composed of nine rotational differential invariants. These invariants are computed by convolving the image with a Gaussian kernel and different combinations of its derivatives (up to third order), achieving better robustness to noise (due to the averaging effect of the Gaussian Function) and illumination changes (due to differentiation). The first five rotational invariant kernels are illustrated in Fig. 2-4. The obtained rotation-invariant descriptor can be used in conjunction with a multi-scale approach to incorporate invariance to scale changes. In order to obtain correspondences, these descriptors are matched using the Mahalanobis distance. Alternatively, the descriptors can be



(a) (b)

Figure 2-5: **SIFT descriptor.** (a) Illustration of the descriptor. The descriptor is composed by 16 histograms arranged in a 4×4 grid, as shown in green. Each of these histograms contains 8 orientation bins, shown by the arrows, which accumulate the orientation of image gradients in a $4\text{px} \times 4\text{px}$ image cell. (b) Example SIFT descriptors (green) extracted on oriented DoG keypoints (yellow). Figures reproduced from [Lowe, 2004] and Vedaldi and Fulkerson [2010].

normalized (using an estimate of the covariance matrix of each component) and then be directly matched according to the Euclidean distance.

While the descriptor of Schmid and Mohr [1997] can achieve invariance to rotation and scale changes, it is not robust to misalignment errors in the detections or local deformations due to perspective projection. In order to address these issues, Lowe [2004] proposed the SIFT descriptor which incorporates ideas from the biologically-inspired complex-cell model of Edelman et al. [1997] that emulates the behaviour of orientation-selective neurons which are insensitive to small translations. In order to achieve a similar behaviour, the SIFT descriptor computes a set histograms of image gradient orientations, each accumulating the gradient orientation information of a particular image region around the detected keypoint, as illustrated in Fig. 2-5. Typically, 16 histograms arranged in 4×4 grid around the keypoint are used for computing a SIFT descriptor, each containing 8 orientation bins. The final descriptor consists of the concatenation of these histograms, thus being 128-dimensional, and is normalized to unit length. Note that while each histogram is associated with a particular shift from the keypoint, each histogram only encodes orientation information

in a small image region. In this sense, the SIFT descriptor is relatively insensitive to small local translations, as the complex cells of Edelman et al. [1997]. While Lowe [2004] proposes to use SIFT descriptors together with DoG keypoints, thus achieving scale-invariant descriptors, they can also be combined with Harris-Affine keypoints if affine invariant descriptors are required (at the cost of possibly lower detector repeatability). For this, SIFT descriptors are computed over image patches which are previously normalized using the keypoints' affine parameters, as illustrated in Fig. 2-2c. In principle, SIFT features can be directly matched with the Euclidean distance. However, because they are composed of histograms, the Hellinger kernel constitutes a better distance as shown by Arandjelović and Zisserman [2012]. In a similar spirit to the normalization used by Schmid and Mohr [1997], Arandjelović and Zisserman [2012] propose a normalization for SIFT features consisting of an L1 normalization step followed by the square root (which they call RootSIFT features), which allows using the Euclidean distance directly for comparison while obtaining the benefits of using the Hellinger distance. While other hand-crafted descriptors have been proposed, such as SURF [Bay et al., 2006], BRISK [Leutenegger et al., 2011], ORB [Rublee et al., 2011], KAZE [Alcantarilla et al., 2012] and AKAZE [Alcantarilla et al., 2013], SIFT (or its normalized version RootSIFT) has been much more widely adopted, and it is still one of the main approaches used for instance-level matching, and particularly for 3D reconstruction [Sattler et al., 2018; Dusmanu et al., 2019].

2.1.3 Trainable local feature descriptors

Despite the large success of SIFT features for many correspondence tasks, its matching performance degrades under large viewpoint or non-affine illumination changes. This limitation led to the development of trainable local feature descriptors, with the hope that the required invariances to viewpoint and illumination-changes could be *learnt* from data. However, trainable methods present additional challenges such as finding suitable training data and a suitable training loss.

In the early approach by Winder and Brown [2007], a parametric descriptor based on modules inspired by hand-crafted descriptors such as SIFT is proposed, and the

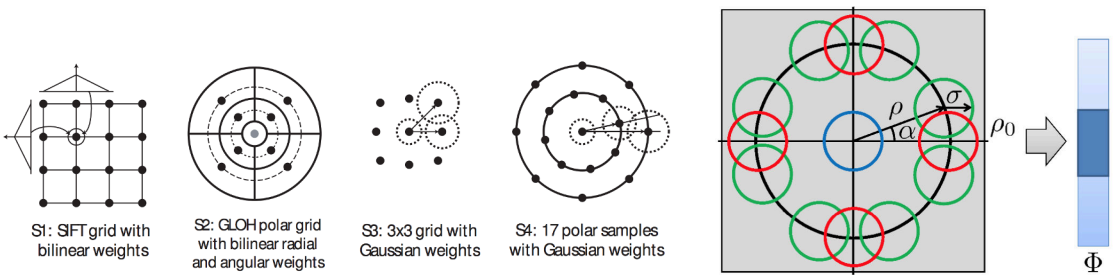


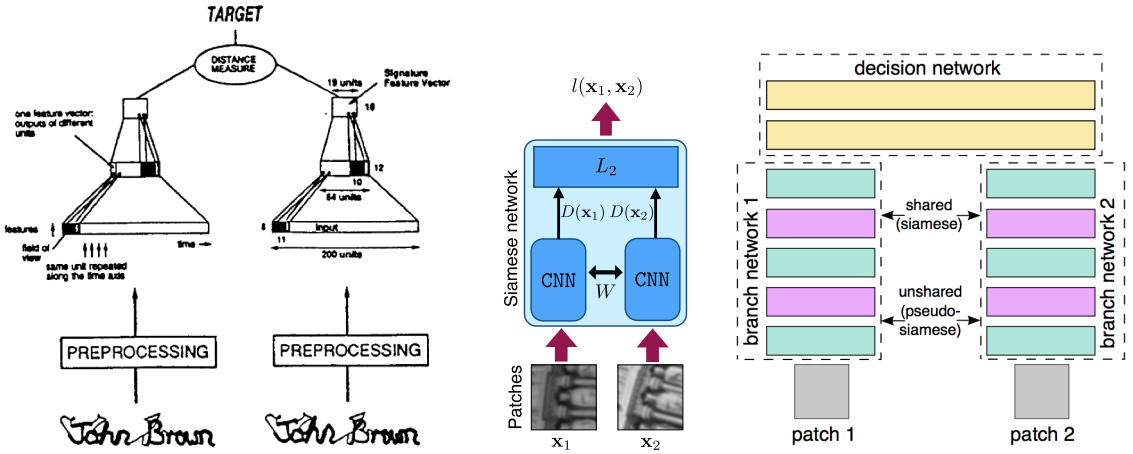
Figure 2-6: **Parametric descriptors inspired by SIFT.** (a) Different spatial-aggregation configurations evaluated by Winder and Brown [2007]. (b) Parametric spatial-aggregation configuration of Simonyan et al. [2014]. Figures reproduced from the respective works.

task is to learn a good set of parameters for it. Simonyan et al. [2014] later extend this approach by casting it as a convex optimisation problem, while also proposing to learn a dimensionality reduction matrix which can improve descriptor matching performance. These approaches are illustrated in Fig. 2-6.

While the previously described approaches were strongly influenced by hand-crafted descriptors such as SIFT, Jahrer et al. [2008] proposed to use a Convolutional Neural Network (CNN) for computing descriptors inspired by the work of LeCun et al. [1998] which used a CNN for character recognition. Jahrer et al. [2008] proposed both a classification-based approach and a regression-based approach.

In the classification-based approach, the CNN model takes an input image patch and classifies it among 600 keypoint *classes*. Correspondences are then assigned to keypoints from the two images belonging to the same class. A similar approach had been used in the past by Lepetit et al. [2005], but using randomized trees instead of a CNN, and focusing on keypoints lying on particular objects. Furthermore, a very similar approach using a CNN was recently proposed by Cieslewski et al. [2018], with the main difference that it operates on the whole images and not on patches.

In the regression-based approach of Jahrer et al. [2008], the CNN model takes input image patches and outputs 128-dimensional descriptors, which can be compared using the Euclidean distance. Other similar regression-based CNN approaches which also compare the output descriptor vectors with the Euclidean distance were later proposed



(a) (b) (c)

Figure 2-7: **Siamese CNN architectures for image comparison.** (a) Early Siamese architecture for signature comparison by Bromley et al. [1994]. (b) Siamese architecture for patch description which uses the Euclidean distance for comparison of Simo-Serra et al. [2015]. (c) Siamese architecture which incorporates a metric learning module by Zagoruyko and Komodakis [2015]. Figures reproduced from the respective works.

by Balntas et al. [2016a], Balntas et al. [2016b], Simo-Serra et al. [2015], Mishchuk et al. [2017] and Tian et al. [2017]. During training, these regression-based CNN use a Siamese configuration with shared weights, as illustrated in Fig. 2-7b. The main differences between these methods lie in the proposed CNN architectures and the losses used for training.

Balntas et al. [2016a] propose the PN-Net model which is trained using triplets of positive and negative pairs and propose a ratio loss which they term SoftPN. The follow-up work by Balntas et al. [2016b] proposes a similar approach but incorporates an in-triplet hard-negative mining approach, where the distance between the positive and negative samples is also taken into account. They also compare the triplet ranking loss with their previously proposed ratio loss and determine that the former leads to superior matching performance. Mishchuk et al. [2017] propose to use a similar triplet ranking loss, but search for a hard-negative in both images, and use the hardest of the two. On the contrary, Tian et al. [2017] propose to use a different loss which jointly optimizes all distances between a set of sampled descriptors from each image, which they claim is a more realistic setting as the number of negative pairs is much larger

than that of positive pairs.

Other authors such as Han et al. [2015] or Zagoruyko and Komodakis [2015], incorporate metric learning into the descriptor learning problem and propose Siamese regression-based CNN architectures which include a similarity computation block, as illustrated in Fig. 2-7c.

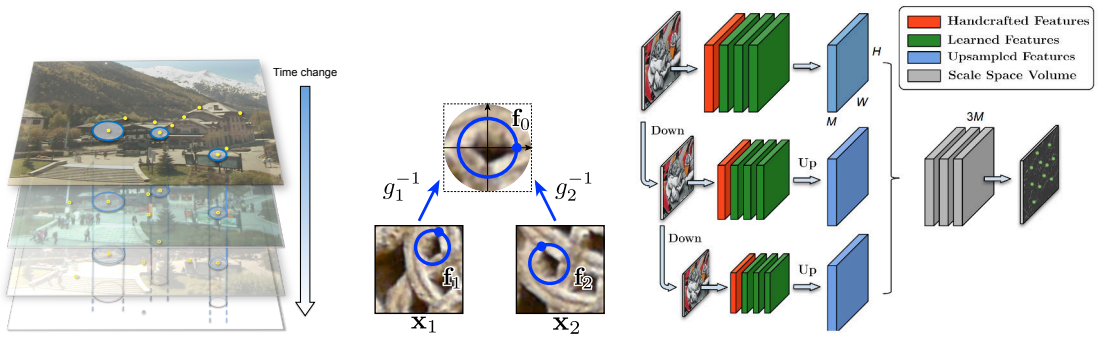
An additional difference between these methods lies in the choice of training data, and how ground-truth annotations for training are obtained. Most methods use correspondences obtained from reconstructed 3D scenes, which can capture realistic distortions from perspective projection as well as diverse illumination conditions [Winder and Brown, 2007; Balntas et al., 2016a; Simo-Serra et al., 2015; Balntas et al., 2016b; Mishchuk et al., 2017; Tian et al., 2017; Han et al., 2015; Zagoruyko and Komodakis, 2015]. A different approach was proposed by Simonyan et al. [2014], who obtain correspondences through homographies estimated using SIFT features. Alternatively, Jahrer et al. [2008] use a self-supervised approach to generate training pairs, by applying randomly-sampled transformations to a set of natural images.

2.1.4 Trainable local feature detectors

The trainable descriptors presented in the previous section operated on local image patches, extracted around hand-crafted keypoints such as DoG. Therefore, the success of such approaches depends, not only on having good descriptors for matching but also on the repeatability of the detections. If the same points are not repeatedly detected on the different images, then there is no chance for obtaining correct correspondences.

Recently, trainable keypoint detectors have also been proposed [Verdie et al., 2015; Lenc and Vedaldi, 2016; Mishkin et al., 2018; Laguna et al., 2019], with the hope of improving the detection repeatability in challenging conditions, compared to hand-crafted methods. Some of these approaches are illustrated in Fig. 2-8.

Verdie et al. [2015] tackle the challenging problem of obtaining repeatable detections in urban scenes under strong illumination or appearance changes, such as in day-night situations or under different weather conditions. For this, they train different regression models to produce a scalar 2D response map, from which detections are



(a) (b) (c)

Figure 2-8: **Different trainable detectors.** (a) The TILDE detector of Verdie et al. [2015] is trained with stacks of aligned webcam images to achieve high repeatability under strong illumination and appearance changes. (b) The covariant detector of Lenc and Vedaldi [2016] estimates a geometric transformation g_i for each input patch $\mathbf{x}_i$, which can be inverted to normalize the patches to the canonical frame. Note that detections f_1 and f_2 are defined implicitly by the estimated transformations g_1 and g_2. (c) The Key.Net architecture of Laguna et al. [2019] combines hand-crafted and trainable operations in a multi-scale approach. Figures reproduced from the respective works.

extracted by selecting local extrema, in a similar way to what is done with traditional hand-crafted detectors. For training, they use webcam images which are spatially aligned, as illustrated in Fig. 2-8a. For each scene, they cluster DoG detections that appear at similar positions under many images as positive samples and select the top-100 positive samples for each scene for training. Then, the model is trained to produce local extrema at these positive positions, resulting in a detector with high-repeatability.

Lenc and Vedaldi [2016] propose a different approach based on the covariance constraint, with the purpose of learning a covariant feature detector. In their approach, a CNN model estimates a geometric transformation from a small image patch, and the inverse of this transformation can be used to normalize the patch to a canonical frame, as shown in Fig. 2-8b. In this approach, detections are not explicitly defined as extrema of a 2D response map, but rather *implicitly* defined by means of a local geometric transformation. Then, the transformations from each patch are aggregated to select the final keypoints using a voting system (which is similar in spirit to the voting system of Hough [1962]). Note that this approach generalizes beyond corner

detection, and can be used to detect oriented circular frames (such as in SIFT), or affine frames (such as in Harris-Affine).

Mishkin et al. [2018] claim that repeatability alone does not ensure good matching, and propose to use a triplet margin ranking loss which operates on distances between auxiliary descriptors extracted at the detected features. They use this approach to train an affine-region detection model that computes the affine parameters from an input image patch, similarly to the Hessian-Affine detector. In this approach, the keypoint selection criterion is based on the shape of the estimated ellipses, rejecting the cases where the estimated ellipses are more elongated than a predefined axis ratio threshold.

Finally, Laguna et al. [2019] propose a hybrid-CNN architecture for keypoint detection, where a part of the parameters is trainable, while the other is manually initialized to compute first and second-order image gradients or combinations of these. These hand-crafted convolutional filters are inspired by classic hand-crafted detectors such as Harris and Stephens [1988] or Mikolajczyk and Schmid [2004]. Their architecture also incorporates a multi-scale approach, where the same network is first applied on an image pyramid, and a final trainable layer produces the output 2D response map from the fused features, as illustrated in Fig. 2-8c. For keypoint selection, they propose to subdivide the image along a grid and select at most one keypoint for each grid cell by using an Index Proposal operation, which is equivalent to the soft-argmax operation used by Yi et al. [2016]. They also propose a multi-scale extension that is less dependent on the chosen grid size.

2.1.5 Jointly-trained detectors and descriptors

In the previous two sections, we presented trainable keypoint detectors and descriptors, which were developed and trained independently. Given the success of end-to-end training for problems such as image classification, it is natural to ask whether it is possible to train jointly a keypoint detector and a descriptor in an end-to-end way. Recent work by Yi et al. [2016], Ono et al. [2018], DeTone et al. [2018], Dusmanu et al. [2019] and Revaud et al. [2019] delve into this problem and propose different

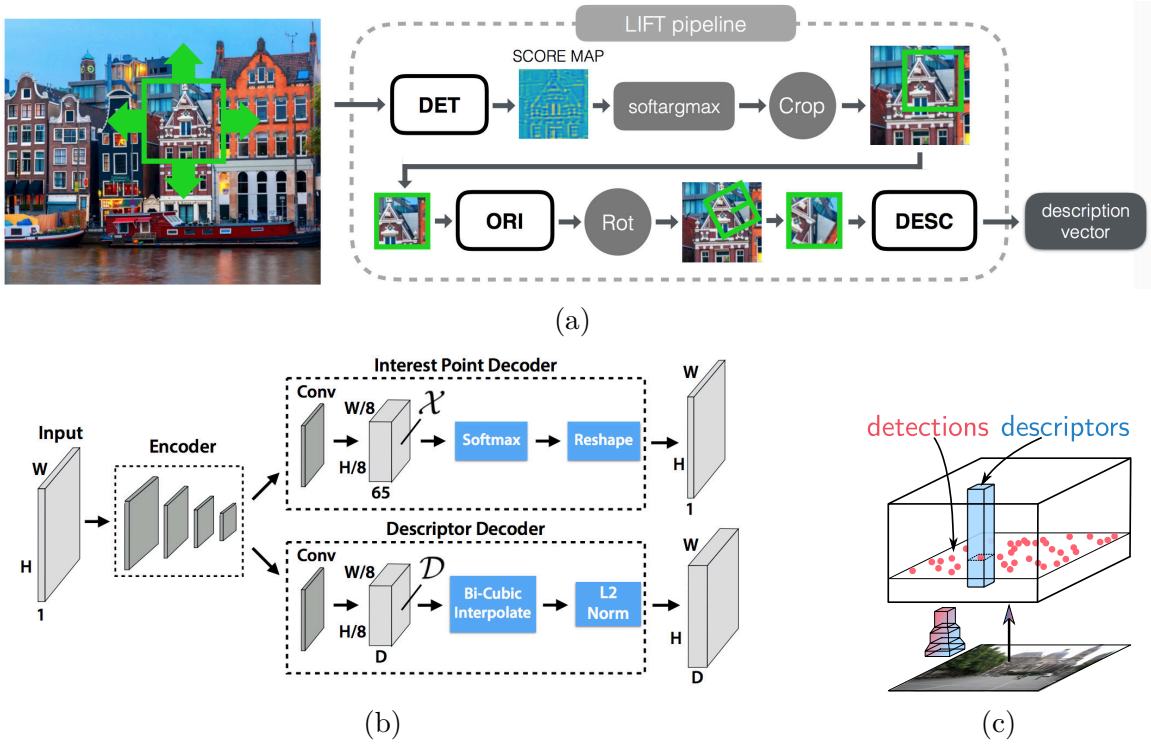


Figure 2-9: **Different jointly-trainable detectors and descriptors.** (a) The LIFT modular pipeline from Yi et al. [2016] contains trainable detection, orientation, and description modules. (b) The Superpoint method of DeTone et al. [2018] uses a shared encoder, from which both detections and descriptions are computed by two independent decoder heads. (c) The D2-Net method of Dusmanu et al. [2019] extracts a deep 3D feature map which has a dual interpretation as a set of densely extracted 1D descriptor vectors, and a set of stacked 2D detection response maps. Figures reproduced from the respective works.

approaches for its solution. Some of these methods are illustrated in Fig. 2-9.

Yi et al. [2016] proposed the LIFT method which follows the modular pipeline used in classic hand-crafted methods, but using trainable modules for detection, orientation estimation and description, as illustrated in Fig. 2-9a. However, training such a complex pipeline in an end-to-end way is non-trivial. Therefore the authors resort to a sequential training scheme, where the descriptor module is trained first, followed by the orientation module (which is conditioned on the learnt descriptor), and finally, the detection module is trained (conditioned on the other two learnt modules). For training, local image patches extracted from a 3D reconstruction dataset are used. While this provides diverse training data it also introduces a bias in the detector,

as the locations of the learnt detections are constrained to lie on the same patches as the DoG detections from the 3D reconstruction, due to the patch-based training scheme. During evaluation, the detection model is decoupled and run over the full input image in a multi-scale fashion, obtaining a 2D response map for each scale. Then, a non-maximum suppression (NMS) operation is used to select a sparse set of keypoints, from which patches are extracted and fed to orientation and description modules, following the classical hand-crafted methods.

Ono et al. [2018] propose LF-Net which extends the approach of Yi et al. [2016], but incorporates both the orientation estimation and the multi-scale treatment into the detector module. In this way, it can operate on full images in both training and evaluation. Ono et al. [2018] also propose different training losses for training the detector and descriptor modules. In particular, they propose an image-level detector loss which minimizes the error between the 2D response maps of both images. This approach is more general and does not constrain the keypoint locations to lie in a neighbourhood of the DoG keypoints, as in the case of LIFT.

DeTone et al. [2018] take a different path, and instead of following the classic sequential pipeline of first detecting keypoints and then computing descriptors, they propose a CNN which computes a common representation from which *both* detectors and descriptors are computed in parallel by two different output decoder heads, as illustrated in Fig. 2-9b. This parallelisation of the model facilitates training, as heads can be trained independently with different losses, with less interference than in the sequential model. On the contrary, the computational cost is increased as descriptors must be extracted densely (albeit with a downscaling factor of 1/8 with respect to the input resolution). A similar approach is followed by Revaud et al. [2019], but they propose a different training scheme to maximize the repeatability of the detections and discriminativeness of the keypoints.

The approach proposed by Dusmanu et al. [2019] is also similar to the one of DeTone et al. [2018]. However, it only features a single head which has a dual interpretation as a densely extracted set of 1D descriptors or a stack of 2D detection responses, as illustrated in Fig. 2-9c. In this way, all parameters are shared between the

detector and descriptor. Note that this approach bears a resemblance with the early method of Schmid and Mohr [1997] using rotational differential invariants, where each component of the descriptor could be seen as a blob detector (*cf.* Fig. 2-4).

2.1.6 Densely extracted descriptors

Up to this point, we have described approaches for obtaining sparse local features, which are typically obtained by first, detecting interest points, and then, by computing descriptors around these interest points. This two-stage approach offers several advantages. On the one side, the detector, which is typically computationally inexpensive, can be run at a high-resolution, producing very accurately localized keypoints (sometimes even with sub-pixel accuracy). On the other side, the detection stage produces a sparse set of image keypoints to be described and matched, therefore reducing the computational burden of these two subsequent stages. However, under strong illumination changes, detectors can suffer from low repeatability, which hinders the subsequent correspondence estimation task.

In order to overcome this issue, it is possible to forego the detection stage and instead compute descriptors densely over a coarse grid on the input image. This approach has been found beneficial for tasks which require matching under strong-illumination changes, such as large-scale visual search [Torii et al., 2015; Arandjelović et al., 2016; Noh et al., 2017]. Recently, densely extracted features have been also employed directly for 3D computer vision tasks, such as 3D reconstruction [Widya et al., 2018], indoor localization and camera pose estimation [Taira et al., 2018], and outdoor localization with night-time queries [Germain et al., 2019; Sattler et al., 2018].

Note that while densely extracted features used for image retrieval are typically computed on a coarse low-resolution grid (*e.g.* 40×30), 3D computer vision tasks will typically require more accurately localized points. Extracting dense and accurately localized features, while keeping computational and memory requirements reasonable is one of the challenges of this approach. Very recently, Germain et al. [2019] have proposed to use a hybrid approach where, given two images to be matched, descriptors are computed over detected keypoints in one image, but extracted densely over the

other image.

In the following chapters of this thesis, we adopt the approach of using densely extracted features for matching, to be able to handle large appearance changes, as in the cases of day-to-night matching, or category-level matching. Furthermore, in chapter 6 we propose an approach for improving the localization accuracy of the obtained matches.

2.1.7 Filtering incorrect matches

In order to obtain correspondences using local image features, putative or tentative matches between descriptors are obtained by nearest-neighbour search, with the Euclidean distance. However, this typically yields a portion of incorrect matches. Therefore, different heuristics or filtering techniques have been proposed to improve the fraction of correct matches in the set of tentative matches.

Second nearest-neighbour test. Lowe [2004] proposes a criterion for eliminating ambiguous matches, as determined by the distance of one descriptor to the first and second nearest-neighbours in the set of descriptors of the other image. If the ratio of distances to the first and second nearest neighbours is above a given threshold (typically around 0.8), then the match is deemed too ambiguous and rejected. One drawback of this technique is that the radio threshold hyperparameter needs to be manually adjusted for each type of descriptor. This approach is illustrated in Fig. 2-10a.

Mutual correspondence test. An alternative criterion is to enforce mutual correspondence (or cross correspondence) as described by Schmid [1996]. The approach consists in rejecting those matches where the descriptors are not *mutually* nearest-neighbours when considering the match assignment in both directions. One advantage of this method is that, contrary to the ratio test, it does not require selecting any hyperparameters. This approach is illustrated in Fig. 2-10b.

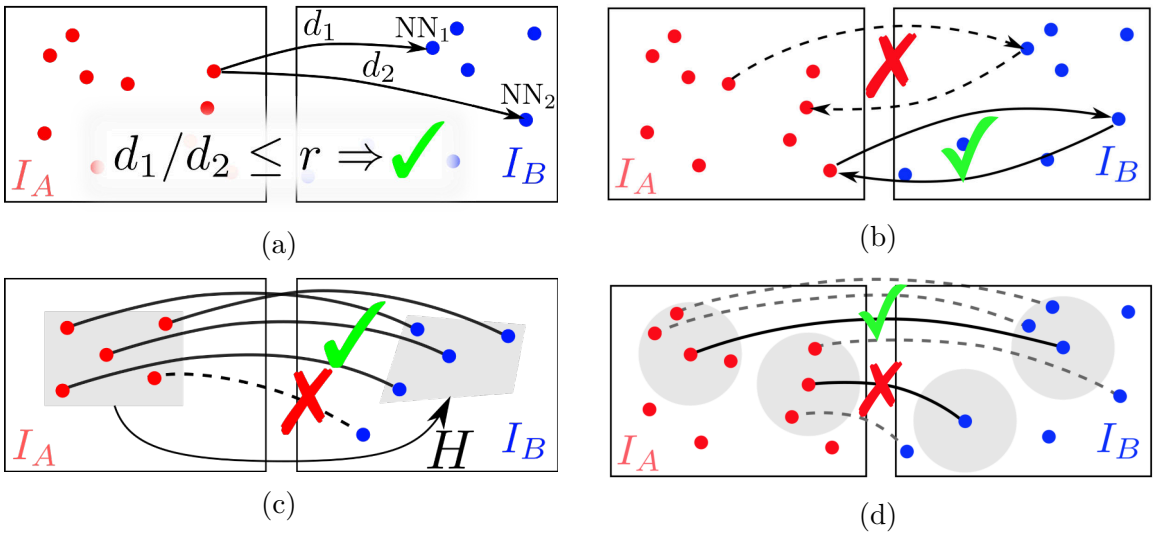


Figure 2-10: **Methods for filtering incorrect matches.** (a) Lowe's second nearest-neighbour test (or ratio test) retains matches where the ratio of distances to the first and second nearest-neighbours is below a threshold. (b) The mutual correspondence test retains correspondences that are mutual when matching in both directions. (c) RANSAC can be used to robustly fit a geometric model (such as homography in the case of planar objects) and reject the matches that are not in agreement with the model. (d) Neighbourhood consensus retains correspondences which have a number of coherent supporting matches in their neighbourhoods.

Global geometric constraints. In many cases, we know by prior knowledge that correspondences should respect a particular geometric constraint. For instance, if the imaged scene is planar, then correspondences between two images should be related by a homography. Or if the scene has an arbitrary 3D structure but is static, then the epipolar constraint must be satisfied. In these cases, the RANSAC algorithm by Fischler and Bolles [1981] can be used to robustly estimate the parameters of these geometric models, while removing the matches that are not in agreement with it, which are typically the incorrect ones. This method has been widely successful and is employed in many algorithms for pose estimation and 3D reconstruction, among others. Several variants and extensions have been proposed [Chum et al., 2003, 2005; Sattler et al., 2009; Lebeda et al., 2012]. This approach is illustrated in Fig. 2-10c.

Neighbourhood consensus. A drawback of the RANSAC method is that a global geometric constraint should be satisfied by all matches. While epipolar geometry can

be used for static scenes, it is not a correct model for scenes which contain objects moving rigidly (*e.g.* cars) or non-rigidly (*e.g.* walking people). A different approach, called neighbourhood consensus, is based on establishing semi-local geometric constraints instead of a global geometric constraint. In this approach, the criterion used to filter matches relies on the local supporting evidence from the neighbouring matches, as illustrated in Fig. 2-10d. The criterion used for retaining or rejecting matches can be based on patterns of distances [Zhang et al., 1995], angles between neighbouring matches [Schmid and Mohr, 1997], or simply by counting the number of consistent matches in a certain image neighbourhood [Schaffalitzky and Zisserman, 2002a; Sivic and Zisserman, 2003; Sattler et al., 2009; Bian et al., 2017]. While simple, these techniques have been remarkably effective in removing incorrect matches and disambiguating local repetitive patterns [Sattler et al., 2009].

Trainable match filtering methods. Recently, trainable approaches have also been proposed for the task of filtering local feature correspondences [Brachmann and Rother, 2019; Yi et al., 2018; Sarlin et al., 2019; Zhang et al., 2019]. Yi et al. [2018] propose a neural-network architecture that operates on 4D match coordinates and classifies each correspondence as either correct or incorrect. Brachmann and Rother [2019] propose the Neural-guided RANSAC, which extends the previous method to produce weights instead of classification labels, which are used to guide RANSAC sampling. Zhang et al. [2019] also extend the work of Yi *et al.* in their proposed Order-Aware Networks, which capture the local context by clustering 4D correspondences into a set of ordered clusters, and the global context by processing these clusters with a multi-layer perceptron. Finally, Sarlin et al. [2019] propose to use a graph neural network followed by an optimisation procedure to estimate correspondences between two set of local features.

In chapter 5, we propose an end-to-end trainable method for feature extraction, matching and match filtering. Our model implements the neighbourhood consensus principle as a 4D convolutional neural network that operates on the 4D space of correspondences and allows *learning* the geometric patterns of correspondence that

allow differentiating a correct match from an incorrect one. In chapter 6, we revisit this approach and propose improvements to make it more efficient, as well as for obtaining better-localized matches.

2.2 Category-level matching

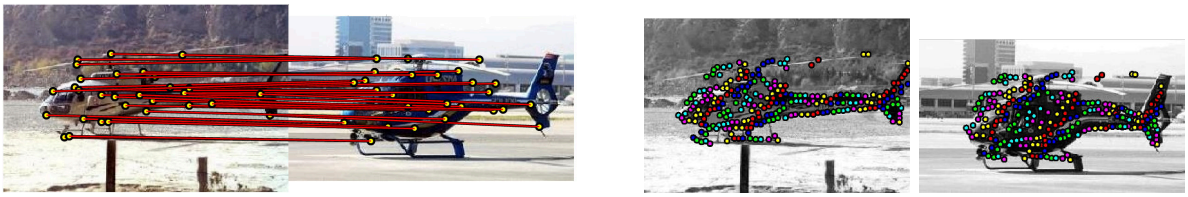
In this section, we present recent work for *category-level matching* (also called semantic matching). In this scenario, we want to find correspondences between images showing different instances of objects from the same object categories. We begin by reviewing the earliest methods based on hand-crafted descriptors in Sec. 2.2.1. Then, in Sec. 2.2.2 we review more recent methods that employ CNN features, both as stand-alone pre-trained features or as part of a trainable pipeline.

2.2.1 Methods based on hand-crafted descriptors

Early methods for category-level matching employed hand-crafted descriptors like SIFT [Lowe, 2004] or HOG [Dalal and Triggs, 2005] together with optimization approaches for image alignment, based on the minimization of a given energy function.

Berg et al. [2005] match sparse features extracted on edges, and propose an approximate solution to an integer quadratic programming (IQP) problem for finding the match assignments which jointly minimize an appearance and distortion cost. Their algorithm produces a small set of geometrically consistent correspondences, which can later be used to fit a thin-plate spline model for obtaining dense correspondences. For feature description, the authors develop a descriptor based on the Geometric Blur operation by Berg and Malik [2001], previously used for template matching. A qualitative example is presented in Fig. 2-11a.

Liu et al. [2011] propose SIFT Flow, which adopts the computational framework of optical flow, but employs more general SIFT features instead of RGB values. Contrary to the approach of Berg et al. [2005] which uses sparse descriptors extracted on edges, the SIFT Flow method uses densely extracted descriptors, computed at each image pixel. This makes match assignment computationally expensive as the search space of



Sparse matches Dense matches (from fitting a thin-plate spline)

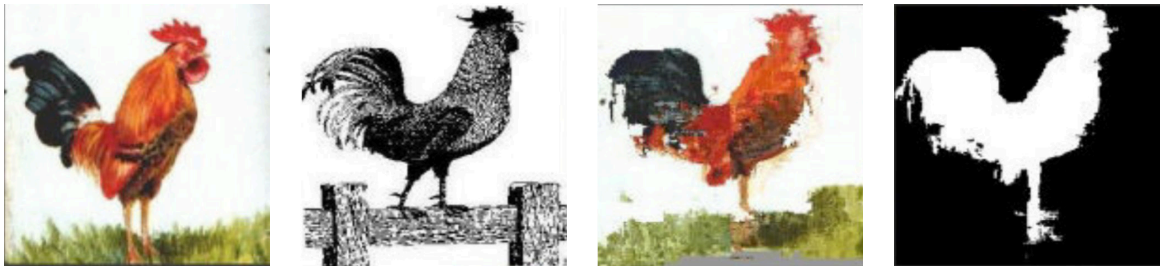
(a) Sparse matching method of Berg et al. [2005] using IQP.

Source image Target image Output Flow field

(b) SIFT Flow method of Liu et al. [2011].

Source image Target image Output Flow field

(c) Graph matching kernel method of Duchenne et al. [2011].



Source image Target image Output Transferred mask

(d) DSP method of Kim et al. [2013]

Source image Target image Output Flow field

(e) Proposal Flow method of Ham et al. [2017]

Figure 2-11: **Sample results from category-level matching methods using hand-crafted features.**

possible correspondences is very large. In order to address this issue, Liu et al. [2011] propose a coarse-to-fine matching scheme which, first, computes correspondences between the lowest resolution levels of two image pyramids built from the input images and, then, progressively processes the higher-resolution levels using the previous result as initialization. Speed-ups are obtained by only performing exhaustive matching at the lowest resolution, and searching in a local 11×11 window in the higher pyramid levels. Experiments show that this approach does not only improve runtime but also leads to better solutions. A qualitative example is presented in Fig. 2-11b.

The graph matching kernel (GMK) approach of Duchenne et al. [2011] could be seen as an intermediate approach between the sparse matching method of Berg et al. [2005] and the dense SIFT Flow method of Liu et al. [2011]. While, similarly to SIFT Flow, the approach of Duchenne et al. [2011] uses features extracted on a grid, these are not extracted at individual pixel positions but on a coarse grid of resolution 30×40. These densely extracted descriptors are then matched by formulating the problem as a graph matching problem and solving the optimization using an extension of the graph cut method of Ishikawa [2003]. Note that while they use a single scale for optimization with a search window of 11×11 features, the actual receptive field of each feature is much larger than in SIFT Flow, which allows the method to still be able to handle large displacements. This work uses the descriptors of Boureau et al. [2010], which are based on the sparse coding of SIFT descriptors. A qualitative example is presented in Fig. 2-11c.

Kim et al. [2013] propose the Deformable Spatial Pyramid (DSP) method, which uses a similar multi-scale approach to SIFT Flow, but has two main differences. First, while the image pyramids used in SIFT Flow are constructed by progressively subsampling the original image, the spatial pyramid in DSP is defined in the opposite order. Following the approach initially proposed by Lazebnik et al. [2006] for image classification, Kim et al. [2013] use a spatial pyramid which starts from a single cell which spans the whole image, and which is progressively subdivided into four smaller cells at each additional level of the pyramid. A final level is then added, where each pixel is considered a cell. Then, a graph is constructed where each node corresponds

to a cell, and edges connect neighbouring cells of the same level (except in the final level) as well as between parent and child cells. The second main difference is that while in SIFT Flow the different pyramid-levels are processed sequentially, in DSP the matches between the multi-level graphs are optimized jointly. Compared to SIFT Flow, the DSP approach achieves better matching accuracy while being noticeably faster. This work uses SIFT descriptors aggregated at each cell for matching. A qualitative example is presented in Fig. 2-11d.

While the previous methods computed correspondences between features extracted at keypoints or on uniform grids, the Proposal Flow method of Ham et al. [2017] performs matching over features computed on category-independent multi-scale region proposals, which had been successfully employed for object detection [Girshick et al., 2014]. These proposals can contain full objects or object parts, as well as salient background regions. Contrary to previous approaches which relied on the optimization of an energy function for matching, Ham et al. [2017] propose to compute the most probable matches through a bayesian formulation. Their best performing approach, called *local offset matching*, uses a principle which is close to neighbourhood consensus to model the probabilities of matching regions. Once matches between region proposals are established, a flow field can be computed and used to align the input images. Proposal Flow uses HOG descriptors [Dalal and Triggs, 2005] extracted on 8×8 patches. A qualitative example is presented in Fig. 2-11e.

2.2.2 Methods based on CNNs

Following the success of CNNs for image classification, a lot of research effort was devoted to gaining a deeper understanding of their inner workings. One of the questions that arose was whether their intermediate representations could be used for estimating correspondences.

Long et al. [2014] studied such problem and showed that it was possible to replace the densely extracted SIFT features in the SIFT Flow method by intermediate CNN features obtaining similar results. For this, they used `conv4` features from a pretrained AlexNet CNN model, originally used for image classification [Krizhevsky et al., 2012].

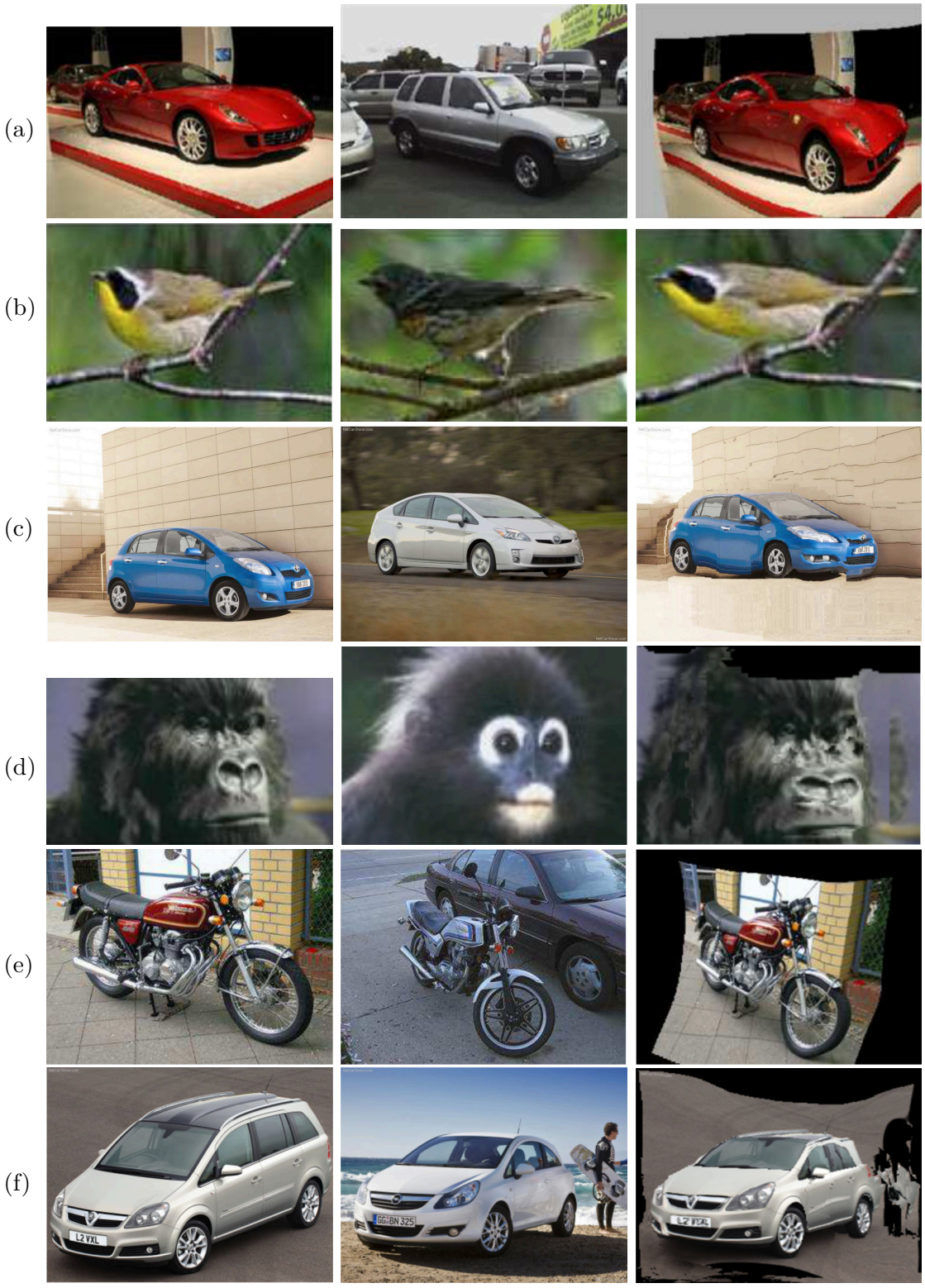


Figure 2-12: **Sample results from category-level matching methods using CNN features.** (a) Method of Ufer and Ommer [2017]. (b) WarpNet method of Kanazawa et al. [2016]. (c) FCSS method of Kim et al. [2018b]. (d) AnchorNet features of Novotny et al. [2017] within the Proposal Flow framework of Ham et al. [2017]. (e) PARN method of Jeon et al. [2018]. (f) RTN method of Kim et al. [2018a].

Ufer and Ommer [2017] use the same AlexNet `conv4` features, but extract them in a multi-scale fashion using an image pyramid. Furthermore, instead of matching all densely extracted features, they propose a keypoint selection method based on thresholding the aggregated activation values across all feature channels and performing non-maximum suppression over the feature's 2D entropy map. Finally, they select a fixed number of top features for each scale. After individual multi-scale features have been extracted using this approach, they propose to match them using an Integer Quadratic Program, followed by a thin-plate spline model fitting to obtain dense correspondences (similarly to the early approach by Berg et al. [2005]). A qualitative example of the method of Ufer and Ommer [2017] is shown in Fig. 2-12a.

While these methods used CNN features for estimating correspondences, these are pretrained features on image classification and are treated as fixed feature extractors. Furthermore, these methods do not propose to learn any additional parameters but are rather based on optimization schemes for match assignment. Other works have gone further, both by attempting to learn better descriptors for the matching task specifically, or by learning to assign correspondences instead of relying on an optimization scheme.

Kanazawa et al. [2016] propose a Siamese architecture which regresses the parameters of a thin-plate-spline (TPS) transformation and show that their model can be trained from weak-supervision using only foreground object masks. A qualitative example is presented in Fig. 2-12b. For training, they use a weakly-supervised approach to obtain a set of natural geometric deformations from a fine-grained dataset of birds, and a self-supervised approach to generate synthetically warped images by sampling from this set of deformations. Finally, corresponding foreground points on the pairs of synthetically warped images are used for computing the training loss. The method we present in Chapter 3 is closely inspired by this approach but uses a different operation for fusing both branches of the Siamese network and can handle category-level matching with multiple classes of objects, as well as instance-level matching.

Kim et al. [2018b] propose to combine pretrained CNN layers with additional layers

which are trained from scratch to compute their fully convolutional self-similarity features (FCSS), which are densely extracted features trained specifically for category-level matching. For training, they propose a weakly-supervised scheme where positive and negative training pairs are collected at each training step by, first, computing putative matches within object bounding boxes by k-NN search and, then, filtering these matches with the mutual correspondence criterion. The pairs of features that satisfy the cross-correspondence criterion are then used as positive pairs, while those which do not are used as negative pairs. These positive and negative pairs are then employed by a contrastive loss to train the model. After the model is trained, FCSS features can be extracted for a pair of images, and correspondences can be computed directly by nearest-neighbour search. A qualitative example of the alignment obtained by matching FCSS features is shown in Fig. 2-12c. The follow-up work [Kim et al., 2017] uses FCSS features within an optimization framework to obtain smoother correspondence fields than those obtained from nearest-neighbour matching alone.

Han et al. [2017] propose the SCNet model that extends the Proposal Flow method of Ham et al. [2017]. As in Proposal Flow, SCNet also employs region proposals for estimating category-level correspondences. However, instead of computing region proposals similarities by comparing HOG features, they propose to compute deep CNN descriptors for each region proposal (similarly to R-CNN Girshick et al. [2014]), which are then processed by an additional fully-connected layer and compared using a rectified cosine similarity. The model is trained using a hinge loss with an L2 regularization term on the similarity matrix, where pairs of positive and negative proposals are defined by exploiting the manually annotated keypoints of the PF-PASCAL dataset.

Novotny et al. [2017] propose the AnchorNet CNN architecture for learning semantic object parts using very weak image-level supervision, where only class labels are required. Their CNN architecture first extracts dense hypercolumn descriptors, which are initialized using a pretrained image classification model. Then, class-specific convolutional filters are used to discover diverse and discriminative object parts. In addition, they propose to combine these class-specific filters by means of a denoising autoencoder to obtain more general class-agnostic features. Besides showing that

meaningful object parts can be learnt from such weak supervision, they show that their learnt class-agnostic features can be used for semantic keypoint transfer, by replacing HOG features in the Proposal Flow framework (yielding similar performance), or SIFT features in DSP (yielding superior performance). A qualitative example of the alignment obtained by using AnchorNet features within the Proposal Flow framework is shown in Fig. 2-12d.

In the follow-up work of Novotny et al. [2018], self-supervision is used to train a Siamese CNN model for image matching, where supervision is provided in the form of synthetically warped natural images. In addition to the estimated dense feature maps, the model computes a 2D scalar confidence map for each image, which estimates the belief that a reliable correspondence can be established using the features at each position, serving a similar purpose as a feature detector. Interestingly, while the model is trained for the matching task only, some channels of the learnt features activate on distinct object parts, showing a duality between obtaining reliable correspondences and learning object keypoints.

While the previous methods focused on learning descriptors for category-level matching [Kim et al., 2018b; Novotny et al., 2017, 2018] or a similarity metric between them [Han et al., 2017], other methods propose to additionally learn to estimate a transformation between two input images.

Jeon et al. [2018] propose the PARN model which uses the spatial pyramid representation from Kim et al. [2013], and uses a stack of Siamese CNNs for estimating an affine transformation field at each cell of the pyramid. While a global affine transformation is estimated for the top cell of the pyramid, residual affine transformations are estimated for all cells on the lower levels. Similarly to the FCSS method [Kim et al., 2018b], supervision is obtained by finding candidate matches and filtering them with the mutual correspondence test. A qualitative example of the PARN method is shown in Fig. 2-12e.

Kim et al. [2018a] propose the RTN model, which uses a similar Siamese CNN architecture for estimating an affine transformation field, but proceeds in a multi-scale recurrent approach that resembles that of SIFT Flow. On a first iteration, features

are extracted with a large stride parameter, which results in a high subsampling factor. Then, these features are matched using a correlation operation and fed into a transformation estimation network that regresses a coarse affine field. In the subsequent iterations, the stride parameter is progressively decreased, and residual and more precisely localized transformations are estimated. Note that contrary to the PARN model which contained a stack of CNNs for processing each cell of the spatial pyramid, a single CNN model is used recurrently in RTN. The network is trained using the weakly-supervised scheme of the FCSS method [Kim et al., 2018b] to generate positive and negative pairs but using a different classification-based loss. A qualitative example of the RTN method is shown in Fig. 2-12f. A similar iterative approach is presented in Chapter 3.

Very recently, Lee et al. [2019] proposed the SFNet method which can estimate a dense flow field for semantic image alignment. Similarly to the Siamese CNN model in RTN, dense features are extracted for both images and matched using a correlation operation. However, the SFNet propose to use a *kernel soft argmax* operation to compute a dense flow field from the correlation values, instead of regressing the parameters of an affine model. For training, synthetically warped images are used, as well as their foreground masks. The supervision is given by a mask consistency loss, a flow consistency loss (which favours mutual correspondence) and a smoothness loss which operates as a regularization term.

Chapter 3

CNN architecture for geometric matching

In this chapter, we develop a trainable architecture for estimating correspondences between a pair of images. However, instead of estimating a sparse set of correspondences by matching local image features, as is often done in instance-level matching, we adopt the approach of estimating the parameters of a geometric model such as affine, homography or thin-plate spline. These geometric transformations offer a strong regularization, which can be helpful for finding correspondences under large appearance changes, such as in the case of category-level matching.

The contributions of this chapter are three-fold. First, we propose a convolutional neural network architecture for geometric matching. The architecture is based on three main components that mimic the standard steps of feature extraction, matching and simultaneous inlier detection and model parameter estimation while being trainable end-to-end. Second, we demonstrate that the network parameters can be trained from synthetically generated imagery without the need for manual annotation and that our matching layer significantly increases generalization capabilities to never seen before images. Finally, we show that the same model can perform both instance-level and category-level matching giving state-of-the-art results on the challenging PF, TSS and Caltech-101 datasets.

3.1 Introduction

Estimating correspondences between images is one of the fundamental problems in computer vision [Forsyth and Ponce, 2002; Hartley and Zisserman, 2003] with applications ranging from large-scale 3D reconstruction [Agarwal et al., 2011] to image manipulation [HaCohen et al., 2011] and semantic segmentation [Rubinstein et al., 2013]. Traditionally, correspondences consistent with a geometric model such as epipolar geometry or planar affine transformation, are computed by detecting and matching local features (such as SIFT [Lowe, 2004] or HOG [Dalal and Triggs, 2005; Ham et al., 2017]), followed by pruning incorrect matches using local geometric constraints [Schmid and Mohr, 1997; Sivic and Zisserman, 2003] and robust estimation of a global geometric transformation using algorithms such as RANSAC [Fischler and Bolles, 1981] or Hough transform [Hough, 1962; Lamdan et al., 1988; Leibe et al., 2008; Lowe, 2004]. This approach works well in many cases but fails in situations that exhibit (i) large changes of depicted appearance due to e.g. intra-class variation [Ham et al., 2017], or (ii) large changes of scene layout or non-rigid deformations that require complex geometric models with many parameters which are hard to estimate in a manner robust to outliers.

In this chapter we build on the traditional approach and develop a convolutional neural network (CNN) architecture that mimics the standard matching process. First, we replace the standard local features with powerful trainable convolutional neural network features [Krizhevsky et al., 2012; Simonyan and Zisserman, 2015], which allows us to handle large changes of appearance between the matched images. Second, we develop trainable matching and transformation estimation layers that can cope with noisy and incorrect matches in a robust way, mimicking the good practices in feature matching such as the second nearest-neighbour test [Lowe, 2004], neighbourhood consensus [Schmid and Mohr, 1997; Sivic and Zisserman, 2003] and Hough transform-like estimation [Hough, 1962; Lamdan et al., 1988; Leibe et al., 2008; Lowe, 2004].

The outcome is a convolutional neural network architecture trainable for the end-task of geometric matching, which can handle large appearance changes, and is

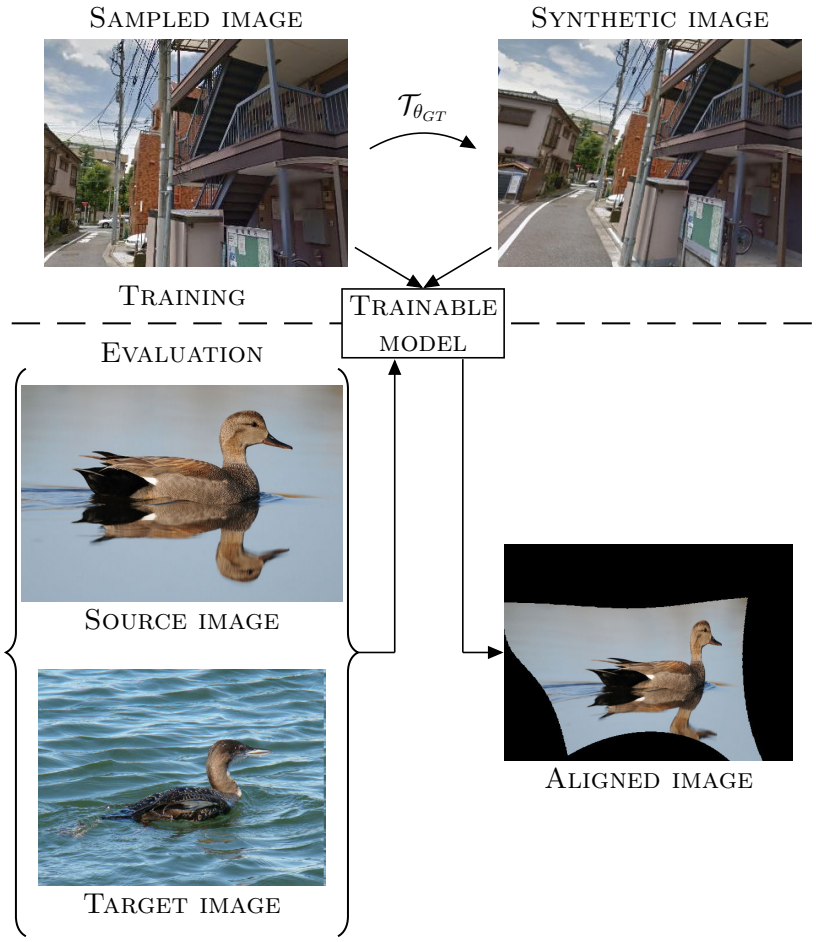


Figure 3-1: **Top:** The proposed model can be trained from synthetic image pairs, avoiding the need for manual annotation. **Bottom:** At evaluation time, the trained geometry estimation network automatically aligns two images with substantial appearance differences. It is able to estimate large deformable transformations robustly in the presence of clutter.

therefore suitable for both instance-level and category-level matching problems.

The contributions of this chapter are three-fold. First, we propose a convolutional neural network architecture for geometric matching, which mimics the standard steps of feature extraction, matching and simultaneous inlier detection and model parameter estimation, while being trainable end-to-end. Second, we demonstrate that the network parameters can be trained from synthetically generated imagery without the need for manual annotation and that our matching layer significantly increases generalization capabilities to never seen before images. Finally, we show that the same model can give state-of-the-art results on several challenging datasets for category-level image

alignment. Our approach is illustrated in Fig. 3-1.

All training and evaluation code, as well as our trained networks, are available at http://www.di.ens.fr/willow/research/cnngeometric/.

3.2 Related work

The classical approach for finding correspondences involves identifying interest points and computing local descriptors around these points [Harris and Stephens, 1988; Schmid and Mohr, 1997; Lowe, 2004; Mikolajczyk and Schmid, 2002; Lowe, 2004; Berg et al., 2005; Bay et al., 2006]. While this approach performs relatively well for instance-level matching, the feature detectors and descriptors lack the generalization ability for category-level matching.

Recently, convolutional neural networks have been used to learn powerful feature descriptors which are more robust to appearance changes than the classical descriptors [Jahrer et al., 2008; Simo-Serra et al., 2015; Han et al., 2015; Zagoruyko and Komodakis, 2015; Balntas et al., 2016a; Noh et al., 2017]. However, these works are focused on learning descriptors [Jahrer et al., 2008; Simo-Serra et al., 2015; Balntas et al., 2016a; Noh et al., 2017], or a similarity measure between descriptors [Han et al., 2015; Zagoruyko and Komodakis, 2015; Altwaijry et al., 2016], and do not target the problem of finding the transformation relating the two input images. In this chapter, we go a step further from CNN descriptors, and seek to also learn to estimate the geometric transformation.

Related are also network architectures for estimating inter-frame motion in video [Weinzaepfel et al., 2013; Fischer et al., 2015; Thewlis et al., 2016] or instance-level homography estimation [DeTone et al., 2016], however their goal is very different from ours, targeting high-precision correspondence with very limited appearance variation and background clutter. Closer to us is the network architecture of [Kanazawa et al., 2016] which, however, tackles a different problem of fine-grained category-level matching (different species of birds) with limited background clutter and small translations and scale changes, as their objects are largely centered in the image. In addition, their

architecture is based on a different matching layer, which we show not to perform as well as the matching layer used in our work.

Some works, such as [Berg et al., 2005; Liu et al., 2011; Duchenne et al., 2011; Kim et al., 2013; Long et al., 2014; Ham et al., 2017], have addressed the hard problem of category-level matching, but rely on traditional non-trainable optimization for matching [Berg et al., 2005; Liu et al., 2011; Duchenne et al., 2011; Kim et al., 2013; Long et al., 2014], or guide the matching using object proposals [Ham et al., 2017]. On the contrary, our approach is fully trainable in an end-to-end manner and does not require any optimization procedure at evaluation time, or guidance by object proposals.

Others [Learned-Miller, 2006; Shokrollahi Yancheshmeh et al., 2015; Zhou et al., 2015] have addressed the problems of instance and category-level correspondence by performing joint image alignment. However, these methods differ from ours as they: (i) require class labels; (ii) don't use CNN features; (iii) jointly align a large set of images, while we align image pairs; and (iv) don't use a trainable CNN architecture for alignment as we do.

3.3 Architecture for geometric matching

In this section, we introduce a new convolutional neural network architecture for estimating parameters of a geometric transformation between two input images. The architecture is designed to mimic the classical computer vision pipeline (e.g. [Philbin et al., 2007]), while using differentiable modules so that it is trainable end-to-end for the geometry estimation task. The classical approach consists of the following stages: (i) local descriptors (e.g. SIFT) are extracted from both input images, (ii) the descriptors are matched across images to form a set of tentative correspondences, which are then used to (iii) robustly estimate the parameters of the geometric model using RANSAC or Hough voting.

Our architecture, illustrated in Fig. 3-2, mimics this process by: (i) passing input images I_A and I_B through a Siamese architecture consisting of convolutional layers,

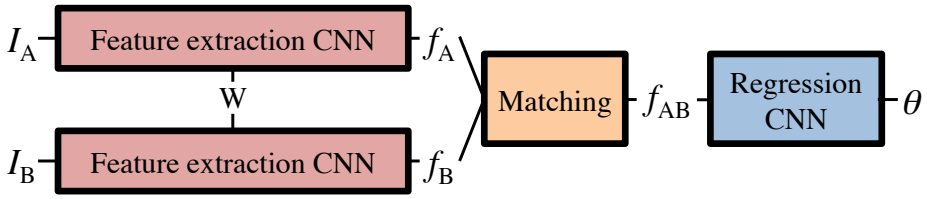


Figure 3-2: **Diagram of the proposed architecture.** Images I_A and I_B are passed through feature extraction networks which have tied parameters W, followed by a matching network which matches the descriptors. The output of the matching network is passed through a regression network which outputs the parameters of the geometric transformation.

thus extracting feature maps f_A and f_B which are analogous to dense local descriptors, (ii) matching the feature maps (densely extracted descriptors) across images into a tentative correspondence map f_{AB}, followed by a (iii) regression network which directly outputs the parameters of the geometric model, θ, in a robust manner. The inputs to the network are the two images, and the outputs are the parameters of the chosen geometric model, e.g. a 6-D vector for an affine transformation.

In the following, we describe each of the three stages in detail.

3.3.1 Feature extraction

The first stage of the pipeline is feature extraction, for which we use a standard CNN architecture. A CNN without fully connected layers takes an input image and produces a feature map $f \in \mathbb{R}^{h \times w \times d}$, which can be interpreted as a $h \times w$ dense spatial grid of d-dimensional local descriptors. A similar interpretation has been used previously in instance retrieval [Azizpour et al., 2015; Babenko and Lempitsky, 2015; Gong et al., 2014; Arandjelović et al., 2016] demonstrating high discriminative power of CNN-based descriptors. Thus, for feature extraction we use the VGG-16 network [Simonyan and Zisserman, 2015], cropped at the `pool4` layer (before the ReLU unit), followed by per-feature L2-normalization. We use a pre-trained model, originally trained on ImageNet [Deng et al., 2009] for the task of image classification. As shown in Fig. 3-2, the feature extraction network is duplicated and arranged in a Siamese configuration such that the two input images are passed through two identical

networks which share parameters.

3.3.2 Matching network

The image features produced by the feature extraction networks should be combined into a single tensor as input to the regressor network to estimate the geometric transformation. We first describe the classical approach for generating tentative correspondences, and then present our matching layer which mimics this process.

Tentative matches in classical geometry estimation. Classical methods start by computing similarities between all pairs of descriptors across the two images. From this point on, the original descriptors are discarded as all the necessary information for geometry estimation is contained in the pairwise descriptor similarities and their spatial locations. Secondly, the pairs are pruned by either thresholding the similarity values, or, more commonly, by only keeping the matches which involve the nearest (most similar) neighbours. Furthermore, the second nearest-neighbour test [Lowe, 2004] prunes the matches further by requiring that the match strength is significantly stronger than the second best match involving the same descriptor, which is very effective at discarding ambiguous matches.

Matching layer. Our matching layer applies a similar procedure. Analogously to the classical approach, only descriptor similarities and their spatial locations should be considered for geometry estimation, and not the original descriptors themselves.

To achieve this, we propose to use a *correlation layer* followed by normalization. Firstly, all pairs of similarities between descriptors are computed in the correlation layer. Secondly, similarity scores are processed and normalized such that ambiguous matches are strongly down-weighted.

In more detail, given L2-normalized dense feature maps $f_A, f_B \in \mathbb{R}^{h \times w \times d}$, the correlation map $c_{AB} \in \mathbb{R}^{h \times w \times (h \times w)}$ outputted by the correlation layer contains at each position the scalar product of a pair of individual descriptors $\mathbf{f}_A \in f_A$ and $\mathbf{f}_B \in f_B$, as detailed in Eq. (3.1).

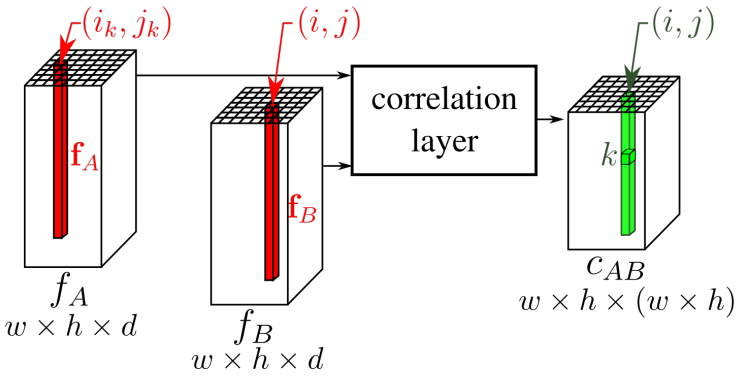


Figure 3-3: **Correlation map computation with CNN features.** The correlation map c_{AB} contains all pairwise similarities between individual features $\mathbf{f}_A \in f_A$ and $\mathbf{f}_B \in f_B$. At a particular spatial location (i, j) the correlation map output c_{AB} contains all the similarities between $\mathbf{f}_B(i, j)$ and all $\mathbf{f}_A \in f_A$.

$$c_{AB}(i, j, k) = \mathbf{f}_B(i, j)^T \mathbf{f}_A(i_k, j_k) \tag{3.1}$$

where (i, j) and (i_k, j_k) indicate the individual feature positions in the $h \times w$ dense feature maps, and $k = h(j_k - 1) + i_k$ is an auxiliary indexing variable for (i_k, j_k).

A diagram of the correlation layer is presented in Fig. 3-3. Note that at a particular position (i, j), the correlation map c_{AB} contains the similarities between $\mathbf{f}_B$ at that position and *all* the features of f_A.

As is done in the classical methods for tentative correspondence estimation, it is important to postprocess the pairwise similarity scores to remove ambiguous matches. To this end, we apply a channel-wise normalization of the correlation map at each spatial location to produce the final tentative correspondence map f_{AB}. The normalization is performed by ReLU, to zero out negative correlations, followed by L2-normalization, which has two desirable effects. First, let us consider the case when descriptor $\mathbf{f}_B$ correlates well with only a single feature in f_A. In this case, the normalization will amplify the score of the match, akin to the nearest-neighbour matching in classical geometry estimation. Second, in the case of the descriptor $\mathbf{f}_B$ matching multiple features in f_A due to the existence of clutter or repetitive patterns, matching scores will be down-weighted similarly to the second nearest-neighbour test [Lowe, 2004]. However, note that both the correlation and the normalization operations are differentiable

with respect to the input descriptors, which facilitates backpropagation thus enabling end-to-end learning.

Discussion. The first step of our matching layer, namely the correlation layer, is somewhat similar to layers used in DeepMatching [Weinzaepfel et al., 2013] and FlowNet [Fischer et al., 2015]. However, DeepMatching [Weinzaepfel et al., 2013] only uses deep RGB patches and no part of their architecture is trainable. FlowNet [Fischer et al., 2015] uses a spatially constrained correlation layer such that similarities are only computed in a restricted spatial neighbourhood thus limiting the range of geometric transformations that can be captured. This is acceptable for their task of learning to estimate optical flow, but is inappropriate for larger transformations that we consider in this chapter. Furthermore, neither of these methods performs score normalization, which we find to be crucial in dealing with cluttered scenes.

Previous works have used other matching layers to combine descriptors across images, namely simple concatenation of descriptors along the channel dimension [DeTone et al., 2016] or subtraction [Kanazawa et al., 2016]. However, these approaches suffer from two problems. First, as the following layers are typically convolutional, these methods also struggle to handle large transformations as they are unable to detect long-range matches. Second, when concatenating or subtracting descriptors, instead of computing pairwise descriptor similarities as is commonly done in classical geometry estimation and mimicked by the correlation layer, image content information is directly outputted. To further illustrate why this can be problematic, consider two pairs of images that are related with the same geometric transformation – the concatenation and subtraction strategies will produce different outputs for the two cases, making it hard for the regressor to deduce the geometric transformation. In contrast, the correlation layer output is likely to produce similar correlation maps for the two cases, regardless of the image content, thus simplifying the problem for the regressor. In line with this intuition, in Sec. 3.7.3 we show that the concatenation and subtraction methods indeed have difficulties generalizing beyond the training set, while our correlation layer achieves generalization yielding superior results.

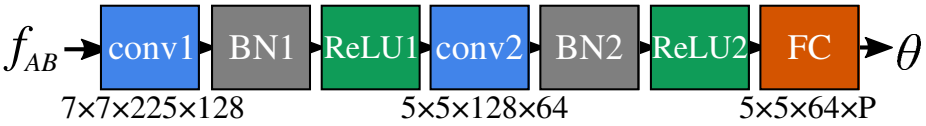


Figure 3-4: **Architecture of the regression network.** It is composed of two convolutional layers without padding and stride equal to 1, followed by batch normalization and ReLU, and a final fully connected layer which regresses to the P transformation parameters.

3.3.3 Regression network

The normalized correlation map is passed through a regression network which directly estimates parameters of the geometric transformation relating the two input images. In classical geometry estimation, this step consists of robustly estimating the transformation from the list of tentative correspondences. Local geometric constraints are often used to further prune the list of tentative matches [Schmid and Mohr, 1997; Sivic and Zisserman, 2003] by only retaining matches which are consistent with other matches in their spatial neighbourhood. Final geometry estimation is done by RANSAC [Fischler and Bolles, 1981] or Hough voting [Hough, 1962; Lamdan et al., 1988; Leibe et al., 2008; Lowe, 2004].

We again mimic the classical approach using a neural network, where we stack two blocks of convolutional layers, followed by batch normalization [Ioffe and Szegedy, 2015] and the ReLU non-linearity, and add a final fully connected layer which regresses to the parameters of the transformation, as shown in Fig. 3-4. The intuition behind this architecture is that the estimation is performed in a bottom-up manner somewhat like Hough voting, where early convolutional layers vote for candidate transformations, and these are then processed by the later layers to aggregate the votes. The first convolutional layers can also enforce local neighbourhood consensus [Schmid and Mohr, 1997; Sivic and Zisserman, 2003] by learning filters which only fire if nearby descriptors in image A are matched to nearby descriptors in image B, and we show qualitative evidence in Sec. 3.7.5 that this indeed does happen.

Discussion. A potential alternative to a convolutional regression network is to use fully connected layers. However, as the input correlation map size is quadratic in

the number of image features, such a network would be hard to train due to a large number of parameters that would need to be learned, and it would not be scalable due to occupying too much memory and being too slow to use. It should be noted that even though the layers in our architecture are convolutional, the regressor can learn to estimate large transformations. This is because one spatial location in the correlation map contains similarity scores between the corresponding feature in image B and *all* the features in image A (c.f. equation (3.1)), and not just the local neighbourhood as in [Fischer et al., 2015].

3.4 Geometric transformations

Three different parametric geometric transformations were employed in this chapter: affine, homography and thin-plate spline. The details of their parametrizations are presented next. As images are warped using the reverse mapping, the transformations map coordinates from the target image B to the source image A.

3.4.1 Affine transformation

An affine transformation is a 6 degree-of-freedom linear transformation capable of modeling translation, rotation, non-isotropic scaling and shear. It can be parametrized by a 6 dimensional vector θ_{AFF}:

$$\theta_{\text{AFF}} = [a_{11}, a_{12}, a_{21}, a_{22}, t_x, t_y], \tag{3.2}$$

such that points $P_B = [x_B, y_B]^T$ are mapped to points $P_A = [x_A, y_A]^T$ according to:

$$P_A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} P_B + \begin{bmatrix} t_x \\ t_y \end{bmatrix}. \tag{3.3}$$

3.4.2 Homography transformation

A homography transformation deforms a given quadrilateral $\mathcal{Q}_B = \{Q_{B1}, \ldots, Q_{B4}\}$ into any other given quadrilateral $\mathcal{Q}_A = \{Q_{A1}, \ldots, Q_{A4}\}$, while keeping collinearity. It

has 8 degrees-of-freedom and is more flexible than the affine transformation, as it can handle perspective since parallel lines need not remain parallel. Homography is the model relating 2-D images (pinhole projections) of a 3-D plane. We adopt the 4-point homography parametrization from [DeTone et al., 2016], which consists of defining the quadrilateral $\mathcal{Q}_B$ of the target image to be the outer edge of the image, and using the coordinates of the quadrilateral $\mathcal{Q}_A$ of the source image as the 8-dimensional vector θ_{HOM}:

$$\theta_{\text{HOM}} = [x_{Q_{A1}}, \ldots, x_{Q_{A4}}, y_{Q_{A1}}, \ldots, y_{Q_{A4}}]. \tag{3.4}$$

This parametrization can be converted to the 3×3 homography matrix H [Hartley and Zisserman, 2003], which is used to perform the actual transformation:

$$\begin{aligned} x_A &= \frac{h_{11}\, x_B + h_{12}\, y_B + h_{13}}{h_{31}\, x_B + h_{32}\, y_B + h_{33}}, \\ y_A &= \frac{h_{21}\, x_B + h_{22}\, y_B + h_{23}}{h_{31}\, x_B + h_{32}\, y_B + h_{33}}, \end{aligned} \tag{3.5}$$

where, h_{ij} are elements of the homography matrix H.

3.4.3 Thin-plate spline transformation

The thin-plate spline (TPS) transformation [Bookstein, 1989] is a parametric model which performs smooth 2-D interpolation given a set of k corresponding control points $\mathcal{C}_A = \{C_{A1}, \ldots, C_{Ak}\}$ and $\mathcal{C}_\mathcal{B} = \{C_{B1}, \ldots, C_{Bk}\}$ between two images. In this chapter, we use $k = 9$ and arrange the control points $\mathcal{C}_B$ in a 3×3 uniform grid on the target image, as illustrated in Fig. 3-5. Because control points $\mathcal{C}_B$ are fixed for all image pairs, the TPS transformation is parametrized only by the control points $\mathcal{C}_A$ in the source image:

$$\theta_{\text{TPS}} = [x_{C_{A1}}, \ldots, x_{C_{A9}}, y_{C_{A1}}, \ldots, y_{C_{A9}}]. \tag{3.6}$$

Then, the thin-plate spline transformation maps points $P_B = [x_B, y_B]^T$ to points

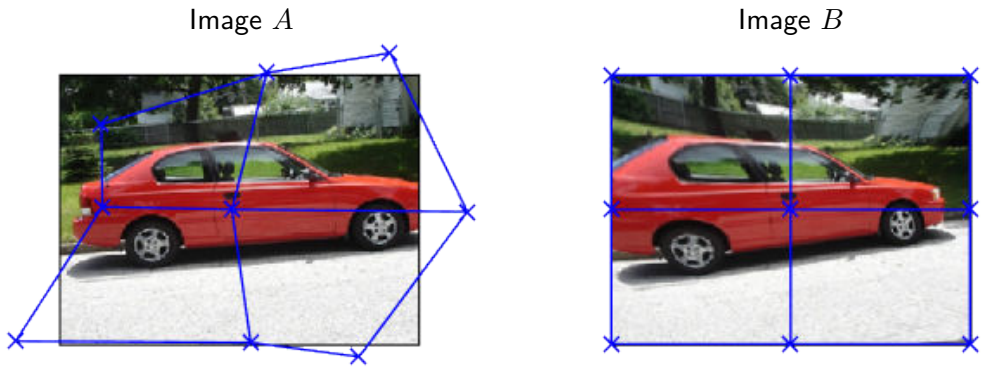


Figure 3-5: **Thin-plate spline control points.** Illustration of the 3×3 TPS grid of control points used in the thin-plate spline transformation model.

$P_A = [x_A, y_A]^T$ according to:

$$\begin{aligned} x_A &= a_x + b_x\, x_B + c_x\, y_B + \sum_{i=1}^{k} w_{xi} U(\|P_B - C_{Bi}\|), \\ y_A &= a_y + b_y\, x_B + c_y\, y_B + \sum_{i=1}^{k} w_{yi} U(\|P_B - C_{Bi}\|). \end{aligned} \tag{3.7}$$

Here, $U(z) = z^2 \log z^2$ and the parameters a, b, c and w are computed from θ_{TPS} by:

$$\begin{aligned} [w_{x1}, \ldots, w_{xk}, a_x, b_x, c_x]^T &= L^{-1}[x_{C_{A1}}, \ldots, x_{C_{Ak}}, 0, 0, 0]^T \\ [w_{y1}, \ldots, w_{yk}, a_y, b_y, c_y]^T &= L^{-1}[y_{C_{A1}}, \ldots, y_{C_{Ak}}, 0, 0, 0]^T, \end{aligned} \tag{3.8}$$

where L^{-1} is a constant matrix which needs to be computed only once, as it depends only on the fixed control points $\mathcal{C}_B$. Please refer to [Bookstein, 1989] for further details.

3.4.4 Hierarchy of transformations

A commonly used approach when estimating image to image transformations is to start by estimating a simple transformation and then progressively increase the model

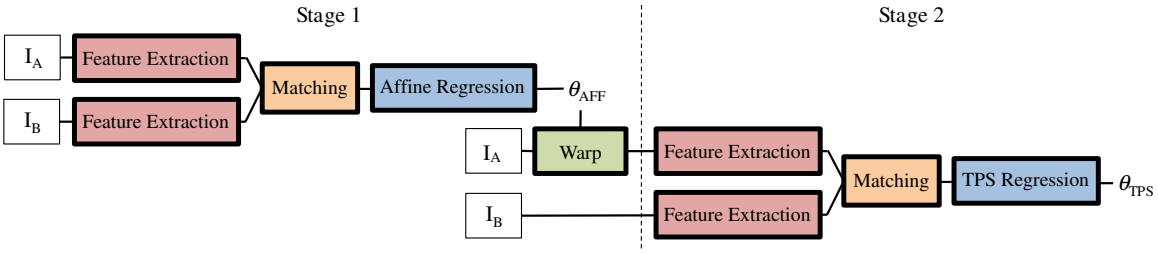


Figure 3-6: **Estimating progressively more complex geometric transformations.** Images A and B are passed through a network which estimates an affine transformation with parameters θ_{AFF} (see Fig. 3-2). Image A is then warped using this transformation to roughly align it with B, and passed along with B through a second network which estimates a thin-plate spline (TPS) transformation that refines the alignment.

complexity, refining the estimates along the way [Lowe, 2004; Berg et al., 2005; Philbin et al., 2007]. The motivation behind this method is that estimating a very complex transformation could be hard and computationally inefficient in the presence of clutter, so a robust and fast rough estimate of a simpler transformation can be used as a starting point, also regularizing the subsequent estimation of the more complex transformation.

We follow the same good practice and start by estimating an affine transformation (or alternatively a homography) which performs a rough alignment. The estimated affine transformation is then used to align image A to image B using an image resampling layer [Jaderberg et al., 2015]. The aligned images are then passed through a second geometry estimation network which estimates the parameters of a thin-plate spline transformation. The final estimate of the geometric transformation is then obtained by composing the two transformations. The process is illustrated in Fig. 3-6, and detailed in Algorithm 2.

3.4.5 Iterative refinement

When input images are related by a large transformation, it is difficult to obtain many good matches, so a single pass through the geometry estimation network might produce a poor alignment. In such cases, performing several iterations of the estimation can be beneficial, as illustrated in Fig. 3-7, since it allows the number of

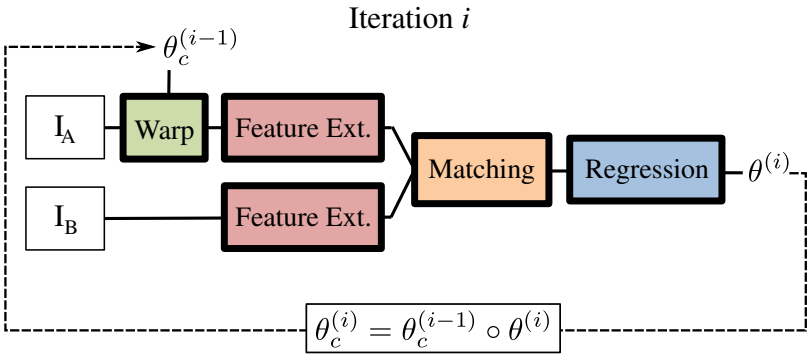


Figure 3-7: **Iterative transformation refinement.** In iteration i, image A is warped using the cumulative transformation estimate $\theta_c^{(i-1)}$ obtained from the previous iteration ($\theta_c^{(0)}$ is initialized to identity). A fine alignment, $\theta^{(i)}$, between image B and the warped image A is estimated and chained onto $\theta_c^{(i-1)}$ to form the refined cumulative transformation estimate $\theta_c^{(i)}$.

matches to progressively grow. This approach has proven to be particularly useful for instance-level alignment, as detailed in section 3.6.4.

3.5 Training

In order to train the parameters of our geometric matching CNN, it is necessary to design the appropriate loss function, and to use suitable training data. We address these two important points next, and also provide details about the implementation.

3.5.1 Loss function

We assume a fully supervised setting, where the training data consists of pairs of images and the desired outputs in the form of the parameters θ_{GT} of the ground-truth geometric transformation. The loss function $\mathcal{L}$ is designed to compare the estimated transformation θ with the ground-truth transformation θ_{GT} and, more importantly, compute the gradient of the loss function with respect to the estimates $\frac{\partial \mathcal{L}}{\partial \theta}$. This gradient is then used in a standard manner to learn the network parameters which minimize the loss function by using backpropagation and Stochastic Gradient Descent.

It is desired for the loss to be general and not specific to a particular type of geometric model, so that it can be used for estimating affine, homography, thin-

plate spline or any other geometric transformation. Furthermore, the loss should be independent of the parametrization of the transformation and thus should not directly operate on the parameter values themselves. We address all these design constraints by measuring loss on an imaginary grid of points $\mathcal{G} = \{G_i\} = \{(x_i, y_i)\}_{i=1...N}$ which is being deformed by the transformation. Namely, we construct a grid of points in image space, transform it using the neural network estimated and ground-truth transformations $\mathcal{T}_\theta$ and $\mathcal{T}_{\theta_{GT}}$ with parameters θ and θ_{GT}, respectively, and measure the discrepancy between the two transformed grids by summing the squared distances between the corresponding grid points:

$$\mathcal{L}(\theta, \theta_{GT}) = \frac{1}{N} \sum_{i=1}^{N} \|G'_i - G''_i\|^2 \tag{3.9}$$

where $G'_i = \mathcal{T}_\theta(G_i)$ and $G''_i = \mathcal{T}_{\theta_{GT}}(G_i)$ are the transformed grid points according to the estimated and ground-truth transformations respectively. The grid points are uniformly distributed in the image using normalized coordinates, i.e. $x_i, y_i \in [-1, 1]$. Note that we construct the coordinate system such that the center of the image is at $(0, 0)$ and that the width and height of the image are equal to 2, i.e. the bottom left and top right corners have coordinates $(-1, -1)$ and $(1, 1)$, respectively.

The gradient of the loss function with respect to the transformation parameters, needed to perform backpropagation in order to learn network weights, can be computed easily if the location of the transformed grid points $G'_i = \mathcal{T}_\theta(G_i)$ is differentiable with respect to θ. This is commonly the case, for example, when $\mathcal{T}$ is an affine transformation, $\mathcal{T}_\theta(G_i)$ is linear in parameters θ and therefore the loss can be differentiated in a straightforward manner.

3.5.2 Training data

Our training procedure requires fully supervised training data consisting of image pairs and a known geometric relation. Training CNNs usually requires a lot of data, and no public datasets exist that contain many image pairs annotated with their geometric transformation. Therefore, we opt for training from synthetically generated

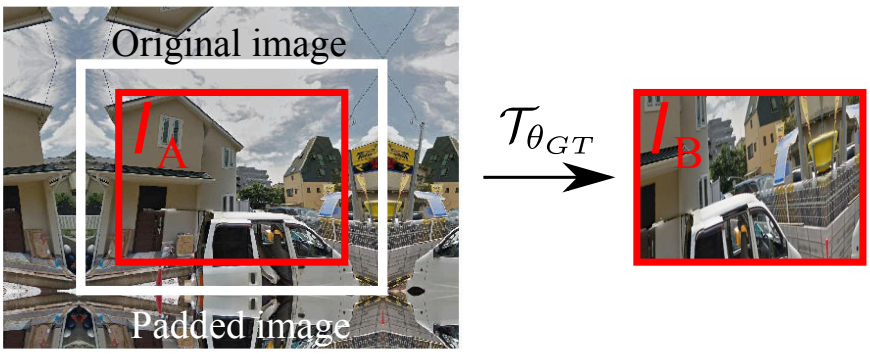


Figure 3-8: **Synthetic image generation.** Symmetric padding is added to the original image to enlarge the sampling region, its central crop is used as image A, and image B is created by performing a randomly sampled transformation $\mathcal{T}_{\theta_{GT}}$.

data, which gives us the flexibility to gather as many training examples as needed, for any 2-D geometric transformation of interest. Given that this training data is obtained for free, the approach can be classified as unsupervised (or self-supervised), even though the loss function requires the ground-truth transformation parameters.

Synthetic pair generation. We generate each training pair (I_A, I_B), by sampling I_A from a public image dataset, and generating I_B by applying a random transformation $\mathcal{T}_{\theta_{GT}}$ to I_A. More precisely, I_A is created from the central crop of the original image, while I_B is created by transforming the original image with added symmetrical padding in order to avoid border artifacts; the procedure is shown in Fig. 3-8.

Synthetic training datasets. The images used for the synthetic pair generation are sampled from the Tokyo Time Machine dataset [Arandjelović et al., 2016] which contains Google StreetView images of Tokyo. We select 20k images for training and 20k for validation.

During training, a batch is first selected from the training split of the dataset, and then a random transformation for each image in the batch is sampled independently from reasonable ranges. These ranges depend on the geometric model chosen for training the network.

In the case of the affine transformation, we chose a rotation angle $\theta \sim \mathcal{U}(-\pi/12, \pi/12)$, a shear angle $\phi \sim \mathcal{U}(-\pi/6, \pi/6)$, anisotropic scaling factors $\lambda_1, \lambda_2 \sim \mathcal{U}(0.75, 1.25)$,

and translations $t_x, t_y \sim \mathcal{U}(-0.25, 0.25)$. These parameters are defined on the SVD decomposition of the affine transformation (see [Hartley and Zisserman, 2003] sec. 2.4.3), and must then be composed to obtain the $[a_{ij}]$ matrix described in section 3.4.1:

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = R(\theta)R(-\phi)\text{diag}(\lambda_1, \lambda_2)R(\phi). \tag{3.10}$$

In the case of the homography and thin-plate spline transformations, the target points $\mathcal{Q}_A$ and $\mathcal{C}_A$ are obtained by perturbing the fixed $\mathcal{Q}_B$ and $\mathcal{C}_B$ with random translations $\delta_x, \delta_y \sim \mathcal{U}(-0.4, 0.4)$:

$$\begin{aligned} Q_{Ai} &= Q_{Bi} + (\delta_x, \delta_y), \\ C_{Ai} &= C_{Bi} + (\delta_x, \delta_y). \end{aligned} \tag{3.11}$$

In all cases, the uniform distribution was used in order not to impose a strong prior on the transformation parameters. The ranges were chosen to roughly cover the observed transformations in the PF-WILLOW dataset.

3.5.3 Implementation details

We use the PyTorch library [Paszke et al., 2017] and train the networks using the Adam [Kingma and Ba, 2015] optimizer with learning rate 10^{-3}, and a batch size of 16. There is no need for jittering as instead of data augmentation we can simply generate more synthetic training data. Input images are resized to 240×240 producing 15×15 feature maps that are passed into the matching layer. Single-stage models specific for each particular geometric transformation (affine, homography or thin-plate spline) are trained, with transformations sampled randomly at training time according to the previously described procedure. Each network is trained until convergence which typically occurs after 20 epochs (25000 iterations), and takes between 4 and 8 hours on a single GPU, depending on the complexity of the geometric model. The training

Algorithm 1: Training procedure using synthetic pairs

input : Image database DB
CNN model M_W
output : Trained CNN model M_W

for *training epochs* **do**
  for I *in* DB **do**
    `// Generate synthetic training pair`
    Sample random transformation θ_{GT}
    I_A = central crop of I;
    $I_B = \mathcal{T}_{\theta_{\text{GT}}}(I)$;
    `// Estimate transformation` θ
    $\theta = M_W(I_A, I_B)$;
    `// Compute loss and update model`
    $L = \mathcal{L}(\theta, \theta_{GT})$;
    $W = \text{update}(W, \frac{\partial \mathcal{L}}{\partial W})$;
  end
end

algorithm is detailed in Algorithm 1.

At evaluation time, the single-stage models featuring different geometric transformations can be used in conjunction as illustrated in Fig. 3-6. Trained networks can also be executed iteratively as described in section 3.4.5 and illustrated in Fig. 3-7. The evaluation algorithm is detailed in Algorithm 2.

3.6 Experimental results

In this section we compare our method to baselines and the state-of-the-art for both category-level and instance-level alignment problems.

In the case of category-level alignment, both qualitative and quantitative evaluation is performed on three different datasets previously used for this task: the PF dataset [Ham et al., 2017], the TSS dataset [Taniai et al., 2016] and the Caltech-101 dataset [Fei-Fei et al., 2006].

In the case of instance-level alignment, qualitative and quantitative evaluation is performed on the Graffiti benchmark [Mikolajczyk and Schmid, 2002]. In addition, qualitative alignment results are presented for the Tokyo Time Machine dataset [Arand-

Algorithm 2: Transformation estimation using two-stage network

input : Source and target images (I_A, I_B)
Stage 1 CNN model M_1
Stage 2 CNN model M_2
output : Aligned image I''_A

`// First stage`
$\theta_1 = M_1(I_A, I_B)$;
$I'_A = \mathcal{T}_{\theta_1}(I_A)$;
`// Second stage`
$\theta_2 = M_2(I'_A, I_B)$;
$I''_A = \mathcal{T}_{\theta_2}(I'_A)$;

jelović et al., 2016].

A different single-stage model for each of the affine, homography and thin-plate spline transformations was trained independently. Both single-stage (Fig. 3-2) and two-stage (Fig. 3-6) alignment strategies were investigated. Furthermore, the iterative refinement procedure described in section 3-7 was used for the Graffiti benchmark.

3.6.1 PF dataset

This dataset contains image pairs depicting different instances of the same classes, such as ducks and cars, but with large intra-class variations, e.g. the cars are often of different make, or the ducks can be of different subspecies. Furthermore, the images contain significant background clutter, as can be seen in Fig. 3-9. It contains a total of 2251 image pairs from two subgroups: PF-WILLOW (900 pairs, introduced in [Ham et al., 2016]) and PF-PASCAL (1251 pairs, introduced in [Ham et al., 2017]). Images from each pair were manually selected to ensure that objects have similar poses.

Evaluation metric. The quality of the obtained alignment is assessed by exploiting the keypoint annotation provided with the PF dataset. The task is to predict the locations of predefined keypoints from image A in image B. We do so by estimating a geometric transformation $\mathcal{T}$ that warps image A into image B, and applying the same transformation to the keypoint locations $\mathbf{P}_A = \{P_A^i\}_{i=1,\dots,n}$ in I_A, to obtain the estimated keypoint locations $\{\mathcal{T}(P_A^i)\}_{i=1,\dots,n}$ in I_B. The alignment quality is

then computed using the standard evaluation metric for this benchmark, the average probability of correct keypoint (PCK) [Yang and Ramanan, 2013], being the proportion of keypoints that are correctly matched. A keypoint is considered to be matched correctly if the distance between its predicted location $\mathcal{T}(P_A^i)$ and its ground-truth position P_B^i is below a predefined threshold L. Therefore, the PCK is computed as follows:

$$\text{PCK} = \frac{|\{P_A^i \in \mathbf{P}_A, d(\mathcal{T}(P_A^i), P_B^i) < L\}|}{n}, \tag{3.12}$$

where the distance threshold is $L = \alpha \cdot \max(h, w)$, $\alpha = 0.1$ and (h, w) are the height and width of the object bounding box, respectively.

Results. We compare our method against SIFT Flow [Liu et al., 2011], Graph-matching kernels (GMK) [Duchenne et al., 2011], Deformable spatial pyramid matching (DSP) [Kim et al., 2013], DeepFlow [Revaud et al., 2015], and all three variants of Proposal Flow (NAM, PHM, LOM) [Ham et al., 2017]. As shown in Tab. 3.1, our method outperforms all others and sets the new state-of-the-art on this data. The best competing methods are based on Proposal Flow and make use of object proposals, which enables them to guide the matching towards regions of images that contain objects. Their performance varies significantly with the choice of the object proposal method, illustrating the importance of this guided matching. On the contrary, our method does not use any guiding, but it still manages to outperform even the best Proposal Flow and object proposal combination.

Furthermore, we also compare against affine transformations estimated with RANSAC using the same descriptors as our method (VGG-16 `pool4`). The parameters of this baseline have been tuned extensively to obtain the best result by adjusting the thresholds for the second nearest-neighbour test and by pruning proposal transformations which are outside of the range of likely transformations. Our affine estimator outperforms the RANSAC baseline on this task by a 2% margin.

Fig. 3-9 illustrates the effectiveness of our method in category-level matching, where challenging pairs of images from the Proposal Flow dataset [Ham et al., 2017],

Methods	PF-PASCAL	PF-WILLOW
DeepFlow [Revaud et al., 2015]	0.21	0.20
GMK [Duchenne et al., 2011]	0.27	0.27
SIFT Flow [Liu et al., 2011]	0.33	0.38
DSP [Kim et al., 2013]	0.30	0.37
Proposal Flow (SS+NAM) [Ham et al., 2017]	0.36	0.52
Proposal Flow (SS+PHM) [Ham et al., 2017]	0.42	0.55
Proposal Flow (SS+LOM) [Ham et al., 2017]	0.45	0.56
RANSAC with our features (affine)	0.44	0.46
Ours (affine)	0.46	0.48
Ours (homography)	0.48	0.49
Ours (TPS)	0.51	0.54
Ours (affine + TPS)	0.51	**0.60**
Ours (homography + TPS)	**0.53**	**0.60**
Ours (2×TPS)	0.52	0.57

Table 3.1: **Results on the PF dataset.** We report the matching accuracy in terms of the PCK ($\alpha = 0.1$), for both PF-PASCAL and PF-WILLOW. All the numbers apart from ours and RANSAC are taken from [Ham et al., 2017].

containing large intra-class variations, are aligned correctly. The method is able to robustly, in the presence of clutter, estimate large translations, rotations, scale changes, as well as non-rigid transformations and some perspective changes.

3.6.2 TSS dataset

The TSS dataset introduced in [Taniai et al., 2016] contains 400 image pairs of three subgroups: FG3DCar contains 195 image pairs of cars, JODS contains 81 image pairs of airplanes, horses and cars and PASCAL contains 124 image pairs of bicycles, motorbikes, buses, cars and trains. For all 400 image pairs, approximate ground-truth optical flow data is provided.

Evaluation metric. The evaluation metric used for the TSS dataset is also the PCK, presented in Eq. (3.12), previously presented for the PF dataset. However, the ground-truth optical flow data available in the TSS dataset allows to compute the PCK using a dense set of keypoints $\mathbf{P}_A = \{P_A^i \in \mathbf{M}_A\}_{i=1,\dots,n}$, composed by the foreground pixels inside the segmentation mask $\mathbf{M}_A$ of the object in I_A. Therefore,

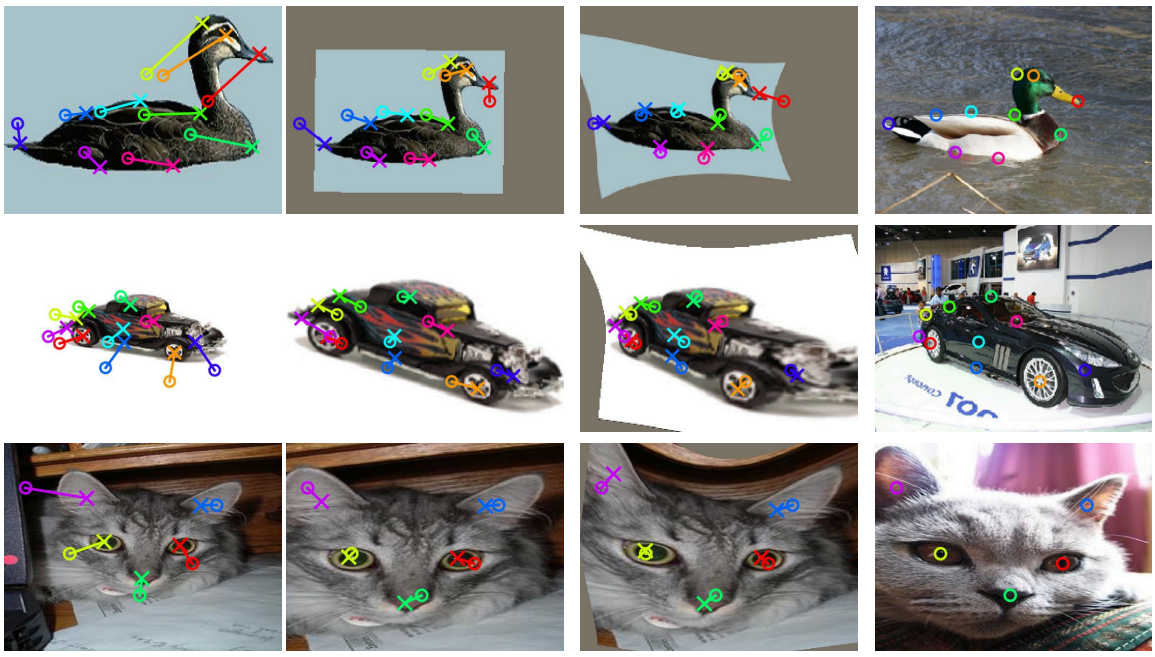


Image A | Aligned A (affine) | Aligned A (aff.+TPS) | Image B

Figure 3-9: **Qualitative results on the PF dataset.** Each row shows one test example from the Proposal Flow dataset. Ground truth matching keypoints, only used for alignment evaluation, are depicted as crosses and circles for images A and B, respectively. Keypoints of same color are supposed to match each other after image A is aligned to image B. To illustrate the matching error, we also overlay keypoints of B onto different alignments of A so that lines that connect matching keypoints indicate the keypoint position error vector. Our method manages to roughly align the images with an affine transformation (column 2), and then perform finer alignment using thin-plate spline (TPS, column 3). The top two examples are from the PF-WILLOW dataset while the bottom one is from PF-PASCAL.

this allows for the alignment to be densely evaluated over the object of interest, in contrast to the PCK computation for the PF dataset, where the alignment is only evaluated in a handful of manually annotated keypoints.

Regarding the distance threshold $L = \alpha \cdot \max(h, w)$ used for the PCK computation, the criterion used in [Ham et al., 2017; Taniai et al., 2016] is adopted, where the reported values are computed with $\alpha = 0.05$ and with (h, w) being the dimensions of the target image.

Results. The quantitative results for the TSS dataset are presented in Tab. 3.2, in terms of the mean PCK over the set of image pairs. For each of the 400 pairs, both

Methods	FG3DCar	JODS	PASCAL	All
DSP [Kim et al., 2013]	0.49	0.47	0.38	0.45
SIFT Flow [Liu et al., 2011]	0.63	0.51	0.36	0.52
Taniai *et al.* [Taniai et al., 2016]	0.83	0.60	0.48	0.67
Proposal Flow (SS+LOM) [Ham et al., 2017]	0.79	0.65	0.53	0.68
Ours (affine)	0.81	0.65	0.51	0.68
Ours (homography)	0.83	0.66	0.52	0.70
Ours (TPS)	0.84	**0.72**	0.51	0.71
Ours (affine + TPS)	**0.89**	**0.72**	0.54	**0.75**
Ours (homography + TPS)	0.88	**0.72**	**0.55**	**0.75**
Ours (2×TPS)	0.86	0.70	0.52	0.72

Table 3.2: **Results on the TSS dataset.** We report the matching accuracy in terms of PCK ($\alpha = 0.05$). The three intermediate columns show the results for each subset of the TSS dataset: FG3DCar, JODS and PASCAL. The last column shows the PCK result over the whole dataset.

the forward (from I_A to I_B) and backward (from I_B to I_A) alignments are computed and evaluated, resulting in a total of 800 evaluation pairs.

The middle columns of Tab. 3.2 present the mean PCK over the three subsets of the TSS dataset: FG3DCar, JODS and PASCAL; and the right-most column presents the mean PCK over the whole TSS dataset. It can be observed that our single-stage models improve the overall average score by up to 3%, while the two-stage models achieve the best results on all the different subsets, and improve by up to 7% over the previously published results.

In Fig. 3-10 we present qualitative results on the TSS dataset. In order to assess the visual quality of the obtained results, we also present the ground-truth aligned and segmented images provided with the dataset in the right-most column.

As it can be observed, the proposed method can produce good alignments results, which are close to the ground-truth alignment.

3.6.3 Caltech-101 dataset

Following the same procedure as in [Kim et al., 2013; Ham et al., 2017], the alignment quality is also evaluated on the Caltech-101 dataset [Fei-Fei et al., 2006]. For each of

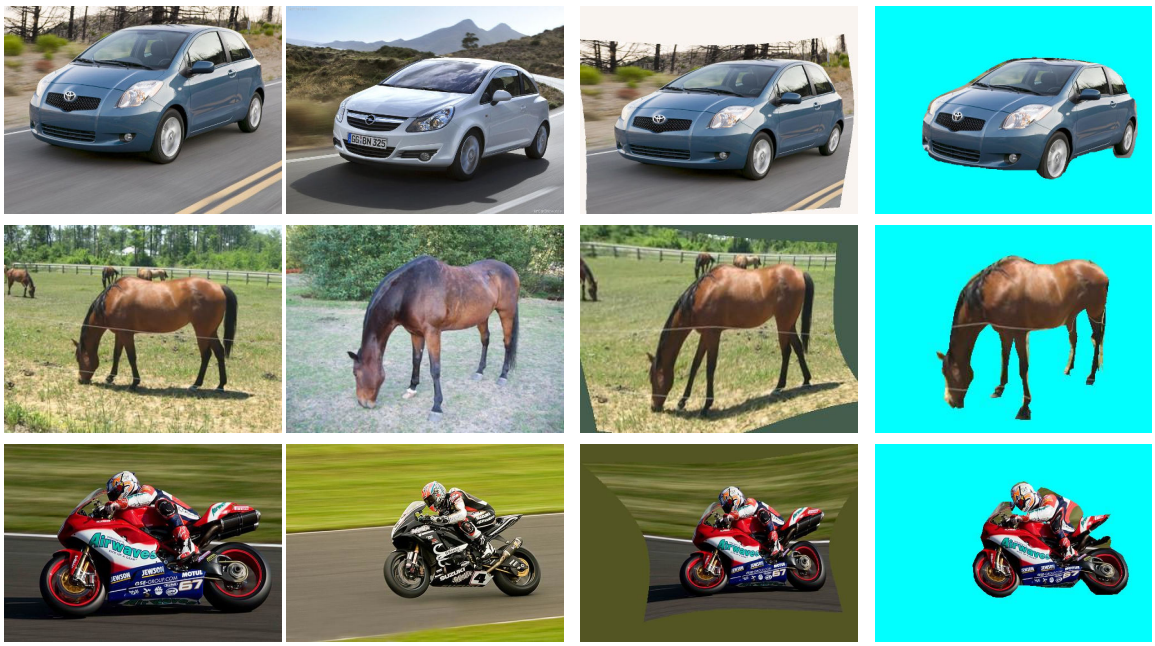


Image A | Image B | Aligned A (aff.+TPS) | Ground-truth

Figure 3-10: **Qualitative results on the TSS dataset.** Each row shows one test example from the TSS dataset. The last column shows the ground-truth alignment used for evaluation. Example 1 is from TSS-FG3DCar, examples 2-3 are from TSS-JODS, and 4-7 from TSS-PASCAL.

the 101 categories, 15 image pairs were chosen randomly, resulting in 1515 evaluation pairs. These pairs are the same as in [Ham et al., 2017].

Evaluation metrics. As no keypoint annotations are provided for the Caltech-101 dataset, PCK cannot be used to assess the matching accuracy. Since segmentations masks are provided, we follow [Kim et al., 2013; Ham et al., 2017] and evaluate the quality of segmentation mask alignment using the following metrics: label transfer accuracy (LT-ACC), intersection-over-union (IoU), and localization error (LOC-ERR).

Let (I_A, I_B) be a pair of images with ground-truth segmentation masks $(\mathrm{M}_A, \mathrm{M}_B)$, and $\mathcal{T}$ be the estimated transformation from I_A to I_B. Then, the transferred annotated mask from the source image, $\mathrm{M}'_A = \mathcal{T}(\mathrm{M}_A)$, is compared with the ground-truth mask in the target image M_B to assess the alignment quality.

The label transfer accuracy metric (LT-ACC), measures the number of correctly transferred foreground and background pixels in the following way:

$$\text{LT-ACC} = \frac{|\{P \in I_B, \mathrm{M}'_A(P) = \mathrm{M}_B(P)\}|}{|P \in I_B|} \tag{3.13}$$

Therefore, the numerator of the LT-ACC adds up the number of background pixels which are correctly mapped to background and the foreground pixels which are correctly mapped to foreground.

The intersection-over-union (IoU), or Jaccard index, as the LT-ACC also compares the mask alignment quality. However, contrary to LT-ACC, it only considers the correctly aligned foreground pixels, ignoring the background. It is computed in the following way:

$$\text{IoU} = \frac{|\mathrm{M}'_A \cap \mathrm{M}_B|}{|\mathrm{M}'_A \cup \mathrm{M}_B|}. \tag{3.14}$$

Finally, the localization error (LOC-ERR) metric measures the spatial error of each transferred pixel [Kim et al., 2013], assuming that the images are related by a translation and anisotropic scaling transformation which aligns the bounding boxes of the source and target images.

To this end, two normalized coordinate systems are defined relative to the source and target image bounding boxes, such that their origins are set on the top-left corners of the object bounding boxes, and the coordinates are normalized by the widths and heights of the bounding boxes.

Let $O_A = (x_{O_A}, y_{O_A})$ and $O_B = (x_{O_B}, y_{O_B})$ be the top-left corners of the objects bounding boxes on I_A and I_B respectively, and (h_A, w_A), (h_B, w_B) the bounding box dimensions.

Then LOC-ERR metric measures the disagreement between the coordinates of the original point $P_A = (x_A, y_A)$ relative to O_A, and its transformed coordinates $P'_A = \mathcal{T}(P_A) = (x'_A, y'_A)$ relative to O_B, in the following way:

$$\text{LOC-ERR} = \frac{1}{|V|} \sum_{(P_A, P'_A) \in V} |\hat{x}_A - \hat{x}'_A| + |\hat{y}_A - \hat{y}'_A|, \tag{3.15}$$

where $(\hat{x}_A, \hat{y}_A)$ and $(\hat{x}'_A, \hat{y}'_A)$ are the normalized coordinates of P_A and $P'_A = \mathcal{T}(P_A)$:

Methods	LT-ACC	IoU	LOC-ERR
DeepFlow [Revaud et al., 2015]	0.74	0.40	0.34
GMK [Duchenne et al., 2011]	0.77	0.42	0.34
SIFT Flow [Liu et al., 2011]	0.75	0.48	0.32
DSP [Kim et al., 2013]	0.77	0.47	0.35
Proposal Flow (RP, LOM) [Ham et al., 2016]	0.78	0.50	0.26
Proposal Flow (SS, LOM) [Ham et al., 2016]	0.78	0.50	0.25
Ours (affine)	0.78	0.51	**0.24**
Ours (homography)	0.80	0.52	**0.24**
Ours (TPS)	0.80	0.53	**0.24**
Ours (affine + TPS)	0.79	**0.55**	0.26
Ours (homography + TPS)	**0.81**	**0.55**	0.25
Ours (2×TPS)	0.80	0.54	0.26

Table 3.3: **Evaluation on the Caltech-101 dataset.** Matching quality is measured in terms of LT-ACC and IoU. The best two Proposal Flow methods (RP, LOM and SS, LOM) are included here. All numbers apart from ours are taken from [Ham et al., 2016].

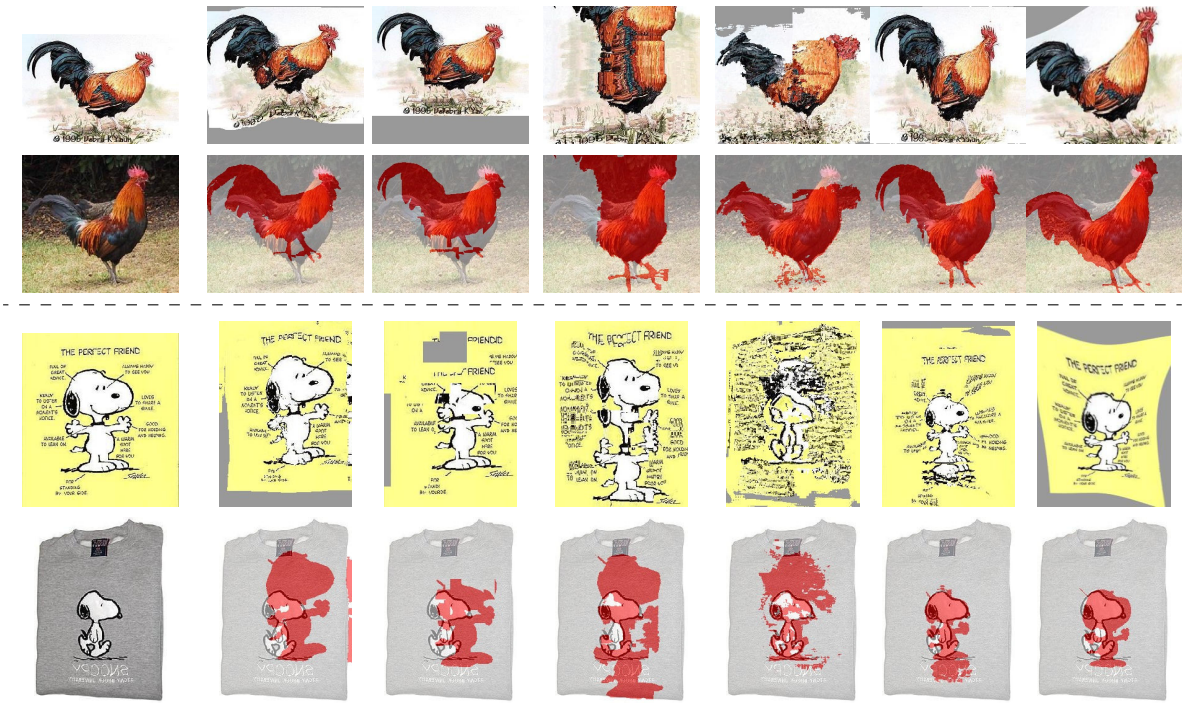
$$
\begin{aligned}
(\hat{x}_A, \hat{y}_A) &= \left(\frac{x_A - x_{O_A}}{w_A}, \frac{y_A - y_{O_A}}{h_A} \right) \\
(\hat{x}'_A, \hat{y}'_A) &= \left(\frac{x'_A - x_{O_B}}{w_B}, \frac{y'_A - y_{O_B}}{h_B} \right)
\end{aligned}
\tag{3.16}
$$

and V is the set of all pairs of points (P_A, P'_A) in which the transformed points $P'_A = \mathcal{T}(P_A)$ fall inside the bounds of image I_B.

Results. The quantitative results on the Caltech-101 dataset are presented in Tab. 3.3. As it can be observed, our approach outperforms the state-of-the-art by a significant margin, obtaining, for example, an IoU of 0.55 compared to the previous best result of 0.50.

In addition, it can be observed that the LOC-ERR metric values do not follow the trend of the other two metrics. This is because the LOC-ERR metric makes the invalid assumption that the images are related with a translation and anisotropic scaling transformation.

The benefit of the two-stage approaches is clear from the more realistic IoU metric, where adding a second stage achieves an IoU of 0.55 compared to 0.53 of the single-stage



(a) Image pairs (b) DeepFlow (c) GMK (d) SIFT Flow (e) DSP (f) PF (g) Ours

Figure 3-11: **Qualitative results on the Caltech-101 dataset.** Each block of two rows corresponds to one example, where column (a) shows the original images – image A in the first row and image B in the second row. The remaining columns of the first row show image A aligned to image B using various methods. The second row shows image B overlaid with the segmentation map transferred from image A. Our results correspond to the two-stage affine+TPS model.

best performing model.

In Fig. 3-11, we present a qualitative comparison of the results obtained by our method and other previous methods on images from this dataset. For each example, the second row presents the transferred segmentation mask $\mathcal{T}(\mathrm{M}_A)$ of I_A overlaid with I_B, for each of the methods. As it can be visually assessed, the proposed approach achieves a superior alignment than most of the previous methods.

3.6.4 Graffiti benchmark

This section presents the results of the proposed method on the challenging Graffiti instance-level matching benchmark [Mikolajczyk and Schmid, 2002]. This benchmark contains 6 images of the same planar scene with increasingly varying viewpoint, with

up to 70° azimuthal rotation from the reference image. Ground-truth homography transformations from the reference image 1 to images 2-6 are available with the dataset. We employed the same homography estimation CNN used for the category-level alignment datasets, trained from synthetic StreetView image pairs. To overcome the large viewpoint variation in this dataset, we perform iterative refinement, as described in section 3.4.5, with a total of 5 iterations. The same CNN model is used for all iterations.

Evaluation metric. In order to measure the quality of the estimated homography transformation, the average endpoint error is used:

$$\text{AEE}(H, H_{GT}) = \frac{1}{N} \sum_{i=1}^{N} \|G'_i - G''_i\| \tag{3.17}$$

where $G'_i = \mathcal{T}_H(G_i)$ and $G''_i = \mathcal{T}_{H_{GT}}(G_i)$ are the transformed sampling points $G = [1, 2, \dots, h] \times [1, 2, \dots, w]$ when applying the estimated and ground-truth homographies, respectively. Note that this is similar to the proposed loss (5.11), but without squaring the distances.

Results. Quantitative results are presented on Tab. 3.4. The proposed method is compared against state-of-the-art methods for this dataset, such as SIFT features (DoG+SIFT) [Lowe, 2004], SIFT features with affine-covariant detectors (DoG-affine+SIFT) [Mikolajczyk and Schmid, 2002] and ASIFT [Yu and Morel, 2011]. Features are matched and filtered using the second nearest-neighbour test [Lowe, 2004] and the homography transformation is estimated with the locally optimized RANSAC algorithm [Lebeda et al., 2012] with the following parameter settings: distance threshold $\theta = 6$px, estimation confidence $\eta_0 = 0.999$, and the final refinement step using all inliers. The rest of the parameters are set to their default values. The RANSAC algorithm is executed 5 times and the mean error values are reported. In all cases, the standard deviations were below 0.3 pixels.

As it can be observed from Tab. 3.4, although our method does not produce the best results, it still achieves reasonable alignments for pairs $(1, 2)$ up to $(1, 5)$, and

Methods	Image pair				
	$(1,2)$	$(1,3)$	$(1,4)$	$(1,5)$	$(1,6)$
DoG+SIFT [Mikolajczyk and Schmid, 2002]	0.63	1.61	2.89	fail	fail
DoG-affine+SIFT [Mikolajczyk and Schmid, 2002]	0.60	1.57	1.44	2.40	2.75
ASIFT [Yu and Morel, 2011]	**0.45**	**1.35**	**1.02**	**0.96**	**1.62**
Ours (5×homography)	4.17	4.81	3.17	2.55	fail

Table 3.4: **Evaluation on the Graffiti benchmark.** Matching quality is measured in terms of the AEE (px), being the original image of 640×800 px. Our homography estimation model is run recursively five times.

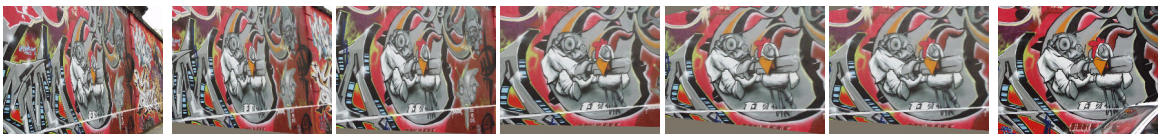
fails for pair $(1,6)$. The lower performance of our method when compared to local interest points methods can be explained by the lower resolution of the input image, which is resized from 800×640px to 240×240px, and also by the low resolution of the extracted CNN features, which is of 15×15. In addition, due to the max-pooling operations, the exact positions of the image features cannot be recovered from the CNN features.

However, it is interesting to point out that while DoG-SIFT fails for pair $(1,5)$, our method does not. This confirms the intuition that CNN feature descriptors have some degree of affine invariance, achieved due to multiple pooling operations. The invariance enables multiple good initial matches to be established, producing a good initial transformation estimate, which is then progressively refined using the iterative procedure.

Qualitative results for the pair $(1,5)$ of the Graffiti benchmark are shown in Fig. 3-12. The figure also shows the progression of the alignment as more refinement iterations are performed. As it can be observed, the obtained alignment is qualitatively good.

3.6.5 Tokyo Time Machine dataset

Qualitative results for the Tokyo Time Machine dataset [Arandjelović et al., 2016] are shown in Fig. 3-13. The images have been captured at different points in time which



Source image Iter. 1 Iter. 2 Iter. 3 Iter. 4 Iter. 5 Target image

Figure 3-12: **Qualitative results on the Graffiti benchmark, pair** $(1, 5)$**.** The first and last columns show the source and target images. The intermediate columns show the progress of the alignment at different iterations.

(a) Image A (b) Image B (c) Aligned im. A (d) Overlay b+c (e) Diff. map

Figure 3-13: **Qualitative results on the Tokyo Time Machine dataset.** Each row shows a pair of images from the Tokyo Time Machine dataset, and our alignment along with a "difference map", highlighting absolute differences between aligned images in the descriptor space. Our method successfully aligns image A to image B despite of viewpoint and scene changes (highlighted in the difference map).

are months or years apart. Note that, by automatically highlighting the differences (in the feature space) between the aligned images, it is possible to detect changes in the scene, such as occlusions, changes in vegetation, or structural differences e.g. new buildings being built.

3.7 Discussions and ablation studies

In this section we examine the importance of various components of our architecture, and discuss the impact of the training set, the learned filters and the limitations of the method.

Methods	PCK
Subtraction [Kanazawa et al., 2016]	0.24
Concatenation [DeTone et al., 2016]	0.34
Ours (without normalization)	0.41
Ours (trained on PASCAL)	0.47
Ours (trained on StreetView)	0.48

Table 3.5: **Ablation studies.** We report ablations on the matching layer and effect of the training dataset. Accuracy is measured in terms of the PCK on the PF-WILLOW dataset using the single-stage affine model.

3.7.1 Correlation versus concatenation and subtraction

Replacing our correlation-based matching layer with feature concatenation or subtraction, as proposed in [DeTone et al., 2016] and [Kanazawa et al., 2016], respectively, incurs a large performance drop, as shown in Tab. 3.5. The behavior is expected as we designed the matching layer to only keep information on pairwise descriptor similarities rather than the descriptors themselves, as is good practice in classical geometry estimation methods, while concatenation and subtraction do not follow this principle.

3.7.2 Normalization

Table 3.5 also shows the importance of the correlation map normalization step, where the normalization improves results from 41% to 48%. The step mimics the second nearest-neighbour test used in classical feature matching [Lowe, 2004], as discussed in Sec. 3.3.2. Note that [Fischer et al., 2015] also uses a correlation layer, but they do not normalize the map in any way, which is clearly suboptimal.

3.7.3 Generalization

In order to assess the influence on the performance of the trained method with respect to the training dataset used, we train a second model using PASCAL VOC 2011

[Everingham et al., 2011] images, instead of StreetView images. As seen in Tab. 3.5, our method is relatively unaffected by the choice of training data as its performance is similar regardless whether it was trained with StreetView or PASCAL images. We also attribute this to the design choice of operating on pairwise descriptor similarities rather than the raw descriptors.

3.7.4 Geometric models

Different configurations of the proposed method have been analyzed, varying the geometric models and using both single-stage (as in Fig. 3-2) and two-stage approaches (as in Fig. 3-6).

Results from Tables 3.1, 3.2, and 3.3 show that the two-stage homography+TPS is the best performing approach, being slightly superior to affine+TPS. On the other hand, TPS alone is the best single-stage approach, but interestingly, two-iteration TPS performs worse than both affine+TPS and homography+TPS. This confirms the intuition discussed in section 3.4.4, where using a simpler geometric model to perform the rough alignment is expected to be more robust than using a more complex one.

3.7.5 What is being learned?

We examine filters of size $7 \times 7 \times 225$ from the first convolutional layer of the regressor, which operate directly on the output of the matching layer, i.e. the tentative correspondence map. We observe that two filter properties emerge from training: (i) filters specialize in detecting matches in specific positions in image A, and (ii) filters learn to mimic local neighbourhood consensus for robust match estimation. In order to visualize this, each $1 \times 1 \times 225$ 1-D slice through the channels of one convolutional filter at a particular spatial location is reshaped as a 15×15 image. Recall that the 225 channels correspond to flattened similarities with image A (see Fig. 3-3 and Eq. (3.1)), therefore these images show the filter's preferences to matches in specific locations in image A. For visualization, we pick the peaks from all slices of filter weights and average them together to produce a single image. Several filters are shown

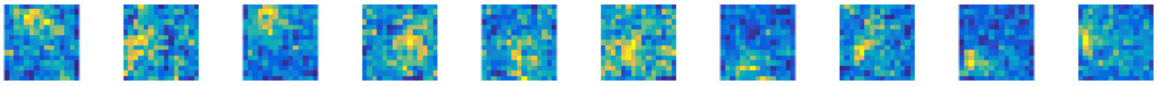


Figure 3-14: **Filter visualization.** Some convolutional filters from the first layer of the regressor, acting on the tentative correspondence map, show preferences to spatially co-located features that transform consistently to the other image, thus learning to perform the local neighbourhood consensus criterion often used in classical feature matching. Refer to the text for more details on the visualization.

in Fig. 3-14. It can be observed that matches form clusters, which means that spatially co-located features in image B (within the 7×7 support of the filter) respond strongly to spatially consistent locations in image A, therefore confirming that this layer has learned to mimic local neighbourhood consensus. Furthermore, it can be observed that the size of the preferred spatial neighbourhood varies across filters, thus showing that filters specialize for certain scale changes. Finally, the fact that the location of the highest filter weights (bright yellow) is different for different filters shows that the filters specialize for different locations in image A.

3.7.6 Limitations

Next, we analyze limitations of the proposed method and discuss possible ways of alleviating them.

Robustness to occlusion. The robustness of the proposed method to occlusion was assessed by computing the PCK on the PF-WILLOW dataset when substituting a rectangular portion of each image in the dataset with a crop from a different unrelated image. Both positions and aspect ratios of these rectangles were independently and randomly sampled for each image. The results are shown in Tab. 3.6. It can be observed that all methods, including Proposal Flow [Ham et al., 2016], degrade significantly when occluding 10% or 20% of the area of the images. Although retraining the proposed method replicating the occlusion procedure helps to improve the performance on the occluded data, it also degrades the performance on the unoccluded case. Therefore, alignment with significant occlusion still presents a challenge for the current methods, including ours.

Methods	Occluded area		
	0%	10%	20%
Proposal Flow (SS, LOM) [Ham et al., 2016]	0.56	0.46	0.30
Ours (homography+TPS)	**0.60**	0.46	0.32
Ours trained with 10% occlusions (hom.+TPS)	0.57	**0.47**	0.36
Ours trained with 20% occlusions (hom.+TPS)	0.48	0.43	**0.38**

Table 3.6: **Robustness to occlusions.** We report the PCK on the PF-WILLOW dataset.

Multiple objects. Currently, the proposed method can only produce a global alignment of the image pair, and handling multiple objects is still a challenge. This is in line with current datasets on category-level image alignment which contain a single foreground object, and with all competing methods which also make this assumption. This limitation could be addressed by incorporating an attention mechanism.

Learning better features. Although the proposed architecture is fully differentiable, which makes it end-to-end trainable for the task of semantic alignment, we have observed that finetuning the feature extraction CNN does not improve alignment performance. This is because our synthetic dataset used for training does not contain rich appearance variations present in the real category-level alignment datasets used for evaluation. While supervision from synthetic data comes with no cost and is useful to train the regression CNN, it is not suited for learning better image features for alignment. As a solution to this problem, we have developed a combined approach using synthetic data for training the regression CNN and real data for finetuning the feature extraction CNN [Rocco et al., 2018a].

Confidence in the estimated transformation. The proposed method does not currently produce a measure of the confidence in the estimated transformation. However, the soft-inlier count presented in [Rocco et al., 2018a] could be employed for this purpose.

Asymmetry in the method. Given an pair of images I_A and I_B, the method is trained to produce the alignment in one direction only. In order to make the method more symmetric, the alignments in both directions could be estimated simultaneously and a cycle consistency loss [Zhou et al., 2016b; Zhu et al., 2017] could be incorporated.

3.7.7 Computational cost

The presented method currently takes about 1.6s per 240×240px image pair, which is $1.5\times$ faster than SIFT Flow and $6\times$ faster than Proposal Flow, when run on a modern CPU. Furthermore, the presented method can also be run on the GPU, which allows to obtain an additional $40\times$ speedup.

3.8 Conclusions

In this chapter we have described a network architecture for geometric matching trainable from synthetic imagery without the need for manual annotations. The architecture is modular and flexible, and can be applied iteratively, in order to estimate large transformations, or in a cascade, enabling estimation of complex transformations. Thanks to our matching layer, the network generalizes well to never seen before imagery, reaching state-of-the-art results on several challenging datasets for category-level matching. The method has also proven useful for instance-level alignment, obtaining reasonable alignment for the challenging Graffiti benchmark. This opens-up the possibility of applying our architecture to other difficult correspondence problems such as matching across large changes in illumination (day/night) [Arandjelović et al., 2016] or depiction style [Aubry et al., 2013].

Chapter 4

End-to-end weakly-supervised semantic alignment

In this chapter, we adopt the correspondence estimation architecture from Chapter 3 but instead of training from synthetically transformed imagery (as done in Chapter 3), we propose a different training scheme using weak image-level supervision that allows training the model directly in an end-to-end manner from real image pairs. The outcome is that parameters are learnt from the rich appearance variation present in different but semantically related images without the need for tedious manual annotation of correspondences for training. To enable weakly-supervised training, we develop a differentiable *soft inlier scoring* module, inspired by the inlier scoring procedure from RANSAC [Fischler and Bolles, 1981]. Our soft inlier scoring module computes the quality of the alignment based only on geometrically consistent correspondences that are in agreement with a geometric model, thereby reducing the effect of background clutter. Our experiments demonstrate that the proposed approach can improve the performance over training solely using the self-supervised approach from Chapter 3, achieving state-of-the-art performance on multiple standard benchmarks for semantic alignment.

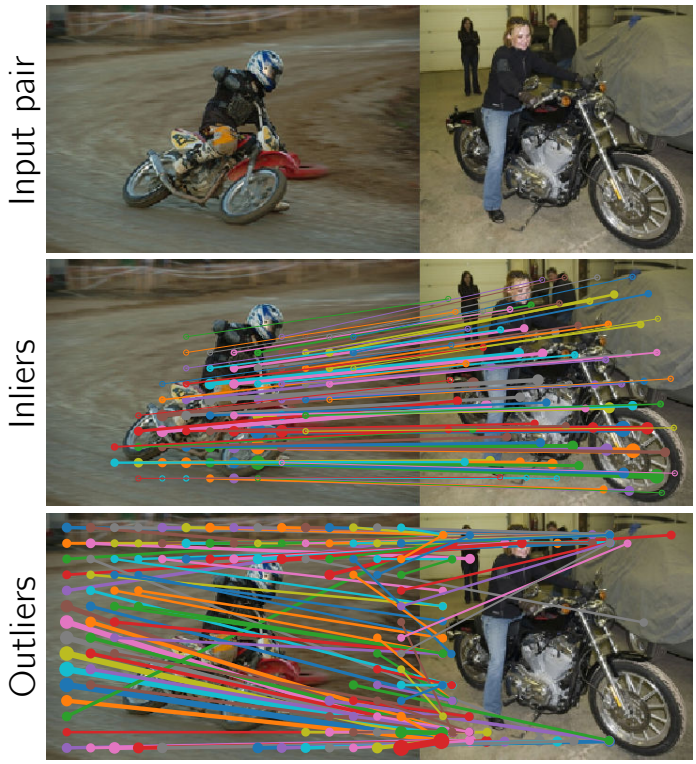


Figure 4-1: **Illustration of the proposed method.** We describe a CNN architecture that, given an input image pair (top), outputs dense semantic correspondence between the two images together with the aligning geometric transformation (middle) and discards geometrically inconsistent matches (bottom). The alignment model is learnt from weak supervision in the form of matching image pairs without correspondences.

4.1 Introduction

Finding correspondence is one of the fundamental problems in computer vision. Initial work has focused on finding correspondence between images depicting the same object or scene with applications in image stitching [Szeliski, 2006], multi-view 3D reconstruction [Hartley and Zisserman, 2003], motion estimation [Weinzaepfel et al., 2013; Fischer et al., 2015] or tracking [Newcombe et al., 2011; Engel et al., 2014]. In this chapter we study the problem of finding category-level correspondence, or *semantic alignment* [Berg et al., 2005; Liu et al., 2011], where the goal is to establish dense correspondence between different objects belonging to the same category, such as the two different motorcycles illustrated in Fig. 4-1. This is an important problem with applications in object recognition [Liu et al., 2011], image editing [Dale et al., 2017], or robotics [Nikandrova and Kyrki, 2015]. This is also an extremely challenging

task because of the large intra-class variation, changes in viewpoint and presence of background clutter.

The current best semantic alignment methods [Kim et al., 2017; Han et al., 2017; Novotny et al., 2017] employ powerful image representations based on convolutional neural networks coupled with a geometric deformation model. However, these methods suffer from one of the following two major limitations. First, the image representation and the geometric alignment model are not trained together in an end-to-end manner. Typically, the image representation is trained on some auxiliary task such as image classification and then employed in an often ad-hoc geometric alignment model. Second, while trainable geometric alignment models exist [Rocco et al., 2017; Brachmann et al., 2017], they require strong supervision in the form of ground truth correspondences, which is hard to obtain for a diverse set of real images on a large scale.

In this chapter, we address both these limitations and develop a semantic alignment model that is *trainable end-to-end* from *weakly supervised* data in the form of matching image pairs without the need for ground truth correspondences. To achieve that we design a novel convolutional neural network architecture for semantic alignment with a differentiable soft inlier scoring module inspired by the RANSAC inlier scoring procedure. The resulting architecture is end-to-end trainable with only image-level supervision. The outcome is that the image representation can be trained from rich appearance variations present in different but semantically related image pairs, rather than synthetically deformed imagery [Kanazawa et al., 2016; Rocco et al., 2017]. We show that our approach significantly improves the performance of the baseline deep CNN alignment model, achieving state-of-the-art performance on multiple standard benchmarks for semantic alignment. Our code and trained models are available at http://www.di.ens.fr/willow/research/weakalign/.

4.2 Related work

The problem of semantic alignment has received significant attention in the last few years with progress in both (i) image descriptors and (ii) geometric models.

Paper	Descriptor	Alignment method	Trainable D	A	E-E	S
Liu *et al.*'11 [Liu et al., 2011]	SIFT	SIFT Flow	✗	✗	✗	-
Kim *et al.*'13 [Kim et al., 2013]	SIFT+PCA	DSP	✗	✗	✗	-
Taniai *et al.*'16 [Taniai et al., 2016]	HOG	TSS	✗	✗	✗	-
Ham *et al.*'16 [Ham et al., 2017]	HOG	PF-LOM	✗	✗	✗	-
Yang *et al.*'17 [Yang et al., 2017]	HOG	OADSC	✗	✗	✗	-
Ufer *et al.*'17 [Ufer and Ommer, 2017]	AlexNet	DSFM	✗	✗	✗	-
Novotny *et al.*'17 [Novotny et al., 2017]	AnchorNet	DSP	✓	✗	✗	w
		PF-LOM	✓	✗	✗	w
Kim *et al.*'17 [Kim et al., 2018b]	FCSS	SIFT Flow	✓	✗	✗	s
		PF-LOM	✓	✗	✗	s
Kim *et al.*'17 [Kim et al., 2017]	FCSS	DCTM	✓	✗	✗	s
Han *et al.*'17 [Han et al., 2017]	VGG-16	SCNet-A	✓	✓	✗	s
		SCNet-AG	✓	✓	✗	s
		SCNet-AG+	✓	✓	✗	s
Rocco *et al.*'17 [Rocco et al., 2017]	VGG-16	CNN Geo.	✓	✓	✓	s
	ResNet-101	CNN Geo.	✓	✓	✓	s
Proposed method	ResNet-101	CNN Geo.	✓	✓	✓	w

Table 4.1: **Comparison of recent related work.** The table indicates employed image descriptor and alignment method. The last four columns show which components of the approach are trained for the semantic alignment task: descriptor (D), alignment (A) or both in end-to-end manner (E-E); and the level of supervision (S): strong (s) or weak (w).

The key innovation has been making the two components trainable from data. We summarize the recent progress in Table 4.1 where we indicate for each method whether the descriptor (D) or the alignment model (A) are trainable, whether the entire architecture is trainable end-to-end (E-E), and whether the required supervision is strong (s) or weak (w).

Early methods, such as [Berg et al., 2005; Liu et al., 2011; Kim et al., 2013], employed hand-engineered descriptors like SIFT or HOG together with hand-engineered alignment models based on minimizing a given matching energy. This approach has been quite successful [Ham et al., 2017; Taniai et al., 2016; Yang et al., 2017; Ufer and Ommer, 2017] using in some cases [Ufer and Ommer, 2017] pre-trained (but fixed)

convolutional neural network (CNN) descriptors. However, none of these methods train the image descriptor or the geometric model directly for semantic alignment.

Others [Novotny et al., 2017; Kim et al., 2018b, 2017] have investigated trainable image descriptors for semantic matching but have combined them with hand-engineered alignment models still rendering the alignment pipeline not trainable end-to-end.

Finally, recent work [Han et al., 2017; Rocco et al., 2017] has employed trainable CNN descriptors together with trainable geometric alignment methods. However, in [Han et al., 2017] the matching is learned at the object-proposal level and a non-trainable fusion step is necessary to output the final alignment making the method non end-to-end trainable. On the contrary, [Rocco et al., 2017] estimate a parametric geometric model, which can be converted into dense pixel correspondences in a differentiable way, making the method end-to-end trainable. However, the method is trained with strong supervision in the form of ground truth correspondences obtained from synthetically warped images, which significantly limits the appearance variation in the training data.

Contributions. We develop a network architecture where both the descriptor and the alignment model are trainable in an end-to-end manner from weakly supervised data. This enables training from real images with rich appearance variation and without the need for manual ground-truth correspondence. We demonstrate that the proposed approach significantly improves alignment results achieving state-of-the-art performance on several datasets for semantic alignment.

4.3 Weakly-supervised semantic alignment

This section presents a method for training a semantic alignment model in an end-to-end fashion using only weak supervision – the information that two images should match – but without access to the underlying geometric transformation at training time. The approach is outlined in Fig. 4-2. Namely, given a pair of images, an alignment network estimates the geometric transformation that aligns them. The

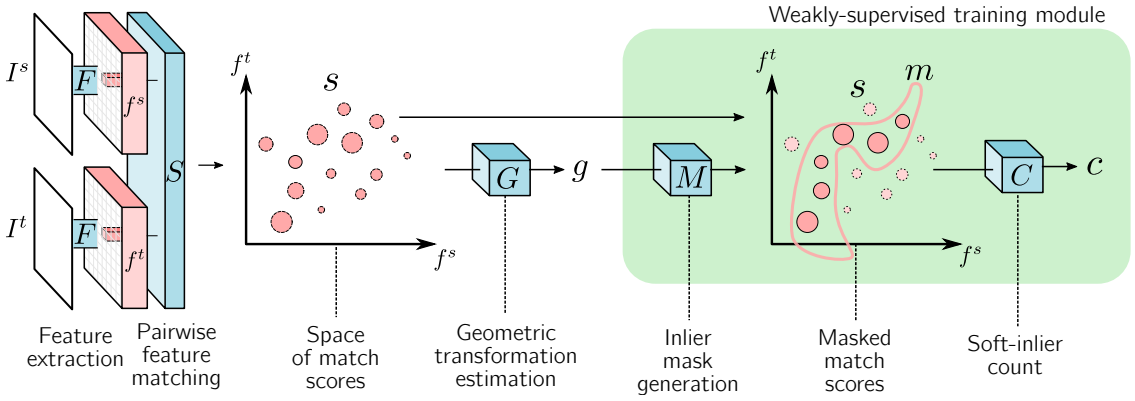


Figure 4-2: **End-to-end weakly-supervised alignment.** Source and target images (I^s, I^t) are passed through an alignment network used to estimate the geometric transformation g. Then, the soft-inlier count is computed (in green) by first finding the inlier region m in agreement with g, and then adding up the pairwise matching scores inside this area. The soft-inlier count is differentiable, which allows the whole model to be trained using back-propagation. Functions are represented in blue and tensors in pink.

quality of the estimated transformation is assessed using the proposed *soft-inlier count* which aggregates the observed evidence in the form of feature matches. The training objective then is to maximize the alignment quality for pairs of images which should match.

The key idea is that, instead of requiring strongly supervised training data in the form of known pairwise alignments and training the alignment network with these, the network is "forced" into learning to estimate good alignments in order to achieve high alignment scores (soft-inlier counts) for matching image pairs. The details of the alignment network and the soft-inlier count are presented next.

4.3.1 Semantic alignment network

In order to make use of the error signal coming from the soft-inlier count, our framework requires an alignment network which is trainable end-to-end. We build on the Siamese CNN architecture described in Chapter 3, illustrated in the left section of Fig. 4-2. The architecture is composed of three main stages – feature extraction, followed by feature matching and geometric transformation estimation – which we review below.

Feature extraction. The input source and target images, (I^s, I^t), are passed through two fully-convolutional feature extraction CNN branches, F, with shared weights. The resulting feature maps (f^s, f^t) are $h \times w \times d$ tensors which can be interpreted as dense $h \times w$ grids of d-dimensional local features $f_{ij:} \in \mathbb{R}^d$. These individual d-dimensional features are L2 normalized.

Pairwise feature matching. This stage computes all pairwise similarities, or match scores, between local features in the two images. This is done with the normalized correlation function $S : \mathbb{R}^{h\times w\times d} \times \mathbb{R}^{h\times w\times d} \to \mathbb{R}^{h\times w\times h\times w}$, defined as:

$$s_{ijkl} = S(f^s, f^t)_{ijkl} = \frac{\langle f^s_{ij:}, f^t_{kl:}\rangle}{\sqrt{\sum_{a,b}\langle f^s_{ab:}, f^t_{kl:}\rangle^2}}, \tag{4.1}$$

where the numerator in (4.1) computes the *raw* pairwise match scores by computing the dot product between features pairs. The denominator performs a normalization operation with the effect of down-weighing ambiguous matches, by penalizing features from one image which have multiple highly-rated matches in the other image. This is in line with the classical second nearest-neighbour test of Lowe [Lowe, 2004]. The resulting tensor s contains all normalized match scores between the source and target features.

Geometric transformation estimation. The final stage of the alignment network consists of estimating the parameters of a geometric transformation g given the match scores s. This is done by a transformation regression CNN, represented by the function G:

$$G : \mathbb{R}^{h\times w\times h\times w} \to \mathbb{R}^K, \quad g = G(s) \tag{4.2}$$

where K is the number of degrees of freedom, or parameters, of the geometric model; *e.g.* $K = 6$ for an affine model. The estimated transformation parameters g are used to define the 2-D warping $\mathcal{T}_g$:

$$\mathcal{T}_g : \mathbb{R}^2 \rightarrow \mathbb{R}^2, \quad (u^s, v^s) = \mathcal{T}_g(u^t, v^t) \tag{4.3}$$

where (u^t, v^t) are the spatial coordinates of the target image, and (u^s, v^s) the corresponding sampling coordinates in the source image. Using $\mathcal{T}_g$, it is possible to warp the source to the target image.

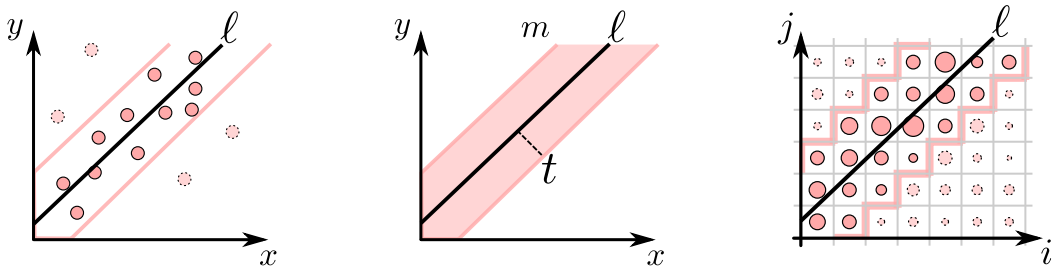
Note that all parts of the geometric alignment network are differentiable and therefore amenable to end-to-end training, including the feature extractor F which can learn better features for the task of semantic alignment.

4.3.2 Soft-inlier count

We propose the *soft-inlier count* used to automatically evaluate the estimated geometric transformation g. Making an effort to maximize this count provides the weak-supervisory signal required to train the alignment network, avoiding the need for expensive manual annotations for g. The soft-inlier count is inspired by the inlier count used in the robust RANSAC method [Fischler and Bolles, 1981], which is reviewed first.

RANSAC inlier count. For simplicity, let us consider the problem of fitting a line to a set of observed points p_i, with $i = 1, \dots N$, as illustrated in Fig. 4-3a. RANSAC proceeds by sampling random pairs of points used to propose line hypotheses, each of which is then scored using the inlier count, and the highest scoring line is chosen; here we only focus on the inlier count aspect of RANSAC used to score a hypothesis. Given a hypothesized line ℓ, the RANSAC inlier scoring function counts the number of observed points which are in agreement with this hypothesis, called the *inliers*. A point p is typically deemed to be an inlier iff its distance to the line is smaller than a chosen distance threshold t, *i.e.* $\mathrm{d}(p, \ell) < t$.

The RANSAC inlier count, c_R, can be formulated by means of an auxiliary indicator



(a) Inliers and outliers (b) Inlier mask function (c) Discretized space

Figure 4-3: **Line-fitting example.** (a) The line hypothesis ℓ can be evaluated in terms of the number of inliers. (b) The inlier mask m specifies the region where the inlier distance threshold is satisfied. (c) In the discretized space setting, where the match score s_{ij} exists for every point (i, j), the soft-inlier count is computed by summing up match scores masked by the inlier mask m from (b).

function illustrated in Fig. 4-3b, which we call the inlier mask function m:

$$c_R = \sum_i m(p_i), \text{ where } m(p) = \begin{cases} 1, & \text{if } \mathrm{d}(p, \ell) < t \\ 0, & \text{otherwise.} \end{cases} \tag{4.4}$$

Soft-inlier count. The RANSAC inlier count cannot be used directly in a neural network as it is not differentiable. Furthermore, in our setting there is no sparse set of matching points, but rather a match score for every match in a discretized match space. Therefore, we propose a direct extension, the *soft-inlier count*, which, instead of counting over a sparse set of matches, sums the match scores over all possible matches.

The running line-fitting example can now be revisited under the discrete-space conditions, as illustrated in Figure 4-3c. The proposed soft-inlier count for this case is:

$$c = \sum_{i,j} s_{ij} m_{ij}, \tag{4.5}$$

where s_{ij} is the match score at each grid point (i,j), and m_{ij} is the discretized inlier mask:

$$m_{ij} = \begin{cases} 1 & \text{if } \mathrm{d}\big((i, j), \ell\big) < t \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

Translating the discrete-space line-fitting example to our semantic alignment problem, s is a 4-D tensor containing scores for all pairwise feature matches between the two images (Section 4.3.1), and matches are deemed to be inliers if they fit the estimated geometric transformation g. More formally, the inlier mask m is now also a 4-D tensor, constructed by thresholding the transfer error:

$$m_{ijkl} = \begin{cases} 1 & \text{if } \mathrm{d}\Big((i,j), \mathcal{T}_g(k,l)\Big) < t \\ 0 & \text{otherwise,} \end{cases} \tag{4.7}$$

where $\mathcal{T}_g(k,l)$ are the estimated coordinates of target image's point (k,l) in the source image according to the geometric transformation g; $\mathrm{d}\Big((i,j), \mathcal{T}_g(k,l)\Big)$ is the transfer error as it measures how aligned is the point (i,j) in the source image, with the projection of the target image point (k,l) into the source image. The soft-inlier count c is then computed by summing the masked matching scores over the entire space of matches:

$$c = \sum_{i,j,k,l} s_{ijkl} m_{ijkl}. \tag{4.8}$$

Differentiability. The proposed soft-inlier count c is differentiable with respect to the transformation parameters g as long as the geometric transformation $\mathcal{T}_g$ is differentiable [Jaderberg et al., 2015], which is the case for a range of standard geometric transformations such as 2D affine, homography or thin-plate spline transformations. Furthermore, it is also differentiable w.r.t. the match scores, which facilitates training of the feature extractor.

Implementation as a CNN layer. The inlier mask m can be computed by warping an identity mask m^{Id} with the estimated transformation $\mathcal{T}_g$, where m^{Id} is constructed by thresholding the transfer error of the identity transformation:

$$m^{Id}_{ijkl} = \begin{cases} 1 & \mathrm{d}\Big((i,j), (k,l)\Big) < t \\ 0 & \text{otherwise.} \end{cases} \tag{4.9}$$

The warping is implemented using a spatial transformer layer [Jaderberg et al.,

2015], which consists of a grid generation layer and a bilinear sampling layer. Both of these functions are readily available in most deep learning frameworks.

Optimization objective. Given a training pair of images that should match, the goal is to maximize the soft-inlier count c, or, equivalently, to minimize $\mathcal{L} = -c$.

Analogy to RANSAC. Please also note that our method is similar in spirit to RANSAC [Fischler and Bolles, 1981], where (i) transformations are proposed (by random sampling) and then (ii) scored by their support (number of inliers). In our case, during training (i) the transformations are proposed (estimated) by the regressor network G and (ii) scored using the proposed soft-inlier score. The gradient of this score is used to improve both the regressor G and feature extractor F (see Fig. 4-2). In turn, the regressor produces better transformations and the feature extractor better feature matches that maximize the soft-inlier score on training images.

4.4 Evaluation and results

In this section we provide implementation details, benchmarks used to evaluate our approach, and quantitative and qualitative results.

4.4.1 Implementation details

Semantic alignment network. For the underlying semantic alignment network, we use the best-performing architecture from the code release of Rocco et al. [2017] which employs a ResNet-101 model [He et al., 2016], cropped after `conv4-23`, as the feature extraction CNN F. Note that this is a better performing model than the one described in the original paper [Rocco et al., 2017], mainly due to use of ResNet versus VGG-16 [Simonyan and Zisserman, 2015]. Given an image pair, the model produces a thin-plate spline geometric transformation $\mathcal{T}_g$ which aligns the two images; $\mathcal{T}_g$ has 18 degrees of freedom. The network is initialized with the pre-trained weights from the code release of Rocco et al. [2017], and we finetune it with our weakly supervised

method. Note that the initial model has been trained in a self-supervised way from synthetic data, not requiring human supervision [Rocco et al., 2017], therefore not affecting our claim of weakly supervised training[1].

Training details. Training and validation image pairs are obtained from the training set of PF-PASCAL, described in Section 4.4.2. All input images are resized to 240×240, and the value $t = L/30$ (where $L = h = w$ is the size of the extracted feature maps) was used for the transfer error threshold. The whole model is trained end-to-end, including the affine parameters in the batch normalization layers. However, the running averages of the batch normalization layers are kept fixed, in order to be less dependent on the particular statistics of the training dataset. The network is implemented in PyTorch [Paszke et al., 2017] and trained using the Adam optimizer [Kingma and Ba, 2015] with learning rate $5 \cdot 10^{-8}$, no weight decay and batch size of 16. The training dataset is augmented by horizontal flipping, swapping the source and target images, and random cropping. Early stopping is required to avoid overfitting, given the small size of the training set. This results in 13 training epochs, taking about an hour on a modern GPU.

4.4.2 Evaluation benchmarks

Evaluation is performed on three standard image alignment benchmarks: PF-PASCAL, Caltech-101 and TSS.

PF-PASCAL [Ham et al., 2017]. This dataset contains 1351 semantically related image pairs from 20 object categories, which present challenging appearance differences and background clutter. We use the split proposed in [Han et al., 2017], which divides the dataset into roughly 700 pairs for training, 300 pairs for validation, and 300 pairs for testing. Keypoint annotations are provided for each image pair, which are used only for evaluation purposes. Alignment quality is evaluated in terms of the percentage of correct keypoints (PCK) metric [Yang and Ramanan, 2013], which counts the

[1] The initial model is trained with a supervised loss, but the "supervision" is automatic due to the use of synthetic data.

number of keypoints which have a transfer error below a given threshold. We follow the procedure employed in [Han et al., 2017], where keypoint (x, y) coordinates are normalized in the $[0, 1]$ range by dividing with the image width and height respectively, and the value $\alpha = 0.1$ is employed as the distance threshold.

Caltech-101 [Fei-Fei et al., 2006]. Although originally introduced for the image classification task, the dataset was adopted in [Kim et al., 2013] for assessing semantic alignment, and has been then extensively used for this purpose [Ham et al., 2017; Kim et al., 2018b; Han et al., 2017; Rocco et al., 2017]. The evaluation is performed on 1515 semantically related image pairs, 15 pairs for each of the 101 object categories of the dataset. The semantic alignment is evaluated using three different metrics: (i) the label transfer accuracy (LT-ACC); (ii) the intersection-over-union (IoU), and; (iii) the object localization error (LOC-ERR). The label transfer accuracy and the intersection-over-union both measure the overlap between the annotated foreground object segmentation masks, with the former putting more emphasis on the background class and the latter on the foreground object. The localization error computes a dense displacement error. However, given the lack of dense displacement annotations, the metric computes the ground-truth transformation from the source and target bounding boxes, thus assuming that the transformation is a simple translation with axis-aligned anisotropic scaling. This assumption is unrealistic as, amongst others, it does not cover rotations, affine or deformable transformations. Therefore, we believe that LOC-ERR should not be reported any more, but report it here for completeness and in order to adhere to the currently adopted evaluation protocol.

TSS [Taniai et al., 2016]. The recently introduced TSS dataset contains 400 semantically related image pairs, which are split into three different subsets: FG3DCar, JODS and PASCAL, according to the origin of the images. Ground-truth flow is provided for each pair, which was obtained by manual annotation of sparse keypoints, followed by automatic densification using an interpolation algorithm. The evaluation metric is the PCK computed densely over the foreground object. The distance

threshold is defined as $\alpha \max(w^s, h^s)$ with (w^s, h^s) being the dimensions of the source image, and $\alpha = 0.05$.

Assessing generalization. We train a single semantic alignment network with the 700 training pairs from PF-PASCAL *without* using the keypoint annotations, and stress that our weakly-supervised training objective only uses the information that the image pair should match. *The same* model is then used for all experiments – evaluation on the test sets of PF-PASCAL, Caltech-101 and TSS datasets. This poses an additional difficulty as these datasets contain images of different object categories or of different nature. While PF-PASCAL contains images of common objects such as car, bicycle, boat, etc, Caltech-101 contains images of much less common categories such as accordion, buddha or octopus. On the other hand, while the classes of TSS do appear in PF-PASCAL, the pose differences in TSS are usually smaller than in PF-PASCAL, which modifies the challenge into obtaining a very precise alignment.

4.4.3 Results

In the following, our alignment network trained with *weak supervision* is compared to the state-of-the-art alignment methods, many of which require *manual annotations* or *strong supervision* (*cf.* Table 4.1).

PF-PASCAL. From Table 4.2 it is clear that our method sets the new state-of-the-art, achieving an overall PCK of 75.8%, which is a 3.6% improvement over the best competitor [Han et al., 2017]. This result is impressive as the two methods are trained on the same image pairs, with ours being weakly supervised while [Han et al., 2017] make use of bounding box annotations.

Method	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	d.table	dog	horse	moto	person	plant	sheep	sofa	train	tv	**all**
PF-LOM (HOG)	73.3	74.4	54.4	50.9	49.6	73.8	72.9	63.6	46.1	79.8	42.5	48.0	68.3	66.3	42.1	62.1	65.2	57.1	64.4	58.0	62.5
SCNet-A (VGG-16)	67.6	72.9	69.3	59.7	**74.5**	72.7	73.2	59.5	51.4	78.2	39.4	50.1	67.0	62.1	69.3	68.5	78.2	63.3	57.7	59.8	66.3
SCNet-AG (VGG-16)	83.9	81.4	70.6	62.5	60.6	81.3	81.2	59.5	53.1	81.2	**62.0**	58.7	65.5	73.3	51.2	58.3	60.0	69.3	61.5	**80.0**	69.7
SCNet-AG+ (VGG-16)	**85.5**	84.4	66.3	**70.8**	57.4	82.7	82.3	71.6	**54.3**	**95.8**	55.2	59.5	68.6	75.0	56.3	60.4	60.0	**73.7**	**66.5**	76.7	72.2
CNNGeo (VGG-16)	75.2	80.1	73.4	59.7	43.8	77.9	84.0	67.7	44.3	89.6	33.9	67.1	60.5	72.6	54.0	41.0	60.0	45.1	58.3	37.2	65.0
CNNGeo (ResNet-101)	82.4	80.9	**85.9**	47.2	57.8	83.1	**92.8**	**86.9**	43.8	91.7	28.1	**76.4**	70.2	**76.6**	68.9	65.7	**80.0**	50.1	46.3	60.6	71.9
Proposed	83.7	**88.0**	83.4	58.3	68.8	**90.3**	92.3	83.7	47.4	91.7	28.1	76.3	**77.0**	76.0	**71.4**	**76.2**	**80.0**	59.5	62.3	63.9	**75.8**

Table 4.2: **Per-class PCK on the PF-PASCAL dataset.** We compare our method against PF-LOM [Ham et al., 2016], SCNet [Han et al., 2017] and CNNGeo [Rocco et al., 2017]. Our method sets the new state-of-the-art on this dataset.

The benefits of weakly supervised training can be seen by comparing our method with the ResNet-101+CNNGeo [Rocco et al., 2017] approach presented in chapter 3. The two use the same base alignment network (*cf.* Section 4.4.1), but ResNet-101+CNNGeo was trained only on synthetically deformed image pairs, while ours employs the proposed weakly supervised fine-tuning. The 3.9% boost clearly demonstrates the advantage obtained by training on real image pairs and thus encountering rich appearance variations, as opposed to using synthetically transformed pairs in ResNet-101+CNNGeo [Rocco et al., 2017].

Caltech-101. Table 4.3 presents the quantitative results for this dataset. The proposed method beats state-of-the-art results in terms of the label-transfer accuracy and intersection-over-union metrics. Weakly supervised training again improves the results, by 2%, over the synthetically trained ResNet-101+CNNGeo. In terms of the localization-error metric, our model does not attain state-of-the-art performance, but we argue that this metric is not a good indication of the alignment quality, as explained in section 4.4.2. This claim is further backed up by noticing that the relative ordering of various methods based on this metric is in direct opposition with the other two metrics.

Method	LT-ACC	IoU	LOC-ERR
HOG+PF-LOM [Ham et al., 2017]	0.78	0.50	0.26
FCSS+SIFT Flow [Kim et al., 2018b]	0.80	0.50	0.21
FCSS+PF-LOM [Kim et al., 2018b]	0.83	0.52	0.22
VGG-16+SCNet-A [Han et al., 2017]	0.78	0.50	0.28
VGG-16+SCNet-AG [Han et al., 2017]	0.78	0.50	0.27
VGG-16+SCNet-AG+ [Han et al., 2017]	0.79	0.51	0.25
HOG+OADSC [Yang et al., 2017]	0.81	0.55	**0.19**
VGG-16+CNNGeo [Rocco et al., 2017]	0.80	0.55	0.26
ResNet-101+CNNGeo [Rocco et al., 2017]	0.83	0.61	0.25
Proposed	**0.85**	**0.63**	0.24

Table 4.3: **Evaluation results on the Caltech-101 dataset.**

Method	FG3D.	JODS	PASC.	avg.
HOG+PF-LOM [Ham et al., 2017]	0.786	0.653	0.531	0.657
HOG+TSS [Taniai et al., 2016]	0.830	0.595	0.483	0.636
FCSS+SIFT Flow [Kim et al., 2018b]	0.830	0.656	0.494	0.660
FCSS+PF-LOM [Kim et al., 2018b]	0.839	0.635	0.582	0.685
HOG+OADSC [Yang et al., 2017]	0.875	0.708	**0.729**	**0.771**
FCSS+DCTM [Kim et al., 2017]	0.891	0.721	0.610	0.740
VGG-16+CNNGeo [Rocco et al., 2017]	0.839	0.658	0.528	0.675
ResNet-101+CNNGeo [Rocco et al., 2017]	0.901	**0.764**	0.563	0.743
Proposed	**0.903**	**0.764**	0.565	0.744

Table 4.4: **Evaluation results on the TSS dataset.**

TSS. The quantitative results for the TSS dataset are presented in Table 4.4. We set the state-of-the-art for two of the three subsets of the TSS dataset: FG3DCar and JODS. Although our weakly supervised training provides an improvement over the base alignment network, ResNet-101+CNNGeo, the gain is modest. We believe the reason is a very different balancing of classes in this dataset compared to our training. Recall our model is trained *only once* on the PF-PASCAL dataset, and is then applied without any further training on TSS and Caltech-101.

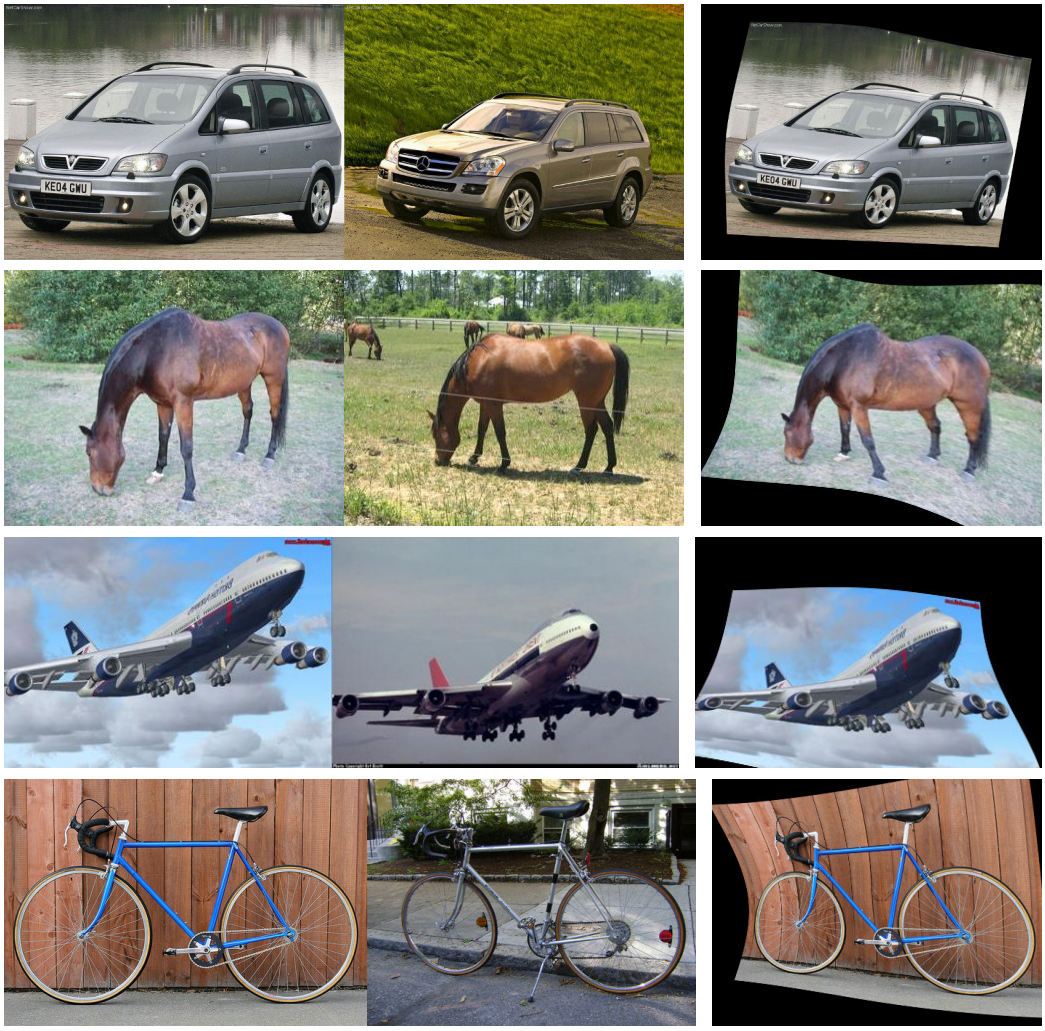
Qualitative results. Figures 4-4a, 4-4b and 4-5 show qualitative results on the Caltech-101, TSS and PF-PASCAL datasets, respectively. Our method is able to align images across prominent viewpoint changes, in the presence of significant clutter, while simultaneously tolerating large intra-class variations.

4.5 Conclusions

In this chapter we have presented a network architecture and training procedure for semantic image alignment inspired by the robust inlier scoring used in the widely successful RANSAC fitting algorithm [Fischler and Bolles, 1981]. The architecture requires supervision only in the form of matching image pairs and sets the new state-of-the-art on multiple standard semantic alignment benchmarks, even beating alignment methods that require geometric supervision at training time. However, handling

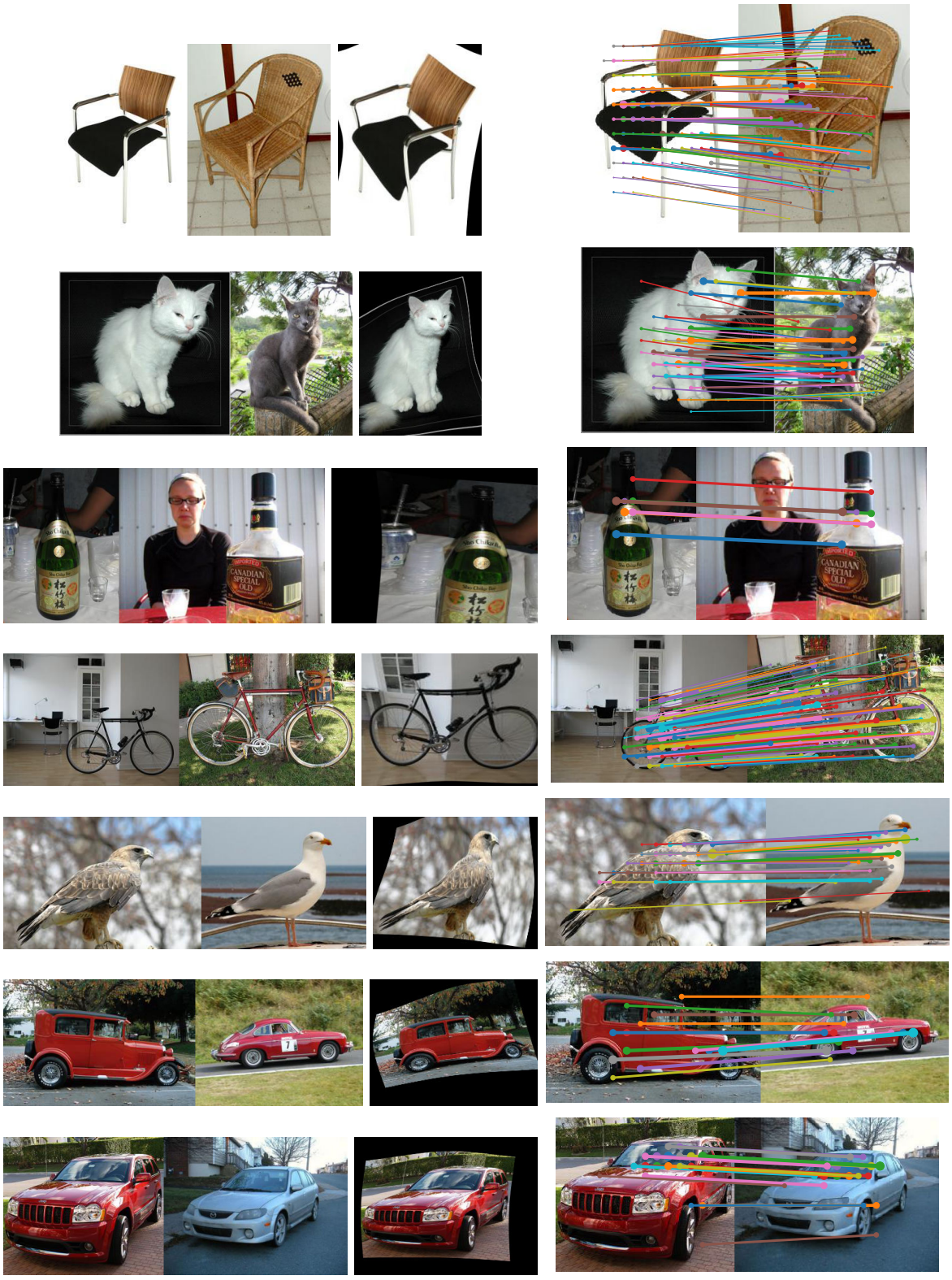
multiple objects and non-matching image pairs still remains an open challenge. These results open-up the possibility of learning powerful correspondence networks from large-scale datasets such as ImageNet.

(a) Caltech-101



(b) TSS

Figure 4-4: **Alignment examples on the Caltech-101 and TSS datasets.** Each row shows the (left) source and (middle) target images, and (right) the automatic semantic alignment.



(a) Semantic alignment (b) Strongest inlier matches

Figure 4-5: **Alignment examples on the PF-PASCAL dataset.** Each row corresponds to one example. (a) shows the (right) automatic semantic alignment of the (left) source and (middle) target images. (b) shows the strongest inlier feature matches.

Chapter 5

Neighbourhood consensus networks

In this chapter, we develop a trainable model for reliably estimating sets of semi-dense correspondences between pairs of images. Contrary to the approach used in Chapters 3 and 4 where a global geometric model was used to guide the matching, in this chapter we rely on semi-local constraints for obtaining correspondences in challenging situations such as in scenes with strong appearance differences or repetitive patterns. In particular, the contributions of this chapter are threefold. First, we develop an end-to-end trainable convolutional neural network architecture that identifies sets of spatially consistent matches by analyzing neighbourhood consensus patterns in the 4D space of all possible correspondences between a pair of images without the need for a global geometric transformation. Second, we demonstrate that the model can be trained effectively from weak supervision in the form of matching and non-matching image pairs without the need for costly manual annotation of point to point correspondences. Third, we show the proposed neighbourhood consensus network can be applied to a range of matching tasks including both category- and instance-level matching, obtaining the state-of-the-art results on the PF, TSS, InLoc and HPatches benchmarks.

5.1 Introduction

Finding visual correspondences is one of the fundamental image understanding problems with applications in 3D reconstruction [Agarwal et al., 2011], visual localization [Sattler et al., 2018; Taira et al., 2018] or object recognition [Liu et al., 2011]. In recent years, significant effort has gone into developing trainable image representations for finding correspondences between images under strong appearance changes caused by viewpoint or illumination variations [Jahrer et al., 2008; Fischer et al., 2014; Zagoruyko and Komodakis, 2015; Han et al., 2015; Balntas et al., 2016a; Simonyan et al., 2014; Simo-Serra et al., 2015; Balntas et al., 2016b; Yi et al., 2016]. However, unlike in other visual recognition tasks, such as image classification or object detection, where trainable image representations have become the *de facto* standard, the performance gains obtained by trainable features over the classic hand-crafted ones have been only modest at best [Schonberger et al., 2017].

One of the reasons for this plateauing performance could be the currently dominant approach for finding image correspondence based on matching *individual* image features. While we have now better local patch descriptors, the matching is still performed by variants of the nearest neighbour assignment in a feature space followed by separate disambiguation stages based on geometric constraints. This approach has, however, fundamental limitations. Imagine a scene with textureless regions or repetitive patterns, such as a corridor with almost textureless walls and only few distinguishing features. A small patch of an image, depicting a repetitive pattern or a textureless area, is indistinguishable from other portions of the image depicting the same repetitive or textureless pattern. Such matches will be either discarded [Lowe, 2004] or incorrect. As a result, matching individual patch descriptors will often fail in such challenging situations.

In this chapter, we take a different direction and develop a trainable neural network architecture that disambiguates such challenging situations by analyzing local neighbourhood patterns in a full set of dense correspondences. The intuition is the following: in order to disambiguate a match on a repetitive pattern, it is necessary to

analyze a larger context of the scene that contains a unique non-repetitive feature. The information from this unique match can then be propagated to the neighbouring uncertain matches. In other words, the certain unique matches will *support* the close-by uncertain ambiguous matches in the image.

This powerful idea goes back to at least 1990s [Zhang et al., 1995; Schmid and Mohr, 1997; Schaffalitzky and Zisserman, 2002a; Sivic and Zisserman, 2003; Bian et al., 2017], and is typically known as *neighbourhood consensus* or more broadly as *semi-local constraints*. The neighbourhood consensus has been typically carried out on sparsely detected local invariant features as a filtering step performed *after* a hard assignment of features by nearest neighbour matching using the Euclidean distance in the feature space. Furthermore, the neighbourhood consensus has been evaluated by manually engineered criteria, such as a certain number of locally consistent matches [Schaffalitzky and Zisserman, 2002a; Sivic and Zisserman, 2003; Bian et al., 2017], or consistency in geometric parameters including distances and angles between matches [Zhang et al., 1995; Schmid and Mohr, 1997].

In this chapter, we go one step further and propose a way of *learning* neighbourhood consensus constraints directly from training data. Moreover, we perform neighbourhood consensus *before* hard assignment of feature correspondence; that is, on the complete set of dense pair-wise matches. In this way, the decision on matching assignment is done only after taking into account the spatial consensus constraints, hence avoiding errors due to early matching decisions on ambiguous, repetitive or textureless matches.

Contributions. The contributions of this chapter are threefold: **First**, we develop a neighbourhood consensus network – a convolutional neural network architecture for dense matching that learns local geometric constraints between neighbouring correspondences without the need for a global geometric model. **Second**, we show that parameters of this network can be trained from scratch using a weakly supervised loss-function that requires supervision at the level of image pairs without the need for manually annotating individual correspondences. **Third**, we show

that the proposed model is applicable to a range of matching tasks producing high-quality dense correspondences, achieving state-of-the-art results on both category- and instance-level matching benchmarks. Code, training data and models are available at http://www.di.ens.fr/willow/research/ncnet/.

5.2 Related work

The method we present in this chapter relates to several lines of research, which we review below.

Matching with hand-crafted image descriptors. Traditionally, correspondences between images have been obtained by hand crafted local invariant feature detectors and descriptors [Lowe, 2004; Mikolajczyk and Schmid, 2002; Tuytelaars and Mikolajczyk, 2008] that were extracted from the image with a controlled degree of invariance to local geometric and photometric transformations. Candidate (tentative) correspondences were then obtained by variants of nearest neighbour matching. Strategies for removing ambiguous and non-distinctive matches include the widely used second nearest neighbour ratio test [Lowe, 2004], or enforcing matches to be mutual nearest neighbours. Both approaches work well for many applications, but have the disadvantage of discarding many correct matches, which can be problematic for challenging scenes, such as indoor spaces considered in this chapter that include repetitive and textureless areas. While successful, handcrafted descriptors have only limited tolerance to large appearance changes beyond the built-in invariance.

Matching with trainable descriptors. The majority of trainable image descriptors are based on convolutional neural networks (CNNs) and typically operate on patches extracted using a feature detector such as DoG [Lowe, 2004], yielding a sparse set of descriptors [Jahrer et al., 2008; Fischer et al., 2014; Balntas et al., 2016a; Simonyan et al., 2014; Simo-Serra et al., 2015; Balntas et al., 2016b] or use a pre-trained image-level CNN feature extractor [Noh et al., 2017; Savinov et al., 2017]. Others have recently developed trainable methods that comprise both feature

detection and description [Yi et al., 2016; Choy et al., 2016; Noh et al., 2017]. The extracted descriptors are typically compared using the Euclidean distance, but an appropriate similarity score can be also learnt in a discriminative manner [Zagoruyko and Komodakis, 2015; Han et al., 2015], where a trainable model is used to both extract descriptors and produce a similarity score. Finding matches consistent with a geometric model is typically performed in a separate post-processing stage [Long et al., 2014; Jahrer et al., 2008; Fischer et al., 2014; Balntas et al., 2016a; Simonyan et al., 2014; Simo-Serra et al., 2015; Balntas et al., 2016b; Yi et al., 2016; Choy et al., 2016; Noh et al., 2017].

Trainable image alignment. Recently, end-to-end trainable methods have been developed to produce correspondences between images according to a parametric geometric model, such as an affine, perspective or thin-plate spline transformation [Rocco et al., 2017, 2018a]. In addition, Recurrent Transformer Nets (RTN) [Kim et al., 2018a] employ locally-varying affine deformation fields. In these works, all pairwise feature matches are computed and used to estimate the geometric transformation parameters using a CNN. Unlike previous methods that capture only a sparse set of correspondences, this geometric estimation CNN captures interactions between a full set of dense correspondences. However, these methods currently only estimate a low complexity parametric transformation, and therefore their application is limited to only coarse image alignment tasks. In contrast, we target a more general problem of identifying reliable correspondences between images of a general 3D scene. Our approach is not limited to a low dimensional parametric model, but outputs a generic set of locally consistent image correspondences, applicable to a wide range of computer vision problems ranging from category-level image alignment to camera pose estimation. The proposed method builds on the classical ideas of neighbourhood consensus, which we review next.

Match filtering by neighbourhood consensus. Several strategies have been introduced to decide whether a match is correct or not, given the supporting evidence

from the neighbouring matches. The early examples analyzed the patterns of distances [Zhang et al., 1995] or angles [Schmid and Mohr, 1997] between neighbouring matches. Later work simply counts the number of consistent matches in a certain image neighbourhood [Schaffalitzky and Zisserman, 2002a; Sivic and Zisserman, 2003], which can be built in a scale invariant manner [Sattler et al., 2009], using a regular image grid [Bian et al., 2017], or an adaptive neighbourhood size by considering a certain number of nearby matches [Ma et al., 2019]. While simple, these techniques have been remarkably effective in removing random incorrect matches and disambiguating local repetitive patterns [Sattler et al., 2009]. Inspired by this simple yet powerful idea we develop a neighbourhood consensus network – a convolutional neural architecture that (i) analyzes the *full set of dense matches* between a pair of images and (ii) *learns* patterns of locally consistent correspondences directly from data.

Other modern match filtering methods. While the idea of using neighbourhood consensus to remove outlier matches dates back to the 1990s [Zhang et al., 1995; Schmid and Mohr, 1997], it is still an actively researched topic. Recently, Bian et al. [Bian et al., 2017] proposed the Grid-based Motion Statistics (GMS) approach, where the images to be matched are partitioned into a set of cells and the number of matches between each cell are used to distinguish inliers from outliers. Also, Ma et al. [Ma et al., 2019] propose the Locality Preserving Matching (LPM) approach, where the sizes of the neighbourhoods are not explicitly defined, but rather inferred by nearest-neighbour search, and which proposes an optimization scheme to determine inliers by minimizing a global cost function. While these methods build on the idea of neighbourhood consensus, they are manually engineered and have no trainable parameters. Our proposed neighbourhood consensus network seeks to combine the power of the neighbourhood consensus approach with that of trainable convolutional neural networks. Other recent trainable methods for match filtering have also been proposed [Yi et al., 2018; Zhang et al., 2019; Brachmann and Rother, 2019], although they are particularly focused on robust estimation of the essential and fundamental matrices of two-view geometry. In particular, the Context Normalization Network

(CNe) of Yi et al. [Yi et al., 2018] combines a deep fully connected network (MLP) that operates on the level of individual 4D match coordinates with a Context Normalization (CN) Layer which acts at a global, image level. We compare our proposed method with GMS [Bian et al., 2017], LPM [Ma et al., 2019] and CNe [Yi et al., 2018].

Flow and disparity estimation. Related are also methods that estimate optical flow or stereo disparity such as [Lucas and Kanade, 1981; Horn and Schunck, 1981; Hirschmüller, 2007; Sun et al., 2010; Brox and Malik, 2010], or their trainable counterparts [Dosovitskiy et al., 2015; Sun et al., 2018; Kendall et al., 2017]. These works also aim at establishing reliable point to point correspondences between images. However, we address a more general matching problem where images can have large viewpoint changes (indoor localization) or major changes in appearance (category-level matching). This is different from optical flow where image pairs are usually consecutive video frames with small viewpoint or appearance changes, and stereo where matching is often reduced to a local search around epipolar lines. The optical flow and stereo problems are well addressed by specialized methods that explicitly exploit the problem constraints (such as epipolar line constraint, small motion, smoothness, etc.).

5.3 Proposed approach

In this chapter, we combine the robustness of neighbourhood consensus filtering with the power of trainable neural architectures. We design a model which learns to discriminate a reliable match by recognizing patterns of supporting matches in its neighbourhood. Furthermore, we do this in a fully differentiable way, such that this trainable matching module can be directly combined with strong CNN image descriptors. The resulting pipeline can then be trained in an end-to-end manner for the task of feature matching. An overview of our proposed approach is presented in Fig. 5-1. There are five main components: (i) dense feature extraction and matching, (ii) the neighbourhood consensus network, (iii) a soft mutual nearest neighbour filtering, (iv) extraction of correspondences from the output 4D filtered match tensor, and (v)

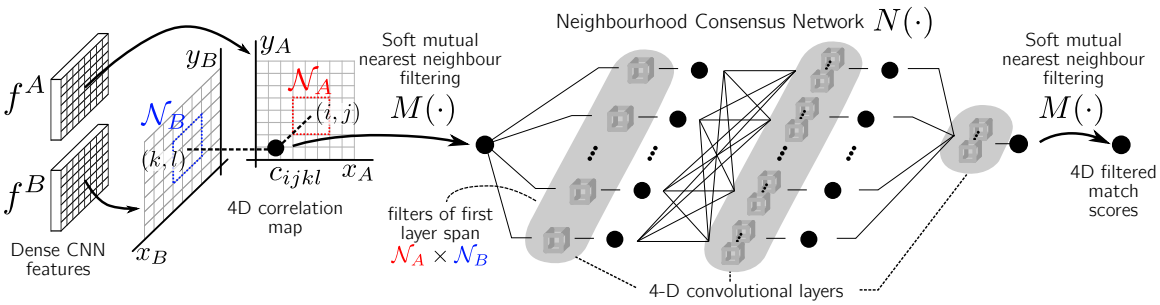


Figure 5-1: **Overview of the proposed method.** A fully convolutional neural network is used to extract dense image descriptors f^A and f^B for images I_A and I_B, respectively. Scores for all pairs of individual feature matches f^A_{ij} and f^B_{kl} are stored in the 4-D correlation map c_{ijkl} (here shown as a 3-D illustration). These matches are further processed by the proposed soft-nearest neighbour filtering and neighbourhood consensus network to produce the final set of output correspondences.

weakly supervised training loss. These components are described next.

5.3.1 Dense feature extraction and matching

In order to produce an end-to-end trainable model, we follow the common practice of using a deep convolutional neural network (CNN) as a dense feature extractor.

Then, given an image I, this feature extractor will produce a dense set of descriptors, $\{f^I_{ij}\} \in \mathbb{R}^d$, with indices $i = 1, \ldots, h$ and $j = 1, \ldots, w$, and (h, w) denoting the number of features along image height and width (*i.e.* the spatial resolution of the features), and d the dimensionality of the features.

While classic hand-crafted neighbourhood consensus approaches are applied *after* a hard assignment of matches is done, this is not well suited for developing a matching method that is differentiable and amenable for end-to-end training. The reason is that the step of selecting a particular match is not differentiable with respect to the set of all the possible features. In addition, in case of repetitive features, assigning the match to the first nearest neighbour might result in an incorrect match, in which case the hard assignment would lose valuable information about the subsequent closest neighbours.

Therefore, in order to have an approach that is amenable to end-to-end training, all pairwise feature matches need to be computed and stored. For this we use an

approach similar to the one presented in Chapter 3. Given two sets of dense feature descriptors $f^A = \{f^A_{ij}\}$ and $f^B = \{f^B_{ij}\}$ corresponding to the images to be matched, the exhaustive pairwise cosine similarities between them are computed and stored in a 4-D tensor $c \in \mathbb{R}^{h\times w\times h\times w}$ referred to as *correlation map*, where:

$$c_{ijkl} = \frac{\langle f^A_{ij}, f^B_{kl}\rangle}{\|f^A_{ij}\|_2 \|f^B_{kl}\|_2}. \tag{5.1}$$

Note that, by construction, elements of c in the vicinity of index $ijkl$ correspond to matches between features that are in the local neighbourhoods $\mathcal{N}_A$ and $\mathcal{N}_B$ of descriptors f^A_{ij} in image A and f^B_{kl} in image B, respectively, as illustrated in Fig. 5-1; this structure of the 4-D correlation map tensor c will be exploited in the next section.

5.3.2 Neighbourhood consensus network

The correlation map contains the scores of *all* pairwise matches. In order to further process and filter the matches, we propose to use 4-D convolutional neural network (CNN) for the neighbourhood consensus task (denoted by $N(\cdot)$), which is illustrated in Fig. 5-1.

Determining the correct matches from the correlation map is, *a priori*, a significant challenge. Note that the number of correct matches are of order of hw, while the size of the correlation map is of the order of $(hw)^2$. This means that the great majority of the information in the correlation map corresponds to *matching noise* due to incorrectly matched features.

However, supported by the idea of neighbourhood consensus presented in Sec. 4.1, we can expect correct matches to have a coherent set of supporting matches surrounding them in the 4-D space. These geometric patterns are equivariant with translations in the input images; that is, if the images are translated, the matching pattern is also translated in the 4-D space by an equal amount. This property motivates the use of 4-D convolutions for processing the correlation map as the same operations should be performed regardless of the location in the 4-D space. This is analogous to the motivation for using 2-D convolutions to process individual images – it makes sense

to use convolutions, instead of for example a fully connected layer, in order to profit from weight sharing and keep the number of trainable parameters low. Furthermore, it facilitates sample-efficient training as a single training example provides many error signals to the convolutional weights, since the *same* weights are applied at all positions of the correlation map. Finally, by processing matches with a 4D convolutional network we establish a strong locality prior on the relationships between the matches. That is, by design, the network will determine the quality of a match by examining only the information in a local 2D neighbourhood in each of the two images.

The proposed neighbourhood consensus network has several convolutional layers, as illustrated in Fig. 5-1, each followed by ReLU non-linearities. The convolutional filters of the first layer of the proposed CNN span a local 4-D region of the matches space, which corresponds to the Cartesian product of local neighbourhoods $\mathcal{N}_A$ and $\mathcal{N}_B$ in each image, respectively. Therefore, each 4-D filter of the first layer can process and detect patterns in all pairwise matches of these two neighbourhoods. This first layer has N_1 filters that can specialize in learning different local geometric deformations, producing N_1 output channels, that correspond to the agreement with these local deformations at each 4-D point of the correlation tensor. These output channels are further processed by subsequent 4-D convolutional layers. The aim is that these layers capture more complex patterns by combining the outputs from the previous layer, analogously to what has been observed for 2-D CNNs [Zeiler and Fergus, 2014]. Finally, the neighbourhood consensus CNN produces a single channel output, which has the same dimensions as the 4D input matches.

To make the method invariant to the particular order of the input images, that is, that it will produce the same matches regardless of whether an image pair is input to the net as (I^A, I^B) or (I^B, I^A), we define the following *symmetric* version of the network N by applying it twice in the following way:

$$S(c) = N(c) + \left(N(c^{\mathrm{T}})\right)^{\mathrm{T}}, \tag{5.2}$$

where c is the correlation map defined in (5.1) and by c^{T} we mean swapping the pairs

of dimensions corresponding to the first and second images: $\left(c^{\mathrm{T}}\right)_{ijkl} = c_{klij}$. This final output constitutes the *filtered matches* using the neighbourhood consensus network, where matches with inconsistent *local* patterns are downweighted or removed. Further filtering can be done by means of a *global* filtering strategy, as presented next.

5.3.3 Soft mutual nearest neighbour filtering

Although the proposed neighbourhood consensus network can suppress and amplify matches based on the supporting evidence in their neighbourhoods – that is, at a semi-local level – it cannot enforce global constraints on matches, such as being a *reciprocal* match, where matched features are required to be mutual nearest neighbours:

$$(f^A_{ab}, f^B_{cd}) \text{ mutual N.N.} \iff \begin{cases} (a,b) = \text{argmin}_{ij} \, \|f^A_{ij} - f^B_{cd}\| \\ (c,d) = \text{argmin}_{kl} \, \|f^A_{ab} - f^B_{kl}\|. \end{cases} \tag{5.3}$$

Filtering the matches by imposing the hard mutual nearest neighbour condition expressed by (5.3) would eliminate the great majority of candidate matches, which makes it unsuitable for usage in an end-to-end trainable approach, as this hard decision is non-differentiable.

We therefore propose a softer version of the mutual nearest neighbour filtering ($M(\cdot)$), both in the sense of *softer decision* and *better differentiability properties*, that can be applied on dense 4-D match scores:

$$c' = M(c), \quad \text{where} \quad c'_{ijkl} = r^A_{ijkl} r^B_{ijkl} c_{ijkl}, \tag{5.4}$$

and r^A_{ijkl} and r^B_{ijkl} are the ratios of the score of the particular match c_{ijkl} with the best scores along each pair of dimensions corresponding to images A and B respectively:

$$r^A_{ijkl} = \frac{c_{ijkl}}{\max_{ab} c_{abkl}}, \quad \text{and} \quad r^B_{ijkl} = \frac{c_{ijkl}}{\max_{cd} c_{ijcd}}. \tag{5.5}$$

This soft mutual nearest neighbour filtering operates as a gating mechanism on the input, downweighting the scores of matches that are not mutual nearest neighbours.

Note that the proposed formulation is indeed a *softer* version of the mutual nearest neighbours criterion as c'_{ijkl} equals the matching score c_{ijkl} iff (f^A_{ij}, f^B_{kl}) are mutual nearest neighbours, and is decreased to a value in $[0, c_{ijkl})$ otherwise. On the contrary, the "hard" mutual nearest neighbour matching would assign $c'_{ijkl} = 0$ in the latter case.

While this filtering step has no trainable parameters, it can be inserted in the CNN pipeline at both training and evaluation stages, and it will help to enforce the global *reciprocity* constraint on matches. In the proposed approach, the soft mutual nearest neighbour filtering is used to filter both the correlation map, as well as the output of the neighbourhood consensus CNN, as illustrated in Fig. 5-1.

5.3.4 Lightweight model

Given the correlation tensor c, and the previously defined symmetric neighbourhood-consensus network (S) and soft mutual nearest neighbour filtering (M) operations, the full proposed method can be expressed as:

$$\text{NCNet}(c) = (M \circ S \circ M)(c). \tag{5.6}$$

However, due to memory requirements, one might prefer to use *lighter-weight* neighbourhood consensus network N instead of its symmetric version S:

$$\text{L-NCNet}(c) = (M \circ N \circ M)(c), \tag{5.7}$$

Due to the lower memory requirements, L-NCNet is useful for running the network on higher resolution images. Note that switching from NCNet to the lightweight L-NCNet simply results in neglecting the second term of (5.2) and can be done without retraining.

5.3.5 Extracting correspondences from the correlation map

Suppose that we want to match two images I^A and I^B, whose *raw* correlation map is c. Then, the output of our model $c' = \text{NCNet}(c)$ is a 4-D *filtered* correlation map, which contains (filtered) scores for all pairwise matches. However, for various applications, such as image warping, geometric transformation estimation, pose estimation, visualization, etc, it is desirable to obtain a set of point-to-point image correspondences between the two images. To achieve this, a hard assignment can be performed in either of two possible directions, from features in image A to features in image B, or vice versa.

For this purpose, two scores are defined from the correlation map, by performing soft-max in the dimensions corresponding to images A and B:

$$s^A_{ijkl} = \frac{\exp(c'_{ijkl})}{\sum_{ab}\exp(c'_{abkl})} \quad \text{and} \quad s^B_{ijkl} = \frac{\exp(c'_{ijkl})}{\sum_{cd}\exp(c'_{ijcd})}. \tag{5.8}$$

Note that the scores are: (i) positive, (ii) normalized using the soft-max function, which makes $\sum_{ab} s^B_{ijab} = 1$. Hence we can interpret them as discrete conditional probability distributions of f^A_{ij}, f^B_{kl} being a match, given the position (i, j) of the match in A or (k, l) in B. If we denote (I, J, K, L) the discrete random variables indicating the position of a match (*a priori* unknown), and (i, j, k, l) the particular position of a match, then:

$$\begin{aligned} &\mathbb{P}\left(K = k, L = l \mid I = i, J = j\right) = s^B_{ijkl}, \quad \text{and} \\ &\mathbb{P}\left(I = i, J = j \mid K = k, L = l\right) = s^A_{ijkl}. \end{aligned} \tag{5.9}$$

Then, the hard-assignment in one direction can be done by just taking the most likely match (the mode of the distribution) as follows:

$$\begin{aligned} &f^A_{ij} \text{ matches } f^B_{kl} \text{ with score } \rho^A_{ij} \text{ iff:} \\ &(k, l) = \underset{cd}{\text{argmax}}\, \mathbb{P}\left(K{=}c, L{=}d \,|\, I{=}i, J{=}j\right) = \underset{cd}{\text{argmax}}\, s^B_{ijcd}, \\ &\text{with} \quad \rho^A_{ij} := s_{ijkl}. \end{aligned} \tag{5.10}$$

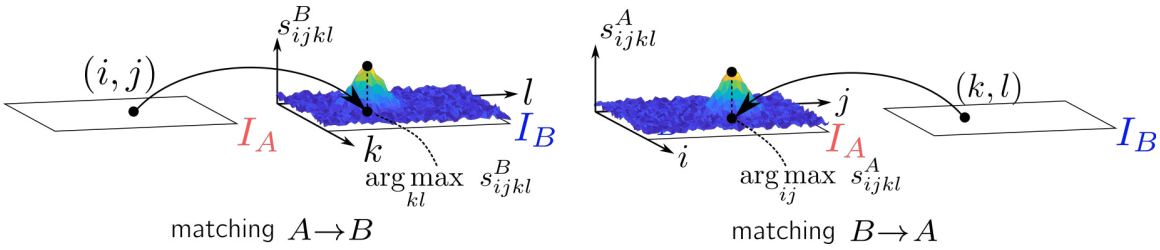


Figure 5-2: **Extracting correspondences from the correlation map**. We illustrate the process of extracting correspondences in both directions, $A \rightarrow B$ and $B \rightarrow A$.

The matches of f^B_{kl} over f^A are obtained analogously. This process is illustrated in Fig. 5-2.

This probabilistic intuition allows us to model the match uncertainty using a probability distribution and will be also useful to motivate the loss used for weakly-supervised training, which will be described next.

5.3.6 Weakly-supervised training

In this section we define the loss function used to train our network. One option is to use a strongly-supervised loss, but this requires dense annotations consisting of all pairs of corresponding points for each training image pair. Obtaining such exhaustive ground-truth is complicated – dense manual annotation is impractical, while sparse annotation followed by an automatic densification technique typically results in imprecise and erroneous training data. Another alternative is to resort to synthetic imagery which would provide point correspondences by construction, but this has the downside of making it harder to generalize to larger appearance variations encountered in real image pairs we wish to handle. Therefore, it is desirable to be able to train directly from pairs of real images, requiring as little annotation as possible.

For this we propose to use a training loss that only requires a weak-level of supervision consisting of annotation on the level of image pairs. These training pairs (I^A, I^B) can be of two types, positive pairs, labelled with $y = +1$, or negative pairs,

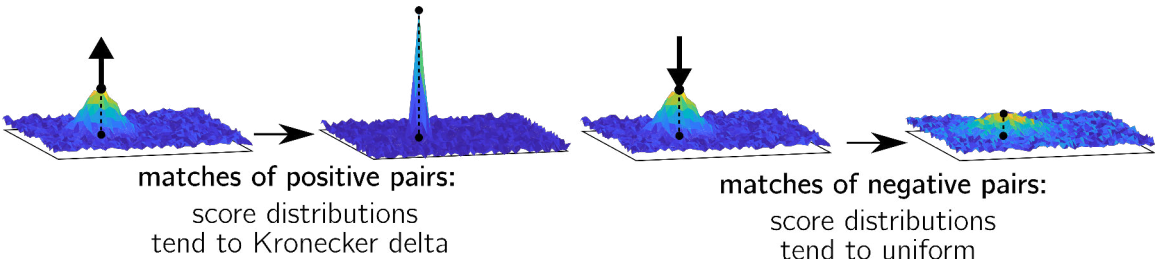


Figure 5-3: **Illustration of the proposed weakly-supervised loss**. For positive pairs, the distribution of match scores is forced towards a Kronecker delta distribution, while for negative pairs it is forced towards a uniform distribution.

labelled with $y = -1$. Then, the following loss function is proposed:

$$\mathcal{L}(I^A, I^B) = -y\left(\bar{\rho}^A + \bar{\rho}^B\right), \tag{5.11}$$

where $\bar{\rho}^A$ and $\bar{\rho}^B$ are the mean matching scores over all hard assigned matches as per (5.10) of a given image pair (I^A, I^B) in both matching directions.

Note that the minimization of this loss maximizes the scores of positive and minimizes the scores of negative image pairs, respectively. As explained in 5.3.5, the hard-assigned matches correspond to the modes of the distributions of (5.9). Therefore, maximizing the score forces the distribution towards a Kronecker delta distribution, having the desirable effect of producing well-identified matches in positive image pairs. Similarly, minimizing the score forces the distribution towards the uniform one, weakening the matches in the negative image pairs. Note that while the only scores that directly contribute to the loss are the ones coming from hard-assigned matches, all matching scores affect the loss because of the normalization in (5.8). Therefore, all matching scores will be updated at each training step. An illustration of the proposed weakly-supervised loss is presented in Fig. 5-3.

5.3.7 Feature relocalization

The localization precision of the extracted features f^I_{ij} depends on the spatial resolution $h \times w$ of the dense feature map f^I. For some tasks, such as pose estimation, precisely localized features are needed. However, in some cases, given hardware constraints,

one cannot increase the spatial resolution $h \times w$ to obtain the required precision, as increasing h and w by a factor of two results in a sixteen-fold increase in the memory consumption and computation time of the whole NCNet model. Therefore, we devise a method to increase the localization precision, with a less severe impact on the memory consumption and computation time.

In this approach, the correlation map c from (5.1) is computed with higher resolution features $2h \times 2w$ leading to a $2h \times 2w \times 2h \times 2w$ correlation map. This correlation map c is then downsampled to resolution $h \times w \times h \times w$ before further processing by the neighbourhood consensus network. Note that by doing this the memory requirements of the correlation tensor are still increased by a factor of 16, but the memory requirements of the 4D convolutional network are kept constant. The downsampling is performed by a 4-D max-pooling operation, with the kernel of size 2:

$$c'_{abcd} = \max_{i \in [2a,2a+1], j \in [2b,2b+1], k \in [2c,2c+1], l \in [2d,2d+1]} c_{ijkl}. \tag{5.12}$$

The downsampled correlation map c' is then processed and used to compute the final matches, which are localized with a precision given by the downsampled resolution $h \times w$. However, one can *re-localize* these features, and reduce the localization error, by simply using the positions of the features that yielded the locally maximal correlation value in the 4-D max-pooling operation given by (5.12). In other words, for a match (f^A_{ab}, f^B_{cd}), the final *re-localized* feature positions (a', b') and (c', d') are computed by:

$$a', b', c', d' = \operatorname*{argmax}_{i \in [2a,2a+1], j \in [2b,2b+1], k \in [2c,2c+1], l \in [2d,2d+1]} c_{ijkl}. \tag{5.13}$$

Note that a similar approach was used in [Badrinarayanan et al., 2017] for upsampling feature maps for the task of semantic segmentation, and in [Widya et al., 2018] for feature localization.

5.4 Experimental results

The proposed approach was evaluated on both category and instance-level matching problems. The same approach is used to obtain reliable matches for both types of matching problems, which are then used to solve the different task proposed by each particular benchmark.

5.4.1 Category-level matching

The proposed method was evaluated on the task of category-level matching, where, given two images containing different instances from the same category (*e.g.* two different cat images) the goal is to match or align the similar semantic parts. Three different standard benchmarks were used and evaluated using their respective metrics. These benchmarks will be presented next.

Proposal Flow. The Proposal Flow benchmark was used for evaluating the task of semantic keypoint transfer, where given annotated keypoints in the source image the task is to determine their positions in the target image. Both PF-Pascal and PF-Willow variants of the PF dataset [Ham et al., 2017] were used, which respectively contain 1251 and 900 semantically related image pairs annotated with sparse keypoints. The performance is measured using the percentage of correct keypoints (PCK), that is, the number of correctly matched annotated keypoints within a tolerance threshold of the ground-truth position. In both cases, the evaluation protocol of [Han et al., 2017; Rocco et al., 2018a] is followed, where the PCK is computed using normalized keypoint coordinates $(\hat{x}, \hat{y}) = (x/w, y/h)$ with (h, w) being the image resolution and the normalized distance threshold $\hat{L} = \alpha$. Note that [Ham et al., 2017; Rocco et al., 2017] used slightly different definitions of PCK, so in order to make a fair comparison, for these methods we report the results from [Han et al., 2017; Kim et al., 2018a] which re-evaluated them using the presented evaluation procedure.

Caltech-101. The Caltech-101 [Fei-Fei et al., 2006] dataset was used for evaluating the task of label transfer, which consists of transferring the semantic segmentation

labels of the source image onto the target image. The same evaluation data and procedure as in Chapter 3 and Ham et al. [2017] was used, which includes 1515 evaluation pairs. The alignment accuracy is quantitatively measured using the label transfer accuracy (LT-ACC), which measures the alignment correctness of both foreground and background labels; and the Jaccard similarity coefficient (IoU), which only measures the alignment correctness of the foreground object. The previously employed localization error metric (LOC-ERR) is not considered here as it was shown to be unrepresentative of the alignment quality. For the mathematical definitions of these metrics, please refer to Chapter 3. For qualitative evaluation, the output aligned images are presented, which provides better qualitative assessment of the alignment than visualizing the transferred segmentation masks.

TSS. Finally, we report results on the TSS dataset [Taniai et al., 2016] which consists of 400 semantically related image pairs which are subdivided into three subgroups: FG3D, JODS and PASCAL. By employing a semi-automated method requiring human intervention, dense ground-truth flow maps were computed for each image pair, which enables dense evaluation of the alignment, in contrast to sparse keypoints or segmentation masks used in the previous two cases. The metric employed to assess the dense alignment is also the percentage of correct keypoints (PCK) – but evaluated densely and not sparsely – with distance threshold $L = \alpha \max(h, w)$, where (h, w) is the target image resolution.

Training. From the three category-level benchmarks, only the PF-Pascal provides a training split [Han et al., 2017], which divides the data into approximately 700 pairs for training, 300 for validation and 300 for testing. In order to train the network in a weakly-supervised manner using the proposed loss (5.11), the 700 training pairs of PF-Pascal are used as positive training pairs, and negative pairs are generated by randomly pairing images of different categories, such as a car with a dog image. The same model trained on the PF-Pascal training split was used for evaluation on all the category-level benchmarks.

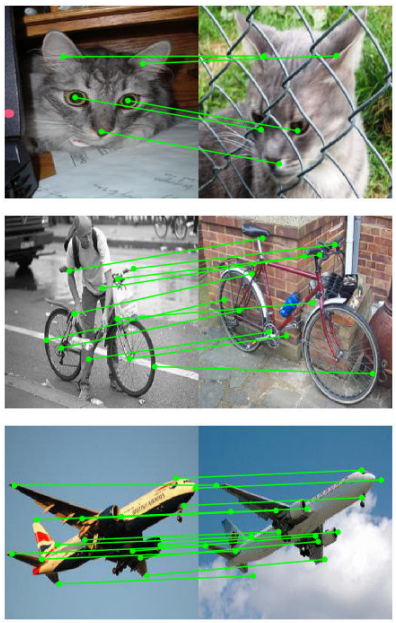
	PF-P	PF-W	Caltech		TSS			
Method	PCK	PCK	IOU	LT-ACC	FG3D	JODS	PASCAL	avg.
HOG+PF-LOM [Ham et al., 2017]	62.5	56.8	0.50	0.78	0.786	0.653	0.531	0.657
SCNet-AG+ [Han et al., 2017]	72.2	70.4	0.51	0.79	0.776	0.608	0.474	0.619
CNNGeo [Rocco et al., 2017]	71.9	81.1	0.61	0.83	0.901	0.764	0.563	0.743
WeakAlign [Rocco et al., 2018a]	75.8	**84.3**	0.63	0.85	0.903	0.764	0.565	0.744
RTN [Kim et al., 2018a]	75.9	71.9	**0.64**	**0.87**	0.901	0.782	**0.633**	0.772
NCNet ($r = 100$, FR)	**78.9**	**84.3**	0.62	0.85	**0.945**	**0.814**	0.571	**0.777**

Table 5.1: **Results for semantic matching on different datasets.** We evaluate on the tasks of keypoint transfer (PF-Pascal/Willow), label transfer (Caltech) and dense alignment (TSS).

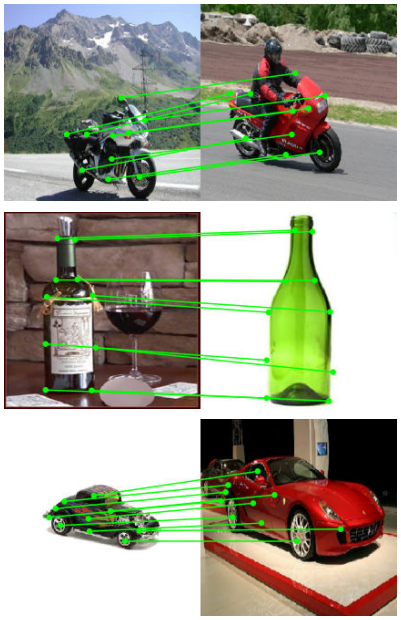
Results. Quantitative results are presented in Table 5.1. The proposed neighbourhood consensus network (NCNet) obtains state-of-the-art results in several of the evaluated benchmarks. Qualitative examples of the semantic keypoint transfer on the PF dataset are shown in Fig. 5-4. Additional qualitative examples of dense alignment on the Caltech-101 and TSS datasets are presented in Fig. 5-5. These qualitative results demonstrate how our approach can correctly match semantic object parts in challenging situations with large changes of appearance and non-rigid geometric deformations. Note that the dense alignments in Fig. 5-5 are obtained directly by bilinear interpolation of the matches outputted by NCNet, not requiring the use of any global geometric model or regularization technique, in contrast with other methods.

5.4.2 Instance-level matching

Next we show that our method is also suitable for instance level matching and evaluate it on two different benchmarks, (i) HPatches [Balntas et al., 2017], consisting mostly of pictures of outdoor planar scenes, paintings or printed photographs and, (ii) InLoc [Taira et al., 2018], consisting in indoor images taken at different times. While in both benchmarks the image pairs contain strong variations of illumination conditions and viewpoint, the InLoc dataset is particularly challenging as indoor spaces are often self-similar and contain large textureless areas. Furthermore, while the HPatches dataset allows for a direct evaluation of the matching accuracy, the InLoc dataset evaluates the matching task as a module in an indoor visual localization pipeline,

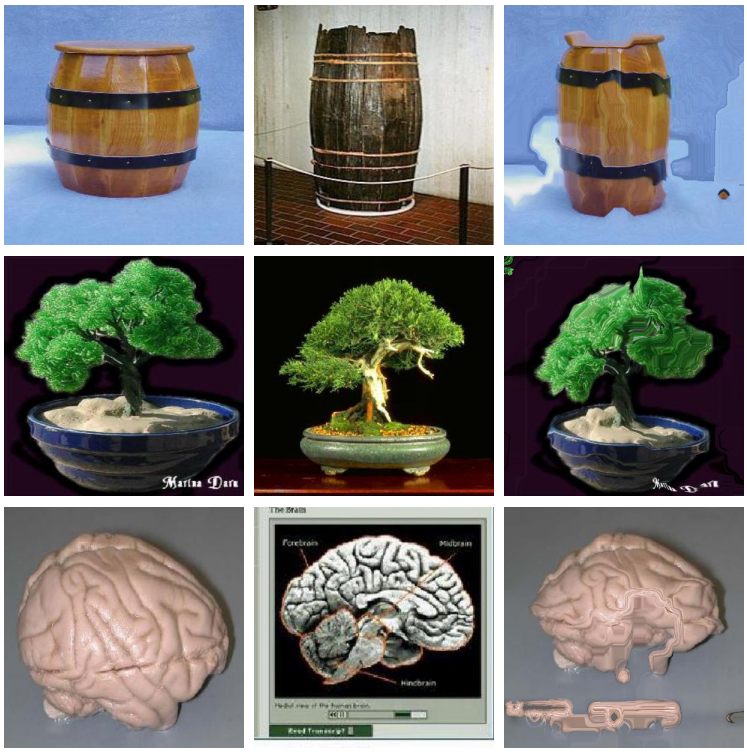


(a) PF-Pascal



(b) PF-Willow

Figure 5-4: **Semantic keypoint transfer.** The annotated (ground truth) keypoints in the left image are automatically transferred to the right image using the dense correspondences between the two images obtained from our NCNet.



(a) Caltech-101

(b) TSS

Figure 5-5: **Dense semantic alignment.** The first two columns show the source and target images, respectively. The right-most column shows the result of transforming the source image by bilinear interpolation using the matches obtained by NCNet such that the result is aligned to the target image. Note that no global geometric model is used for the warping.

where the goal is to estimate an accurate 6DoF camera pose of a query photograph given a large-scale 3D model of a building.

HPatches. We employ the HPatches benchmark [Balntas et al., 2017] to directly evaluate the matching accuracy in the instance-level matching case. The dataset contains 116 sequences, each belonging to a different planar scene and each containing 6 images which are used to form 5 image pairs. These sequences are divided into the *illumination* (57 sequences) and *viewpoint* (59 sequences) subsets, which only contain changes along these factors of variation. The dataset provides a ground-truth homography transformation for each image pair, which is used to assess the accuracy of the extracted matches. Two different measures are used. The first one is the mean matching accuracy (MMA), following the standard evaluation protocol for this dataset [Dusmanu et al., 2019], inspired by [Mikolajczyk and Schmid, 2005]:

$$\mathrm{MMA}\Big(\{(p_i^A, p_i^B)\}_{i=1}^N; t\Big) = \frac{\sum_{i=1}^N \mathbb{1}_{>0}\Big(t - \|\mathcal{T}(p_i^A; H_{GT}) - p_i^B\|_2\Big)}{N}, \tag{5.14}$$

where $\{(p_i^A, p_i^B)\}_{i=1}^N$ is the set of matches between points p_i^A and p_i^B in images A and B respectively, $\mathcal{T}(\cdot, H_{GT})$ is the transformation with the ground-truth homography H_{GT}, $\mathbb{1}_{>0}$ is the indicator function for positive numbers and t is the distance threshold parameter. Similarly to PCK, MMA measures the proportion of matches that are correct up to a certain tolerance threshold t, but contrary to PCK, t is defined as an absolute number of pixels in the original image resolution. The MMA directly evaluates the matching accuracy. The second evaluation metric assesses whether the matches can be used to accurately estimate the homography transformation between each image pair. For this, the obtained matches are used to estimate the homography matrix $\hat{H}$ by running a modern RANSAC variant [Chum et al., 2003, 2005]. The average transfer error (TE) of the estimated homography $\hat{H}$ with respect to the ground-truth homography H_{GT} is computed as:

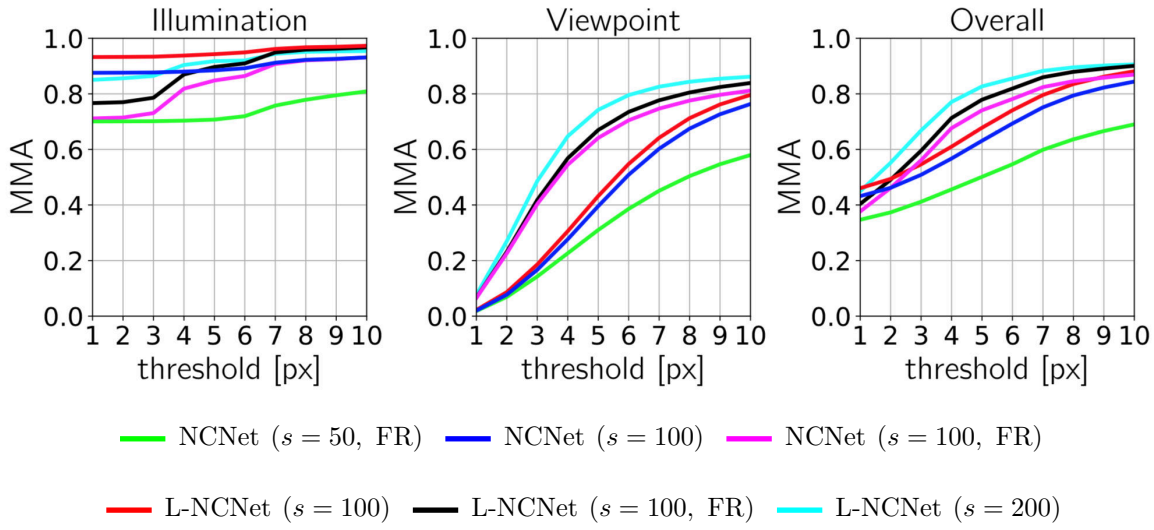
$$\mathrm{TE}\Big(\hat{H}; t\Big) = \frac{\sum_{p^A \in \Omega} \|\mathcal{T}(p^A; H_{GT}) - \mathcal{T}(p^A; \hat{H})\|_2}{|\Omega_A|}, \tag{5.15}$$

where Ω_A is the set of all pixels coordinates over image A, $|\Omega_A|$ the number of pixels of image A, $\mathcal{T}(\cdot, H_{GT})$ is the transformation with the ground-truth homography H_{GT} and $\mathcal{T}(\cdot, \hat{H})$ is the transformation with the estimated homography $\hat{H}$. We use the transfer error to evaluate the quality of $\hat{H}$ as it represents a meaningful geometric distance which is measured in pixels and is invariant to the homography parametrization, which is not the case if the error is computed directly on the homography matrix entries.

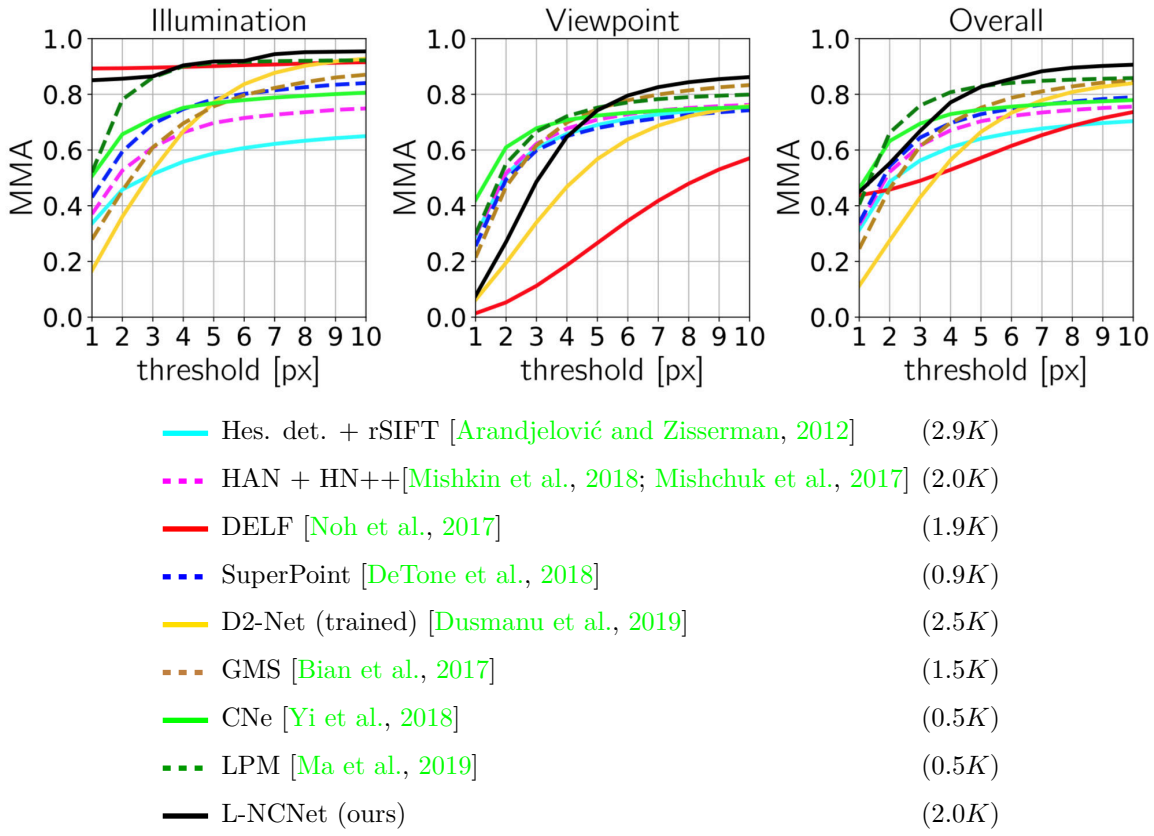
InLoc. We use the InLoc dataset [Taira et al., 2018], which consists of 10K database images (perspective cutouts) extracted from 227 RGBD panoramas, and an additional set of 356 query images captured with a smart-phone camera at a different time (several months later) from the database images. Here, the goal is to estimate an accurate 6DoF camera pose of a query photograph within a large-scale 3D model of a building. We follow the same evaluation protocol as in [Taira et al., 2018] and report the percentage of correctly localized queries at a given camera position error threshold.

Training. As both the HPatches and the InLoc were designed for evaluation and do not provide a training set, we collected an Indoor Venues Dataset (IVD) [Rocco et al., 2018c], consisting of user-uploaded photos, captured at public places such as restaurants, cafes, museums or cathedrals, by crawling Google Maps. It features not only viewpoint and illumination changes, such as the variations present in the HPatches dataset, but also scene modifications due to the passage of time as in the InLoc dataset. The IVD dataset contains 3861 positive image pairs from 89 different venues in 6 different cities, split into *train*: 3481 pairs (80 places) and *validation*: 380 pairs (from the remaining 9 places). The same model trained on IVD was used for evaluation on both the HPatches and InLoc benchmarks.

Results. We use the trained NCNet model to find correspondences in each pair of the HPatches dataset and evaluate the accuracy of these correspondences. The results using the mean matching accuracy (MMA) metric from (5.14) are presented in Fig. 5-6. In Fig. 5-6a, the results of several variants of the proposed method are presented.



(a) *Variants of the proposed method.* We present results when matching using different feature map sizes (s is the size of the feature map along the longest edge), using both NCNet and the lighter version L-NCNet with or without the feature relocalization operation (FR).



(b) *Comparison with state-of-the-art methods.* The proposed L-NCNet (with a feature map size $s = 200$) method obtains the best overall results for threshold values above 5px and the best viewpoint results for thresholds above 5.5px.

Figure 5-6: **HPatches benchmark results.** We report the Mean Matching Accuracy (MMA) as a function of the tolerance threshold for the *illumination* and *viewpoint* subsets, as well as the *overall* results.

These variations include: (i) different sizes of the feature maps (s=50, 100 or 200 along the longest edge), which correspond to running the method with different input image resolutions; (ii) the use of a feature relocalization operation described in Sec. 5.3.7; and (iii) either employing the symmetric NCNet or the lightweight L-NCNet model, which allows for evaluation with larger feature map sizes. Increasing the feature map size improves the localization precision of the matches, increasing the matching accuracy. However the vanilla NCNet cannot handle the larger feature map sizes due to GPU memory constraints. Using the feature relocalization method enables the NCNet to improve the localization precision, improving the MMA over the variants using the same feature map sizes and no relocalization. The best results are obtained with the lighter L-NCNet using the largest feature size of $s = 200$.

In Fig. 5-6b, the best performing variant of NCNet is compared against other state-of-the-art methods on the HPatches benchmark. Our method obtains the best results for viewpoint changes for the larger thresholds. For smaller thresholds, our method suffers from a localization precision that is limited by the resolution of the features used for matching. This is a common issue for all methods that rely on CNN descriptors, contrary to handcrafted descriptors that can be run on much higher resolutions. Note that, while some methods present better accuracy at small threshold values in the viewpoint subset, they tend to perform worse than our method in the illumination subset. Overall, our method obtains the best results for threshold values above 5px.

Next, we evaluate the quality of the homographies estimated with RANSAC using the transfer error (5.15) on the HPatches dataset. An image pair is counted as correctly aligned if the transfer error for that pair is smaller than a 5px threshold. In Tab. 5.2 we present the number of pairs that each method is able to correctly align, from both the illumination and viewpoint subsets, and overall. For the correctly aligned pairs, we also show the average number of inliers, and the effective transfer error for these pairs. NCNet can align more pairs overall than any other method, while obtaining the smallest average transfer error overall.

	Illumination (260 pairs)			Viewpoint (280 pairs)			Overall (540 pairs)		
Method	# corr.	# inl.	TE [px]	# corr.	# inl.	TE [px]	# corr.	# inl.	TE [px]
NCNet	256	**9124**	**0.34**	**268**	**8227**	1.35	**524**	**8665**	**0.86**
SuperPoint	**257**	574	0.96	264	914	**1.16**	521	746	1.06
GMS	224	2478	1.27	235	2782	1.34	459	2634	1.31
CNe	254	486	1.05	252	438	1.26	506	462	1.16
LPM	222	199	1.15	223	945	1.24	445	572	1.20

Table 5.2: **Homography estimation on HPatches** The number of correctly aligned pairs (within a 5px average transfer error threshold) for each method and benchmark subset (illumination, viewpoint or overall) is computed. For the correct pairs, we also present the average number of inliers and transfer error (TE). NCNet obtains the largest number of correctly aligned pairs overall (524 out of 540) with the smallest average transfer error (0.86px).

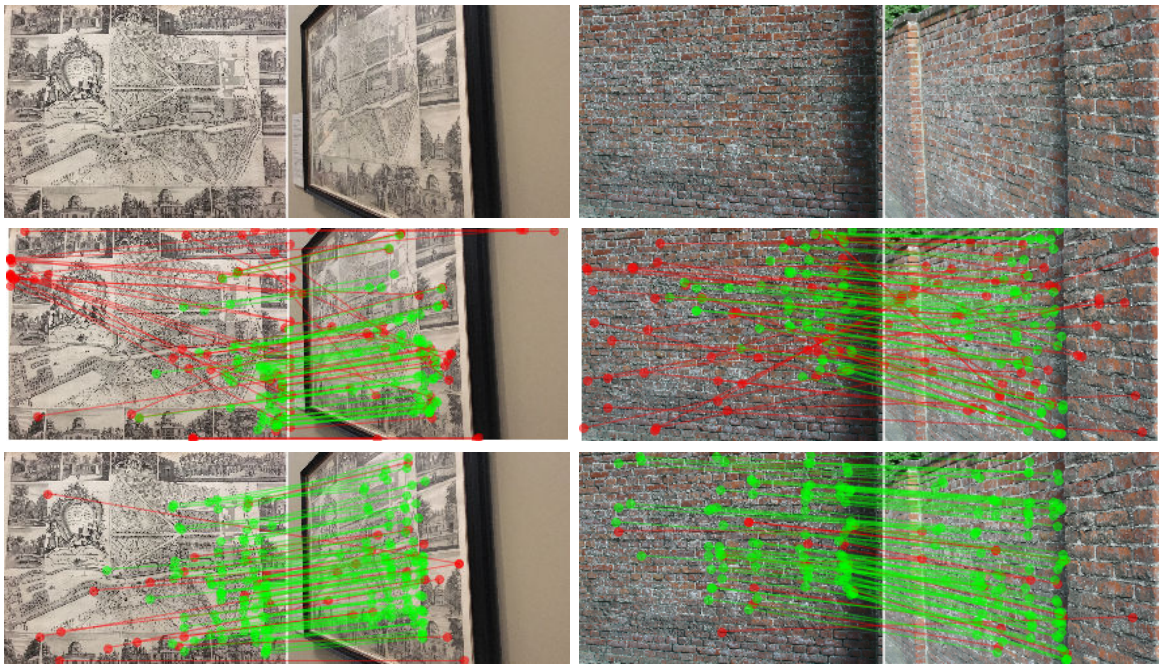


Figure 5-7: **Correspondences on HPatches images.** The top row shows the image pair; the middle row shows the matches obtained directly from the correlation map c (before NCNet filtering); the bottom row shows the matches obtained by the proposed method (after NCNet filtering). Correspondences have been coloured as inliers (green) and outliers (red) w.r.t. the ground-truth homography using a threshold of 5px. Each image shows 100 randomly sampled matches from the top 2000 matches. The proposed NCNet method tends to obtain a larger fraction of correct matches which span a larger portion of the image with respect to the raw matches before NCNet filtering.

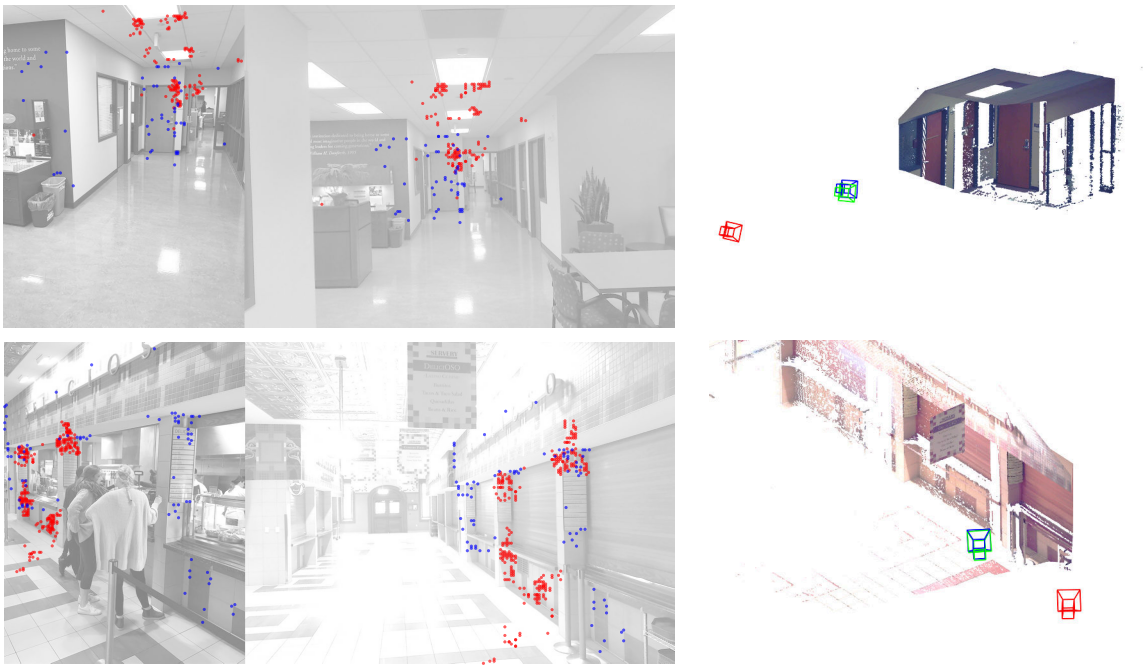
Finally, in Fig. 5-7 we present qualitative examples which illustrate the effect of NCNet in filtering the tentative correspondences from the raw correlation tensor. We plot 100 randomly sampled matches from the top 2000 matches obtained directly from the correlation map, and after filtering with NCNet. Results show that NCNet tends to increase the fraction of correct matches.

For evaluation on the InLoc benchmark, we plug-in our trainable neighbourhood consensus network (NCNet) as a correspondence module into the InLoc indoor localization pipeline [Taira et al., 2018]. This pipeline consists of the following steps: i. retrieval, ii. re-ranking, iii. pose estimation for shortlisted images, and iv. dense pose verification by view synthesis. We integrate NCNet as an intermediate step between the re-ranking (ii) and pose estimation steps (iii) of the InLoc pipeline. In the combined approaches DensePE+NCNet and InLoc+NCNet, the matches generated by NCNet are used to compute the poses in step (iii). DensePE+NCNet contains steps (i-iii) while InLoc+NCNet also contains the dense pose verification step (iv).

In order to evaluate the contribution of NCNet separately from that of the feature extractor network, two additional experiments are performed where NCNet is replaced with hard mutual nearest neighbours matching (MNN), using the same base CNN network (ResNet-101). Results are summarized in Table 5.3 and clearly demonstrate benefits of our approach (DensePE+NCNet) compared to both sparse keypoint (DoG+SIFT) matching (SparsePE) and the CNN feature matching used in [Taira et al., 2018] (DensePE). When inserted into the entire localization pipeline, our approach (InLoc + NCNet) obtains state-of-the-art results on the indoor localization benchmark. For these experiments an input resolution of $r = 3200$ pixels along the longest image edge together with the feature relocalization operation from Sec. 5.3.7 were used. Qualitative results of NCNet in challenging indoor scenes with repetitive structures and texture-less areas are presented in Fig. 5-8.

	Distance (m)			
Method	0.25	0.50	1.00	2.00
SparsePE [Taira et al., 2018]	21.3	30.7	42.6	48.3
DensePE [Taira et al., 2018]	35.3	47.4	57.1	61.1
DensePE+MNN	31.9	50.5	62.0	64.7
DensePE+**NCNet**	37.1	53.5	62.9	66.3
InLoc[Taira et al., 2018]	38.9	56.5	69.9	74.2
InLoc+MNN	37.1	60.2	72.0	76.3
InLoc+**NCNet**	**44.1**	**63.8**	**76.0**	**78.4**

Table 5.3: **InLoc benchmark results.** We show the rate (%) of correctly localized queries within a given distance (m) and 10° angular error.



(a) (b) (c)

Figure 5-8: **Correspondences and poses on InLoc.** Each row shows (a-b) the correspondences used for pose estimation in the case of the proposed NCNet method (blue) against those of the baseline InLoc method (red); and (c) the resulting obtained poses for the proposed NCNet (blue) and InLoc baseline (red) compared to the ground-truth pose (green). In both cases the InLoc baseline produces many mismatches due to repetitive structures (ceiling lamps in the top, and columns in the bottom example) that result in a large pose error. On the other hand, NCNet obtains mostly correct matches resulting in a small pose error.

5.4.3 Ablation studies

In this section we assess the relevance of two different components in the proposed NCNet formulation (5.6), namely the symmetric 4D convolutional network S that implements neighbourhood consensus, and the soft mutual nearest neighbour filtering operation M. For this, we train different variants of the proposed method and evaluate their performance on the PF-Pascal benchmark. No finetuning of the feature extraction network is performed in this ablation. Results are presented in Table 5.4.

The top section of the table shows the performance of two networks containing only the isolated symmetric neighbourhood-consensus S and soft mutual nearest neighbour filtering M components. As it can be observed, the isolated components do not perform well in the task of keypoint transfer. The middle section of the table shows the combinations of these two components. Note that applying the soft mutual nearest neighbour filtering M first and then symmetric neighbourhood consensus module S produces much better results than doing so in the opposite order. The bottom section of the table shows the full proposed model including two stages of soft mutual nearest neighbour filtering M, using both the lightweight L-NCNet and the symmetric NCNet versions.

5.4.4 Implementation details

Model details. The model was implemented in PyTorch [Paszke et al., 2017], and a ResNet-101 network [He et al., 2016] initialized on ImageNet was used for feature extraction (up to the `conv4_23` layer). Different architectures of the neighbourhood consensus network $N(\cdot)$ are used for category- and instance-level matching, as these two problems present different challenges. For category-level matching, a more complex network is used in order to capture the strong appearance differences in these matching problems. For instance-level matching, a simpler network is used, allowing to process the images in higher resolution and obtaining more precisely localized matches, which is required for tasks such as pose estimation. In all cases, the input and output tensors have a single channel, and the intermediate results have 16 channels. For

Method	PF-Pascal PCK ($\alpha = 0.1$)
$S(c)$	12.5
$M(c)$	44.0
$(M \circ S)(c)$	13.6
$(S \circ M)(c)$	77.2
$(M \circ N \circ M)(c)$ (**L-NCNet**)	74.3
$(M \circ S \circ M)(c)$ (**NCNet**)	78.0

Table 5.4: **Ablation studies on the PF-Pascal dataset.** We evaluate different model configurations for filtering the correlation map scores c using the neighbourhood consensus (S or the lightweight N) and soft mutual nearest neighbour filtering (M) modules.

category-level matching, $N(\cdot)$ contains three layers of $5 \times 5 \times 5 \times 5$ filters, resulting in 180K trainable parameters. In the case of instance-level matching, $N(\cdot)$ has two layers of $3 \times 3 \times 3 \times 3$ filters, resulting in 2.6K trainable parameters. Both models are trained using dense feature maps f^A and f^B of size 25×25. For evaluation, the category-level matching model also uses 25×25 dense feature maps, processes an image pair in 0.5s and requires 240MB of memory. For instance-level matching, evaluation is performed using larger feature maps, such as 100×75, in order to obtain a higher localization precision which is required for instance-level matching. In this case, the execution takes 9.3s and requires 5700MB of memory.

Training details. The model is initially trained for 5 epochs using Adam optimizer [Kingma and Ba, 2015], with a learning rate of 5×10^{-4} and keeping the feature extraction layer weights fixed. We used a batch size of 16 and the training takes 9 hours on a standard Tesla T4 GPU. For category level matching, the model is then subsequently finetuned for 5 more epochs, training both the feature extraction and the neighbourhood consensus network, with a learning rate of 1×10^{-5}. In the case of instance level matching, finetuning the feature extraction did not improve the performance.

4D convolutions. As 4D convolutions ($\overset{*}{}_{4D}$) are not currently supported by PyTorch, they were implemented by aggregating the results of multiple 3D convolutions ($\overset{*}{}_{3D}$) over the remaining fourth dimension. Given a 4D input tensor $X \in \mathbb{R}^{h \times w \times d \times t}$ and a 4D weight tensor $W \in \mathbb{R}^{k \times k \times k \times k}$ with k odd (channel and batch dimensions are omitted for simplicity), their convolution can be then computed by:

$$\left(X \underset{4D}{*} W\right)_{i:::} = \sum_{j=0}^{k-1} X_{j':::} \underset{3D}{*} W_{j:::} \text{ with } \; j' = i + j - (k-1)/2 \; , \tag{5.16}$$

and considering 0-indexed tensors and that X takes the value 0 when the index j' is out of range ($j' < 0$ or $j' \geq h$).

In addition, the memory requirements can be reduced by exploiting the fact that the 4D convolutional network N has multiple channels in the hidden layers but single channel input and output. If the correlation tensor can be fitted into the GPU memory, the memory requirements of the forward pass through N can be limited by computing the output tensor $Y = N(X)$ in *chunks* which can then be stacked to obtain the full output tensor. In order to do this, a set of *slices* X_s $s = 1, \ldots, n$ are generated from the input tensor X, and fed to the network progressively. Note that, while the output slices are non-overlapping, the inputs will have an overlap due to the overlapping receptive fields of the output slices. The slicing can be then performed in the following way:

$$\begin{aligned} \{Y_{i:::}\}_{i=a,\ldots,b} = Y_s = N(X_s) \\ \text{with } X_s = \{X_{i:::}\}_{i=a-p,\ldots,b+p} \; , \end{aligned} \tag{5.17}$$

where the input slices X_s are larger than the output slices Y_s by $2p + 1$, which corresponds to the receptive field of the last layer of the network N. In consequence, when using this approach no padding should be performed in the convolutional layers of N.

5.4.5 Limitations

While our method identifies correct matches in many challenging cases, some situations remain difficult. The two typical failure modes include: repetitive patterns combined with large changes in scale, and locally geometrically consistent groups of incorrect matches. Furthermore, the proposed method has quadratic $O(N^2)$ complexity with respect to the number of image pixels (or CNN features) N. This limits the resolution of the images that we are currently able to handle in a 16GB GPU to 1600×1200px (or 3200×2400px if using feature relocalization or slicing), and renders the method relatively slow: the processing time of a 3200×2400px image pair using feature relocalization is ≈ 7 seconds.

5.5 Conclusion

In this chapter we have presented a neighbourhood consensus network — a CNN architecture that learns local patterns of correspondences for image matching without the need for a global geometric model. We have shown the model can be trained effectively from weak supervision and obtains strong results outperforming state-of-the-art on two very different matching tasks. These results open up the possibility for end-to-end learning of other challenging visual correspondence tasks, such as 3D category-level matching [Kanazawa et al., 2018], or visual localization across day/night illumination [Sattler et al., 2018].

Chapter 6

Making neighbourhood consensus networks efficient

In this chapter, we propose the Sparse Neighbourhood Consensus Networks (Sparse-NCNet) which address the main limitations of the NCNet model presented in Chapter 5. In particular, we propose modifications to: (i) reduce memory consumption, (ii) reduce inference time, and (iii) improve the localization of obtained correspondences. Results show that our proposed modifications can reduce the memory footprint and execution time more than 10×, with equivalent matching performance. This is achieved by *sparsifying* the correlation tensor containing tentative matches, subsequently processing it with a 4D CNN using submanifold sparse convolutions. Localization accuracy is significantly improved by processing the input images in higher resolution, which is possible due to the reduced memory footprint, and by a novel two-stage correspondence relocalization module. The proposed Sparse-NCNet method obtains state-of-the-art results on the HPatches Sequences and InLoc visual localization benchmarks, and competitive results in the Aachen Day-Night benchmark.

6.1 Introduction

Finding correspondences between images depicting the same 3D scene is one of the fundamental tasks in computer vision [Julesz, 1962; Marr and Poggio, 1976; Mori

et al., 1973] with applications in 3D reconstruction [Schönberger and Frahm, 2016; Schönberger et al., 2016; Widya et al., 2018], visual localization [Germain et al., 2019; Sattler et al., 2018; Taira et al., 2018] or pose estimation [Gao et al., 2003; Grabner et al., 2018; Persson and Nordberg, 2018]. The predominant approach currently consists of first *detecting* salient local features, by selecting the local extrema of some form of feature selection function, and then *describing* them by some form of feature descriptor [Bay et al., 2006; Lowe, 2004; Rublee et al., 2011]. While hand-crafted features such as Hessian affine detectors [Mikolajczyk and Schmid, 2002] with SIFT descriptors [Lowe, 2004] have obtained impressive performance under strong viewpoint changes and constant illumination [Mikolajczyk et al., 2005], their robustness to illumination changes is limited [Mikolajczyk et al., 2005; Zhou et al., 2016a]. More recently, a variety of trainable keypoint detectors [Laguna et al., 2019; Lenc and Vedaldi, 2016; Mishkin et al., 2018; Verdie et al., 2015] and descriptors [Balntas et al., 2016a,b; Han et al., 2015; Mishchuk et al., 2017; Tian et al., 2017; Zagoruyko and Komodakis, 2015] have been proposed, with the purpose of obtaining increased robustness over hand-crafted methods. While this approach has achieved some success, extreme illumination changes such as day-to-night matching combined with changes in camera viewpoint remain a challenging open problem [Balntas et al., 2019; Dusmanu et al., 2019; Germain et al., 2019]. In particular, all local feature methods, whether hand-crafted or trained, suffer from missing detections under these extreme appearance changes.

In order to overcome this issue, the detection stage can be avoided and, instead, features can be extracted on a dense grid across the image. This approach has been successfully used for both place recognition [Arandjelović et al., 2016; Germain et al., 2019; Noh et al., 2017; Torii et al., 2015] and image matching [Rocco et al., 2018c; Sattler et al., 2018; Widya et al., 2018]. However, extracting features densely comes with additional challenges: it is memory intensive and the localization accuracy of the features is limited by the sampling interval of the grid used for the extraction.

In this chapter we adopt the dense feature extraction approach. In particular, we build on the Neighbourhood Consensus Networks (NCNet) presented in Chapter 5,

(a) Input images (b) Output matches (c) Match confidence

Figure 6-1: **Correspondence estimation with Sparse-NCNet.** Given an input image pair (a), we show the *raw* output correspondences produced by Sparse-NCNet (b) which contain groups of spatially coherent matches. These groups tend to form around highly-confident matches, which are shown in yellow shades (c) (see Sec. 6.5 for a discussion on this behaviour and additional examples).

that allow for jointly trainable feature extraction, matching, and match-filtering to directly output a strong set of (mostly) correct correspondences. Our proposed approach, Sparse-NCNet, seeks to overcome the limitations of the original NCNet formulation, namely: large memory consumption, high execution time and poorly localised correspondences.

Our contributions are the following. First, we propose the efficient Sparse-NCNet model, which is based on a 4D convolutional neural network operating on a *sparse* correlation tensor, which is obtained by storing only the most promising correspondences, instead of the set of all possible correspondences. Sparse-NCNet processes this sparse correlation tensor with submanifold sparse convolutions [Graham et al., 2018] and can obtain equivalent results to NCNet while being several times faster (up to 10×) and requiring much less memory (up to 20×) without decrease in performance compared to the original NCNet model. Second, we propose a two-stage relocalization module to improve the localization accuracy of the correspondences

output by Sparse-NCNet. Finally, we show that the proposed model significantly outperforms state-of-the-art results on the HPatches Sequences [Balntas et al., 2017] benchmark for image matching with challenging viewpoint and illumination changes and the InLoc [Taira et al., 2018] benchmark for indoor localization and camera pose estimation. Furthermore, we show our model obtains competitive results on the Aachen Day-Night benchmark [Sattler et al., 2018], which evaluates day-night feature matching for the task of camera localization. An example of the correspondences produced by our method is presented in Fig. 6-1. Our code and models are available online at http://www.di.ens.fr/willow/research/sparse-ncnet/.

6.2 Related work

Matching with trainable local features. Most recent work in trainable local features has focused on learning more robust keypoint *descriptors* [Balntas et al., 2016a,b; Han et al., 2015; Mishchuk et al., 2017; Tian et al., 2017; Zagoruyko and Komodakis, 2015]. Initially these descriptors were used in conjunction with classic hand-crafted keypoint detectors, such as DoG [Lowe, 2004]. Recently, trainable keypoint *detectors* were also proposed [Laguna et al., 2019; Lenc and Vedaldi, 2016; Mishkin et al., 2018; Verdie et al., 2015], as well as methods providing both *detection and description* [DeTone et al., 2018; Dusmanu et al., 2019; Ono et al., 2018; Revaud et al., 2019; Yi et al., 2016]. From these, some adopt the classic approach of first performing detection on the whole image and then computing descriptors from local image patches, cropped around the detected keypoints [Ono et al., 2018; Yi et al., 2016], while the most recent methods compute a joint representation from which both detections and descriptors are computed [DeTone et al., 2018; Dusmanu et al., 2019; Revaud et al., 2019]. In most cases, local features obtained by these methods are independently matched using nearest-neighbour search with the Euclidean distance [Balntas et al., 2016a,b; Mishchuk et al., 2017; Tian et al., 2017], although some works have proposed to learn the distance function as well [Han et al., 2015; Zagoruyko and Komodakis, 2015]. As discussed in the previous section, local features are prone to loss of detections

under extreme lighting changes [Germain et al., 2019]. In order to alleviate this issue, in this chapter we adopt the usage of dense features, which are described next.

Matching with densely extracted features. Motivated by applications in large-scale visual search, others have found that using densely extracted features provides additional robustness to illumination changes compared to local features extracted at detected keypoints, which suffer from low repeatability under strong illumination changes [Torii et al., 2015; Zhao et al., 2013]. This approach was also adopted by later work [Arandjelović et al., 2016; Noh et al., 2017]. Such densely extracted features used for image retrieval are typically computed on a coarse low resolution grid (*e.g.* 40×30). However, such coarse localization of the dense features is not an issue for visual retrieval, as the dense features are not directly matched, but rather aggregated into a single image-level descriptor, which is used for retrieval. Recently, densely extracted features have been also employed directly for 3D computer vision tasks, such as 3D reconstruction [Widya et al., 2018], indoor localization and camera pose estimation [Taira et al., 2018], and outdoor localization with night queries [Germain et al., 2019; Sattler et al., 2018]. In these methods, correspondences are obtained by nearest-neighbour search performed on extracted descriptors, and filtered by the mutual nearest-neighbour criterion [Oron et al., 2017]. In this chapter, we build on the NCNet method presented in Chapter 5, where the match filtering function is learnt from data. Different recent methods for learning to filter matches are discussed next.

Learning to filter incorrect matches. When using both local features extracted at keypoints or densely extracted features, the obtained matches by nearest-neighbour search contain a certain portion of incorrect matches. In the case of local features, a heuristic approach such as Lowe's ratio test [Lowe, 2004] can be used to filter these matches. However the ratio threshold value needs to be manually tuned for each method. To avoid this issue, filtering by mutual nearest neighbours can be used instead [Dusmanu et al., 2019]. Recently, trainable approaches have also been proposed for the task of filtering local feature correspondences [Brachmann and Rother, 2019; Yi

et al., 2018; Sarlin et al., 2019; Zhang et al., 2019]. Yi *et al.* [Yi et al., 2018] propose a neural-network architecture that operates on 4D match coordinates and classifies each correspondence as either correct or incorrect. Brachmann *et al.* [Brachmann and Rother, 2019] propose the Neural-guided RANSAC, which extends the previous method to produce weights instead of classification labels, which are used to guide RANSAC sampling. Zhang *et al.* [Zhang et al., 2019] also extend the work of Yi *et al.* in their proposed Order-Aware Networks, which capture local context by clustering 4D correspondences onto a set of ordered clusters, and global context by processing these clusters with a multi-layer perceptron. Finally, Sarlin *et al.* [Sarlin et al., 2019] describe a graph neural network followed by an optimisation procedure to estimate correspondences between two sets of local features. These methods were specifically designed for filtering local features extracted at keypoint locations and not features extracted on a dense grid. Furthermore, these methods are focused only on learning match filtering, and are decoupled from the problem of learning how to detect and describe the local features.

In this chapter we build on the NCNet method (Chapter 5) for filtering incorrect matches, which was designed for dense features. Furthermore, contrary to the above described methods, our approach performs feature extraction, matching and match filtering in a single pipeline.

Improved feature localization. Recent methods for local feature detection and description which use a joint representation [DeTone et al., 2018; Dusmanu et al., 2019] as well as methods for dense feature extraction [Rocco et al., 2018c; Widya et al., 2018] suffer from poor feature localization, as the features are extracted on a low-resolution grid. Different approaches have been proposed to deal with this issue. The D2-Net method [Dusmanu et al., 2019] follows the approach used in SIFT [Lowe, 2004] for refining the keypoint positions, which consists of locally fitting a quadratic function to the feature detection function around the feature position and solving for the extrema. The SuperPoint method [DeTone et al., 2018] uses a CNN decoder that produces a one-hot output for each 8×8 pixel cell of the input image (in case a

keypoint is effectively detected in this region), therefore achieving pixel-level accuracy. Others [Widya et al., 2018] use the intermediate higher resolution features from the CNN to improve the feature localization, by assigning to each pooled feature the position of the feature with highest L2 norm from the preceding higher resolution map (and which participated in the pooling). This process can be repeated up to the input image resolution.

The relocalization approach of NCNet (Chapter 5) is based on a max-argmax operation on the 4D correlation tensor of exhaustive feature matches. This approach can only increase the resolution of the output matches by a factor of 2. In contrast, we describe a new two-stage relocalization module that builds on the approach used in NCNet, by combining a hard relocalization stage that has similar effects to NCNet's max-argmax operation, with a soft-relocalization stage that obtains sub-feature-grid accuracy via interpolation.

Sparse Convolutional Neural Networks were recently introduced [Graham, 2015, 2014] for the purpose of processing sparse 2D data, such as handwritten characters [Graham, 2014]; 3D data, such as 3D point-clouds [Graham, 2015]; or even 4D data, such as temporal sequences of 3D point clouds [Choy et al., 2019a]. These models have shown great success in 3D point-cloud processing tasks such as semantic segmentation [Choy et al., 2019a; Graham et al., 2018] and point-cloud registration [Choy et al., 2019b; Gojcic et al., 2020]. In this chapter, we use networks with *submanifold sparse convolutions* [Graham et al., 2018] for the task of filtering correspondences between images, which can be represented as a sparse set of points in a 4D space of image coordinates. In submanifold sparse convolutions, the active sites remain constant between the input and output of each convolutional layer. As a result, the sparsity level remains fixed and does not change after each convolution operation. To the best of our knowledge this is the first time these models are applied to the task of match filtering.

6.3 Sparse Neighbourhood Consensus Networks

In this section we detail the proposed Sparse Neighbourhood Consensus Networks. We start with a brief review of Neighbourhood Consensus Networks (Chapter 5) identifying their main limitations. Next, we describe our approach which overcomes these limitations.

6.3.1 Review: Neighbourhood Consensus Networks

The Neighbourhood Consensus Network is a method for feature extraction, matching and match filtering. Contrary to most methods, which operate on local features, NCNet operates on dense feature maps $(f^A, f^B) \in \mathbb{R}^{h \times w \times c}$ with c channels, which are extracted over a regular grid of $h \times w$ spatial resolution. These are obtained from the input image pair $(I_A, I_B) \in \mathbb{R}^{H \times W \times 3}$ by a fully convolutional feature extraction network. The resolution $h \times w$ of the extracted dense features is typically 1/8 or 1/16 of the input image resolution $H \times W$, depending on the particular feature extraction network architecture used.

Next, the exhaustive set of all possible matches between the dense feature maps f^A and f^B is computed and stored in a 4D correlation tensor $c^{AB} \in \mathbb{R}^{h \times w \times h \times w}$. Finally, the correspondences in c^{AB} are filtered by a 4D CNN. This network can detect coherent spatial matching patterns and propagate information from the most certain matches to their neighbours, robustly identifying the correct correspondences. This last filtering step is inspired by the neighbourhood consensus procedure [Bian et al., 2017; Schaffalitzky and Zisserman, 2002a; Schmid and Mohr, 1997; Sivic and Zisserman, 2003; Zhang et al., 1995], where a particular match is verified by analysing the existence of other coherent matches in its spatial neighbourhood in both images.

Despite its promising results, the original formulation of Neighbourhood Consensus Networks has three main drawbacks that limit its practical application: it is (i) memory intensive, (ii) slow, and (iii) matches are poorly localised. These points are discussed in detail next.

High memory requirements. The high memory requirements are due to the computation of the correlation tensor $c^{AB} \in \mathbb{R}^{h \times w \times h \times w}$ which stores all matches between the densely extracted image features $(f^A, f^B) \in \mathbb{R}^{h \times w \times c}$. Note that the number of elements in the correlation tensor $(h \times w \times h \times w)$ grows quadratically with respect to the number of features $(h \times w)$ of the dense feature maps (f^A, f^B), therefore limiting the ability to increase the feature resolution. For instance, for dense feature maps of resolution 200×150, the correlation tensor would require by itself 3.4GB of GPU memory in the standard 32-bit float precision. Furthermore, processing this correlation tensor using the subsequent 4D CNN would require more than 50GB of GPU memory, which is much more than what is currently available on most standard GPUs. While 16-bit half-float precision could be used to halve these memory requirements, they would still be prohibitively large.

Long processing time. In addition, Neighbourhood Consensus Networks are slow as the full dense correlation tensor must be processed. For instance, processing the $100 \times 75 \times 100 \times 75$ correlation tensor containing matches between a pair of dense feature maps of 100×75 resolution takes approximately 10 seconds on a standard Tesla T4 GPU.

Poor match localization. Finally, the high-memory requirements limit the maximum feature map resolution that can be processed, which in turn limits the localization accuracy of the estimated correspondences. For instance, for a pair images with 1600×1200px resolution, where correspondences are computed using a dense feature map with a resolution of 100×75, the output correspondences are localised within an error of 8 pixels. This can be problematic if correspondences are used for tasks such as pose estimation, where small errors in the localization of correspondences in image-space can yield high camera pose errors in 3D space.

In this chapter, we devise strategies to overcome the limitations of the original NCNet method, while keeping its main advantages, such as the usage of dense feature maps which avoids the issue of missing detections, and the processing of multiple

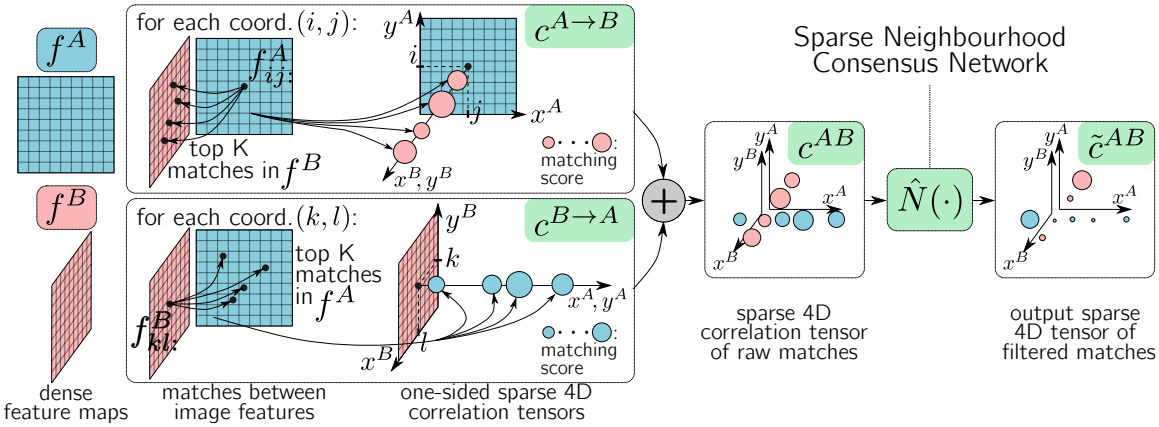


Figure 6-2: **Overview of Sparse-NCNet.** From the dense feature maps f^A and f^B, their top K matches are computed and stored in the one-sided sparse 4D correlation tensors $c^{A\to B}$ and $c^{B\to A}$, which are later combined to obtain the symmetric sparse correlation tensor c^{AB}. The raw matching score values in c^{AB} are processed by the 4D Sparse-NCNet $\hat{N}(\cdot)$ producing the output tensor $\tilde{c}^{AB}$ of filtered matching scores.

matching hypotheses to avoid early matching errors. Our efficient Sparse-NCNet approach is described next.

6.3.2 Sparse-NCNet: Efficient Neighbourhood Consensus Networks

In this section, we describe the Sparse-NCNet approach in detail. An overview is presented in Fig. 6-2. Similar to NCNet, the first stage of our proposed method consists in dense feature extraction. Given a pair of RGB input images $(I^A, I^B) \in \mathbb{R}^{H\times W\times 3}$, $L2$-normalized dense features $(f^A, f^B) \in \mathbb{R}^{h\times w\times c}$ are extracted via a fully convolutional network $F(\cdot)$:

$$f^A = F(I^A), f^B = F(I^B). \tag{6.1}$$

Then, these dense features are matched and stored into a *sparse correlation tensor*. Contrary to the original NCNet formulation, where *all* the pairwise matches between the dense features are stored and processed, we propose to keep *only the top K matches* for a given feature, measured by the cosine similarity. In detail, each feature $f^A_{ij:}$ from image A at position (i,j) is matched with its K *nearest-neighbours* in f^B, and vice

versa. The one-sided sparse correlation tensor, matching from image A to image B ($A \rightarrow B$) is then described as:

$$c_{ijkl}^{A \rightarrow B} = \begin{cases} \langle f_{ij:}^{A}, f_{kl:}^{B} \rangle & \text{if } f_{kl:}^{B} \text{ within K-NN of } f_{ij:}^{A} \\ 0 & \text{otherwise} \end{cases}. \tag{6.2}$$

To make the sparse correlation map invariant to the ordering of the input images, we also perform this in the reverse direction ($B \rightarrow A$), and add the two one-sided correlation tensors together to obtain the final (symmetric) *sparse correlation tensor*:

$$c^{AB} = c^{A \rightarrow B} + c^{B \rightarrow A}. \tag{6.3}$$

This tensor uses a sparse representation, where only non-zero elements need to be stored. Note that the number of stored elements is, at most, $h \times w \times K \times 2$ which is in practice much less than the $h \times w \times h \times w$ elements of the dense correlation tensor, obtaining great memory savings in both the storage of this tensor and its subsequent processing. For example, for a feature map of size 100×75 and $K = 10$, the sparse representation takes 3.43MB vs. 215MB of the dense representation, resulting in a 12× reduction of the processing time. In the case of feature maps with 200×150 resolution, the sparse representation takes 13.7MB vs. 3433MB for the dense representation. This allows Sparse-NCNet to also process feature maps at this resolution, something that was not possible with NCNet due to the high memory requirements. The proposed *sparse correlation tensor* is a compromise between the common procedure of taking the best scoring match and the approach taken by NCNet, where all pairwise matches are stored. In this way, we can keep sufficient information in order to avoid early mistakes, while keeping low memory consumption and processing time.

Then the sparse correlation tensor is processed by a permutation-invariant CNN ($\hat{N}(\cdot)$), to produce the output filtered correlation map $\tilde{c}^{AB}$:

$$\tilde{c}^{AB} = \hat{N}(c^{AB}). \tag{6.4}$$

The permutation invariant CNN $\hat{N}(\cdot)$ consists of applying the 4D CNN $N(\cdot)$ twice such that the same output matches are obtained regardless of the order of the input images:

$$\hat{N}(c^{AB}) = N(c^{AB}) + \Big(N\Big((c^{AB})^T\Big)\Big)^T, \tag{6.5}$$

where by transposition we mean exchanging the first two dimensions with the last two dimensions, which correspond to the coordinates of the two input images. The 4D CNN $N(\cdot)$ operates on the 4D space of correspondences, and is trained to perform the neighbourhood consensus filtering. Note that while $N(\cdot)$ is a sparse CNN using submanifold sparse convolutions [Graham et al., 2018], where the active sites between the sparse input and output remain constant, the convolution kernel filters are dense (*i.e.* hypercubic).

While in the original NCNet method, a soft mutual nearest-neighbour operation $M(\cdot)$ is also performed, we have removed it as we noticed its effect was not significant when operating on the sparse correlation tensor. From the output correlation tensor $\tilde{c}^{AB}$, the output matches are computed by applying argmax at each coordinate:

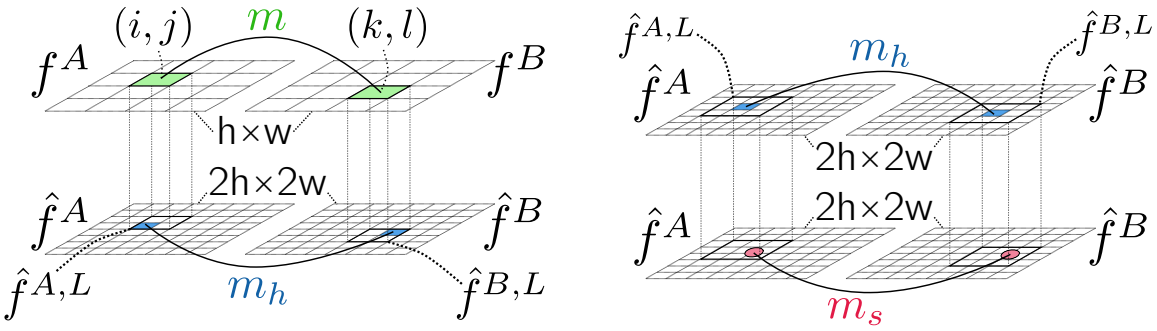
$$\Big((i,j),(k,l)\Big) \text{ a match if } \begin{cases} (i,j) = \underset{(a,b)}{\text{argmax}}\ \tilde{c}^{AB}_{abkl}, \text{ or} \\ (k,l) = \underset{(c,d)}{\text{argmax}}\ \tilde{c}^{AB}_{ijcd} \end{cases}, \tag{6.6}$$

where (i,j) is the match coordinate in the sampling grid of f^A, and (k,l) is the match coordinate in the sampling grid of f^B.

6.3.3 Match relocalization by guided search

While the sparsification of the correlation tensor presented in the previous section allows processing higher resolution feature maps, these are still several times smaller in resolution than the input images. Hence, they are not suitable for applications that require (sub)pixel feature localization such as camera pose estimation or 3D-reconstruction.

To address this issue, in this chapter we propose a two-stage relocalization module



(a) Hard relocalization (b) Soft relocalization

Figure 6-3: **Two-stage relocalization module.** (a) The hard relocalization step allows to increase by 2× the localization accuracy of the matches m outputted by Sparse-NCNet, which are defined on the $h \times w$ feature maps f^A and f^B. This is done by keeping the most similar match m_h between two 2×2 local features $\hat{f}^{A,L}$ and $\hat{f}^{B,L}$, cropped from the $2h \times 2w$ feature maps $\hat{f}^A$ and $\hat{f}^B$. (b) The soft relocalization step then refines the position of these matches in the $2h \times 2w$ grid, by computing sub-feature-grid soft localization displacements based on the softargmax operation.

based on the idea of guided search. The intuition is that we search for accurately localised matches on $2h \times 2w$ resolution dense feature maps, guided by the coarse matches output by Sparse-NCNet at $h \times w$ resolution. For this, dense features are first extracted at twice the normal resolution $(\hat{f}^A, \hat{f}^B) \in \mathbb{R}^{2h \times 2w \times c}$, which is done by upsampling the input image by 2× before feeding it into the feature extraction CNN $F(\cdot)$. Note that these higher resolution features are used for relocalization only, *i.e.* they are not used to compute the correlation tensor or processed by the 4D CNN for match-filtering, which would be too expensive. Then, these dense features are downsampled back to the normal $h \times w$ resolution by applying a 2×2 max-pooling operation with a stride of 2, obtaining f^A and f^B. These low resolution features $(f^A, f^B) \in \mathbb{R}^{h \times w \times c}$ are processed by Sparse-NCNet, which outputs matches in the form $m = \big((i,j),(k,l)\big)$, with the coordinates (i,j) and (k,l) indicating the position of the match in f^A and f^B, respectively, as described by (6.6).

Having obtained the output matches in $h \times w$ resolution, the first step (hard relocalization) consists in finding the best equivalent match in the $2h \times 2w$ resolution grid. This is done by analysing the matches between two local crops of the high resolution features $\hat{f}^A$ and $\hat{f}^B$, and keeping the highest-scoring one. The second step

(soft relocalization) then refines this correspondence further, by obtaining a sub-feature accuracy in the $2h \times 2w$ grid. These two relocalization steps are illustrated in Fig. 6-3, and are now described in detail.

Hard relocalization. The first step is hard relocalization, which can improve localization accuracy by $2\times$. For each match $m = \Big((i,j),(k,l)\Big)$, the $2\times$ upsampled coordinates $\Big((2i,2j),(2k,2l)\Big)$ are first computed, and 2×2 local feature crops $\hat{f}^{A,L}, \hat{f}^{B,L} \in \mathbb{R}^{2\times 2\times c}$ are sampled around these coordinates from the high resolution feature maps $\hat{f}^{A}$ and $\hat{f}^{B}$:

$$\hat{f}^{A,L} = (\hat{f}^{A}_{ab:})_{\substack{2i\leq a\leq 2i+1,\\ 2j\leq b\leq 2j+1}}, \tag{6.7}$$

and similarly for $\hat{f}^{B,L}$. This is done using a ROI-pooling operation [Girshick, 2015]. Finally, exhaustive matches between the local feature crops $\hat{f}^{A,L}$ and $\hat{f}^{B,L}$ are computed, and the output of the hard relocalization module is the displacement associated with the maximal matching score:

$$\Delta m_h = \Big((\delta i,\delta j),(\delta k,\delta l)\Big) = \underset{(a,b),(c,d)}{\operatorname{argmax}} \langle \hat{f}^{A,L}_{ab:}, \hat{f}^{B,L}_{cd:} \rangle. \tag{6.8}$$

Then, the final match location from the hard relocalization stage is computed as:

$$m_h = 2m + \Delta m_h = \Big((2i+\delta i, 2j+\delta j),(2k+\delta k, 2l+\delta l)\Big). \tag{6.9}$$

Note that the relocalized matches m_h are defined in a $2h \times 2w$ grid, therefore obtaining a $2\times$ increase in localization accuracy with respect to the initial matches m, which are defined in a $h \times w$ grid. Also note that while the implementation is different, the effect of the proposed hard relocalization is similar to the max-argmax operation used in NCNet (Chapter 5), while being more memory efficient as it avoids the computation of the a dense correlation tensor in high resolution.

Soft relocalization. The second step consists of a soft relocalization operation that obtains sub-feature localization accuracy in the $2h \times 2w$ grid of high resolution features $\hat{f}^{A}$ and $\hat{f}^{B}$. For this, new 3×3 local feature crops $(\hat{f}^{A,L}, \hat{f}^{B,L}) \in \mathbb{R}^{3\times 3\times c}$

are sampled around the coordinates of the estimated matches m_h from the previous relocalization stage. Note that no upsampling of the coordinates is done in this case, as the matches are already in the $2h \times 2w$ range. Then, soft relocalization displacements are computed by performing the softargmax operation [Yi et al., 2016] on the matching scores between the central feature of $\hat{f}^{A,L}$ and the whole of $\hat{f}^{B,L}$, and vice versa:

$$\Delta m_s = \Big((\delta i, \delta j), (\delta k, \delta l)\Big) \text{ where } \begin{cases} (\delta i, \delta j) = \underset{(a,b)}{\text{softargmax}} \langle \hat{f}^{A,L}_{ab:}, \hat{f}^{B,L}_{11:} \rangle \\ (\delta k, \delta l) = \underset{(c,d)}{\text{softargmax}} \langle \hat{f}^{A,L}_{11:}, \hat{f}^{B,L}_{cd:} \rangle \end{cases} . \tag{6.10}$$

The intuition of the softargmax operation is that it computes a weighted average of the candidate positions in the crop where the weights are given by the softmax of the matching scores. The final matches from soft relocalization are obtained by applying the soft displacements to the matches from hard relocalization: $m_s = m_h + \Delta m_s$.

6.4 Experimental evaluation

We evaluate the proposed Sparse-NCNet method on three different benchmarks: (i) HPatches Sequences, which evaluates the matching task directly, (ii) InLoc, which targets the problem of indoor 6-dof camera localization and (iii) Aachen Day-Night, which targets the problem of outdoor 6-dof camera localization with challenging day-night illumination changes. We first present the implementation details followed by the results on these three benchmarks.

Implementation details. We train the Sparse-NCNet model following the training protocol from NCNet (Chapter 5). We use the IVD dataset with the weakly-supervised mean matching score loss of Sec. 5.3.6 for training. The 4D CNN $N(\cdot)$ has two sparse convolution layers with 3^4 sized kernels, with 16 output channels in the hidden layer. A value of $K = 10$ is used for computing c^{AB} (6.3). The model is implemented using PyTorch [Paszke et al., 2017], MinkowskiEngine [Choy et al., 2019a] and Faiss [Johnson et al., 2017], and trained for 5 epochs using Adam [Kingma and Ba, 2015] with a

learning rate of 5×10^{-4}. A pretrained ResNet-101 (up to `conv_4_23`) with no strided convolutions in the last block is used as the feature extractor $F(\cdot)$. This feature extraction model is not finetuned as the training dataset is small (3861 image pairs) and that would lead to overfitting and loss of generalisation. The softargmax operation in (6.10) uses a temperature value of 10.

6.4.1 HPatches Sequences

The HPatches Sequences [Balntas et al., 2017] benchmark assesses the matching accuracy under strong *viewpoint* and *illumination* variations. We follow the evaluation procedure from [Dusmanu et al., 2019], where 108 image sequences are employed, each from a different planar scene, and each containing 6 images. The first image from each sequence is matched against the remaining 5 images. The benchmark employs 56 sequences with viewpoint changes, and constant illumination conditions, and 52 sequences with illumination changes and constant viewpoint. The metric used for evaluation is the mean matching accuracy (MMA) [Dusmanu et al., 2019], which assesses the fraction of correct matches under different tolerance:

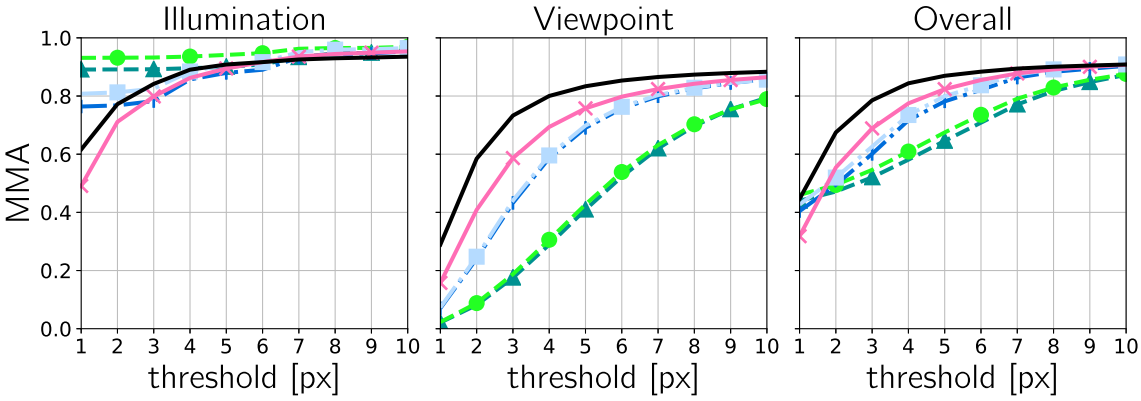
$$\mathrm{MMA}\Big(\{(p_i^A, p_i^B)\}_{i=1}^N; t\Big) = \frac{\sum_{i=1}^N \mathbb{1}_{>0}\Big(t - \|\mathcal{T}_H(p_i^A) - p_i^B\|\Big)}{N}, \tag{6.11}$$

where $\{(p_i^A, p_i^B)\}_{i=1}^N$ is the set of matches to be evaluated, $\mathcal{T}_H(p_i^A)$ is the warped point p_i^A using the ground-truth homography H, $\mathbb{1}_{>0}$ is the indicator function for positive numbers, and t is the chosen tolerance threshold (in pixels).

Sparse-NCNet vs. NCNet. In Fig. 6-4 we compare the matching quality of the proposed Sparse-NCNet model and the NCNet model. We first compare both methods under equal conditions, both without relocalization (methods A1 vs. A2), and with hard relocalization only (methods B1 vs. B2). The results in Fig. 6-4 show that Sparse-NCNet can obtain significant reductions in processing time and memory consumption, while keeping almost the same matching performance. In addition, our proposed two-stage relocalization module can improve performance with a minor increase in

	Method	Feature resolution	Reloc. method	Reloc. resolution	Mean time (s)	Peak VRAM (MB)
A1.	Sparse-NCNet	100 × 75	—	—	0.83	251
A2.	NCNet	100 × 75	—	—	9.81	5763
B1.	Sparse-NCNet	100 × 75	H	200 × 150	1.55	1164
B2.	NCNet	100 × 75	H	200 × 150	10.56	7580
C1.	Sparse-NCNet	100 × 75	H+S	200 × 150	1.56	1164
C2.	Sparse-NCNet	200 × 150	H+S	400 × 300	7.51	2391

(a) Time and GPU memory comparison (Tesla T4 GPU)



(b) MMA on HPatches Sequences

Figure 6-4: **Sparse-NCNet vs. NCNet on HPatches**. Sparse-NCNet can obtain equivalent results to NCNet, both without relocalization (*cf.* A1 vs. A2), and with hard relocalization (H) (*cf.* B1 vs. B2), while greatly reducing execution time and memory consumption. The proposed two-stage relocalization (H+S) brings an improvement in matching accuracy with a minor increase in execution time (*cf.* C1 vs. B1). Finally, the reduced memory consumption in Sparse-NCNet allows for processing in higher resolution, which produces the best results, while still being faster and more memory efficient than NCNet (*cf.* C2 vs. B2).

processing time (methods C1 vs. B1). Finally, the reduced memory consumption allows for processing of higher resolution 200×150 feature maps, which is not possible for NCNet. Our proposed method in higher resolution (method C2) produces the best results while still being 30% faster and 3× more memory efficient than the best NCNet variant (method B2).

Sparse-NCNet vs. state-of-the-art methods. In addition, we compare the performance of Sparse-NCNet against several methods, including state-of-the-art trainable methods such as SuperPoint [DeTone et al., 2018], D2-Net [Dusmanu et al., 2019]

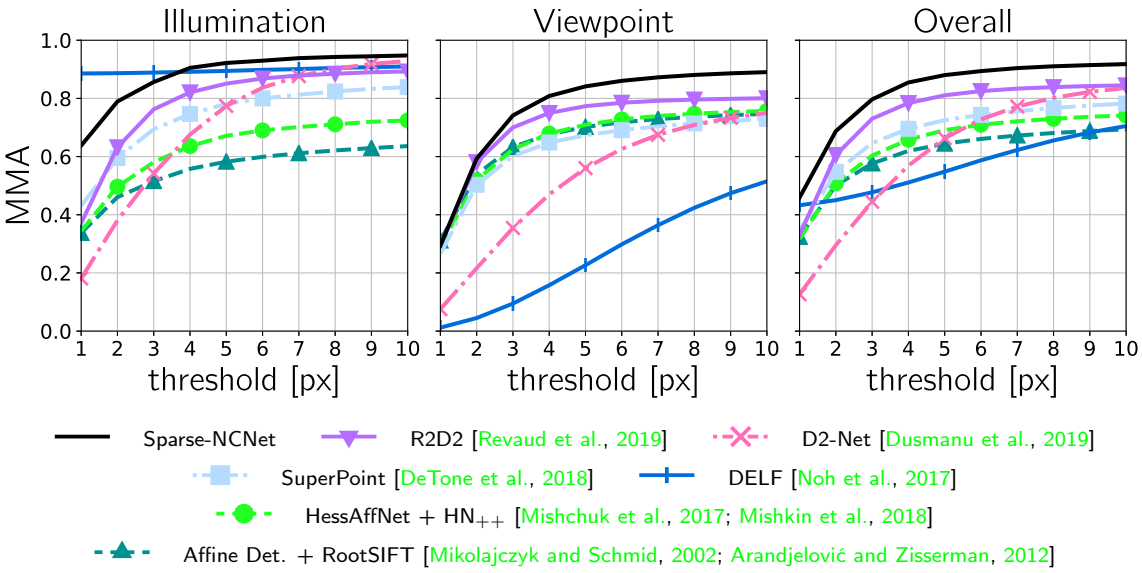


Figure 6-5: **Sparse-NCNet vs. state-of-the-art on HPatches.** The MMA of Sparse-NCNet and several state-of-the-art methods is shown. Sparse-NCNet obtains the best results overall with a large margin over the recent R2D2 method.

or R2D2 [Revaud et al., 2019]. The mean-matching accuracy results are presented in Fig. 6-5. For all other methods, the top 2000 features points were selected from each image, and matched enforcing mutual nearest-neighbours, yielding approximately 1000 correspondences per image pair. For Sparse-NCNet, the top 1000 correspondences were selected for each image pair, for a fair comparison. Sparse-NCNet obtains the best results for the *illumination* sequences for thresholds higher than 4 pixels, and in the *viewpoint* sequences for all threshold values. Sparse-NCNet obtains the best results overall, with a large margin over the state-of-the-art R2D2 method. We believe this could be attributed to the usage of dense descriptors (which avoid the loss of detections) together with an increased matching robustness from performing neighbourhood consensus.

Qualitative results are presented in Figures 6-6 and 6-7. We compare the MMA of Sparse-NCNet with the state-of-the-art methods SuperPoint [DeTone et al., 2018], D2-Net [Dusmanu et al., 2019] and R2D2 [Revaud et al., 2019], which are trainable methods for joint detection and description on local features. The correctly matched points are shown in green, while the incorrectly matched ones are shown in red, for a

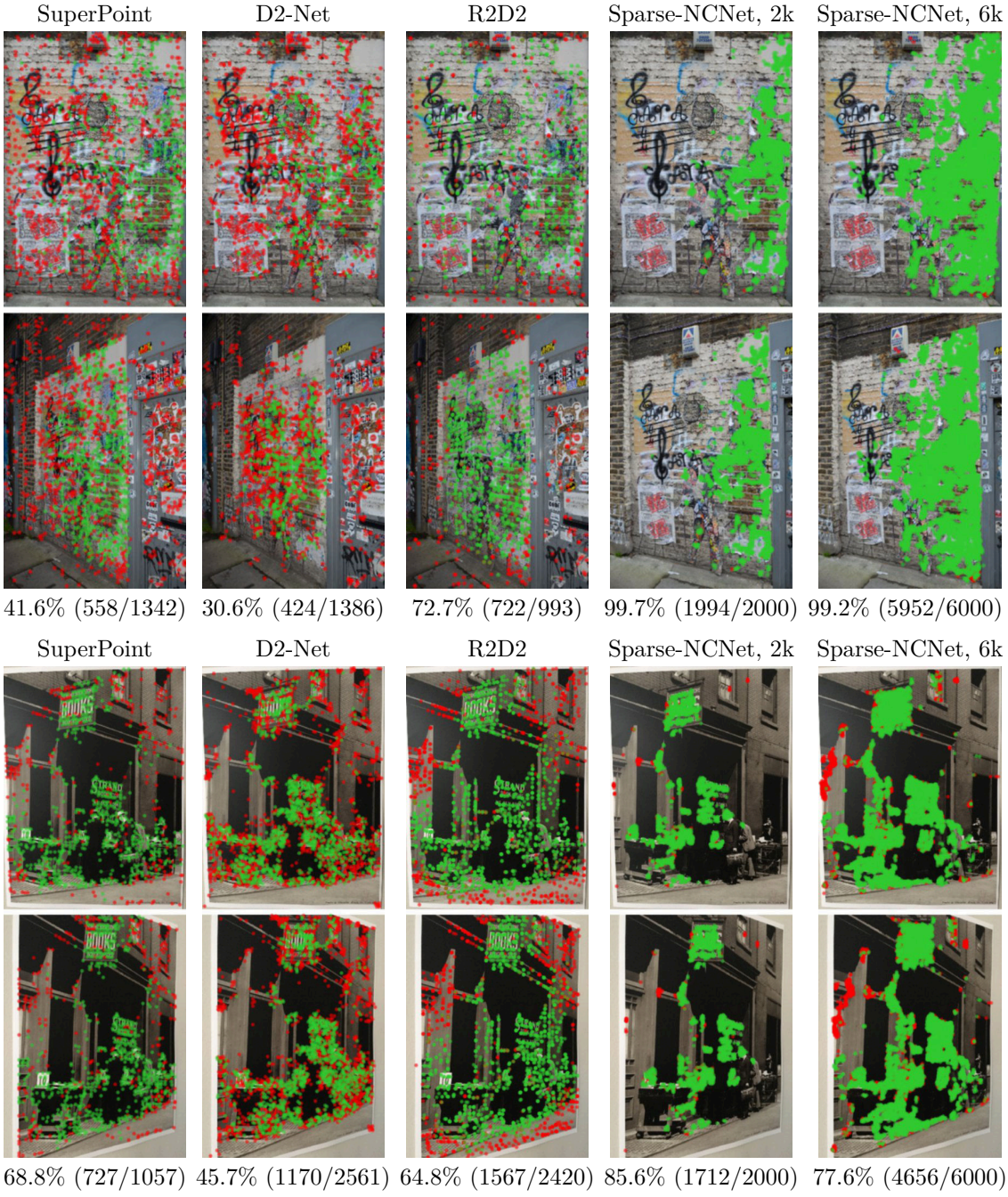


Figure 6-6: **HPatches qualitative results (viewpoint).** We present the results of Sparse-NCNet, along with state-of-the-art methods SuperPoint [DeTone et al., 2018], D2-Net [Dusmanu et al., 2019] and R2D2 [Revaud et al., 2019]. The correct correspondences are shown in green, and the incorrect ones in red for a threshold $t = 3$px. Below each pair we indicate the fraction of correct matches (both in percentage and absolute values). Our method is presented for both the top 2K matches and the top 6K matches, and it obtains the largest fraction of correct matches for both cases. Examples are from the *viewpoint* sequences.

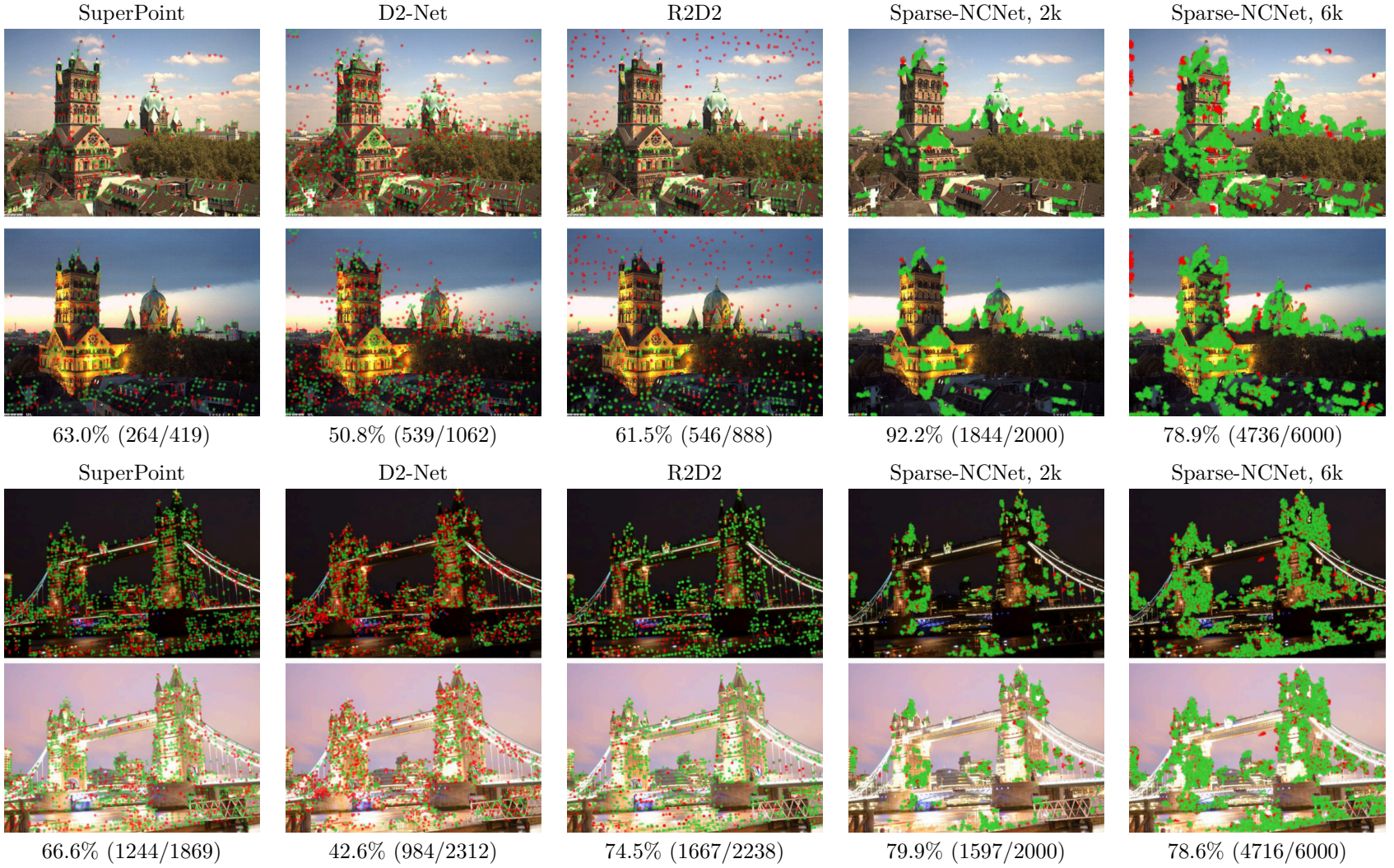


Figure 6-7: **HPatches qualitative results (illumination).** We present the results of Sparse-NCNet, along with state-of-the-art methods SuperPoint [DeTone et al., 2018], D2-Net [Dusmanu et al., 2019] and R2D2 [Revaud et al., 2019]. The correct correspondences are shown in green, and the incorrect ones in red for a threshold $t = 3$px. Below each pair we indicate the fraction of correct matches (both in percentage and absolute values). Our method is presented for both the top 2K and top 6K matches, and it obtains the largest fraction of correct matches for both cases. Examples are from the *illumination* sequences.

threshold value $t = 3$ pixels. For the proposed Sparse-NCNet, results are presented for two different numbers of matches, 2000 and 6000. Results show that our method produces the largest fraction of correct matches, even when considering as many as 6000 correspondences. In particular, note that our method is able to produce a large amount of correct correspondences even under strong illumination changes, as shown in Fig. 6-7. Furthermore, note that the nature of the correspondences produced by Sparse-NCNet is different from those of local feature methods. While local feature methods can only produce correspondences on the detected points, which are the local extrema of a particular feature detection function, our method produces densely packed sets of correspondences. A discussion about this behaviour is presented in Sec. 6.5.

6.4.2 InLoc benchmark

The InLoc benchmark [Taira et al., 2018] targets the problem of indoor localization. It contains a set of *database* images of a building, obtained with a 3D scanner, and a set of *query* images from the same building, captured with a cell-phone several months later. The task is then to obtain the 6-dof camera positions of the query images. We follow the DensePE approach proposed [Taira et al., 2018] to find the top 10 candidate database images for each query, and employ Sparse-NCNet to obtain matches between them. Then, we follow again the procedure in [Taira et al., 2018] to obtain the final estimated 6-dof query pose, which consists of running PnP [Gao et al., 2003] followed by dense pose verification [Taira et al., 2018].

The results are presented in Fig. 6-8. First, we observe that Sparse-NCNet with hard relocalization (H) and a resolution of 100×75 obtains equivalent results to NCNet (methods B vs. C), while being almost $7\times$ faster and requiring $6.5\times$ less memory, confirming what was already observed in the HPatches benchmark (*cf.* B1 vs. B2 in Fig. 6-4a). Moreover, our proposed Sparse-NCNet method with two-stage relocalization (H+S) in the higher 200×150 resolution (method A) obtains the best results and sets a new state-of-the-art for this benchmark. Recall that it is impossible to use the original NCNet on the higher resolution due to its excessive memory

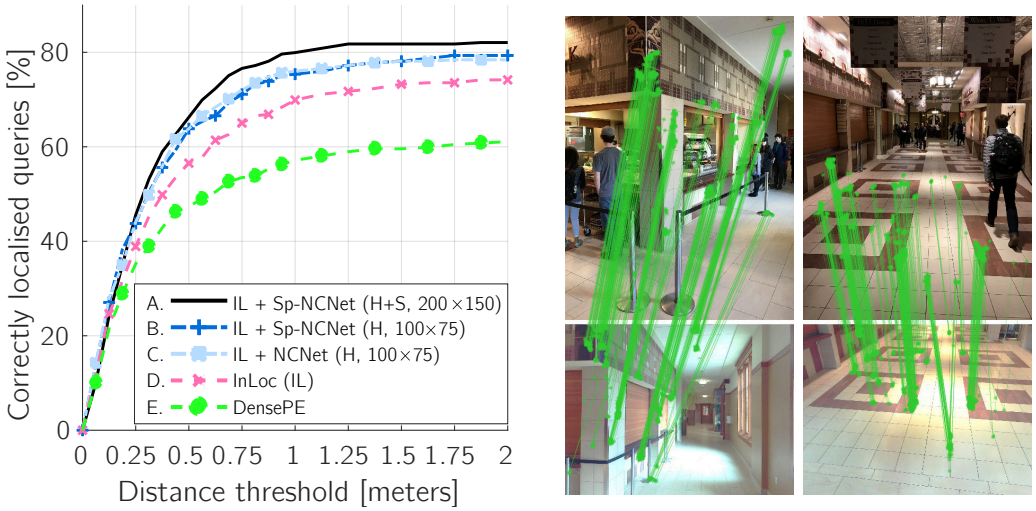


Figure 6-8: **Results on the InLoc benchmark for long-term indoor localization.** (Left) Our proposed method (A) obtains state-of-the-art results on this benchmark. (Right) Our method obtains correspondences in challenging indoor scenes with repetitive patterns and low amount of texture. Top: query images. Bottom: matched database images captured from different viewpoints. Correspondences produced by our approach are overlaid in green. The query and database images were taken several months apart.

requirements.

More qualitative examples are presented in Fig. 6-9. Each image pair is composed of a query image (top row) captured with a cell-phone and a database image (middle row), captured several months earlier with a 3D scanner. Note that the illumination conditions in the two types of images are different. Furthermore, because of the time difference between both images, some objects may have been displaced (*e.g.* furniture) and some aspects of the scene may have changed (*e.g.* wall decoration). For ease of visualisation, we overlay only the top 500 correspondences for each image pair, which appear in green. These correspondences have not been geometrically verified, and therefore contain a certain fraction of incorrect matches. Note however, that most matches are coherent and the few incorrect outliers are likely to be removed when running RANSAC [Fischler and Bolles, 1981] within the PnP pose solver [Gao et al., 2003], therefore obtaining a good pose estimate. Also note how Sparse-NCNet is able to obtain correspondences in low textured areas such as walls or ceilings, or on repetitive patterns such as carpets.

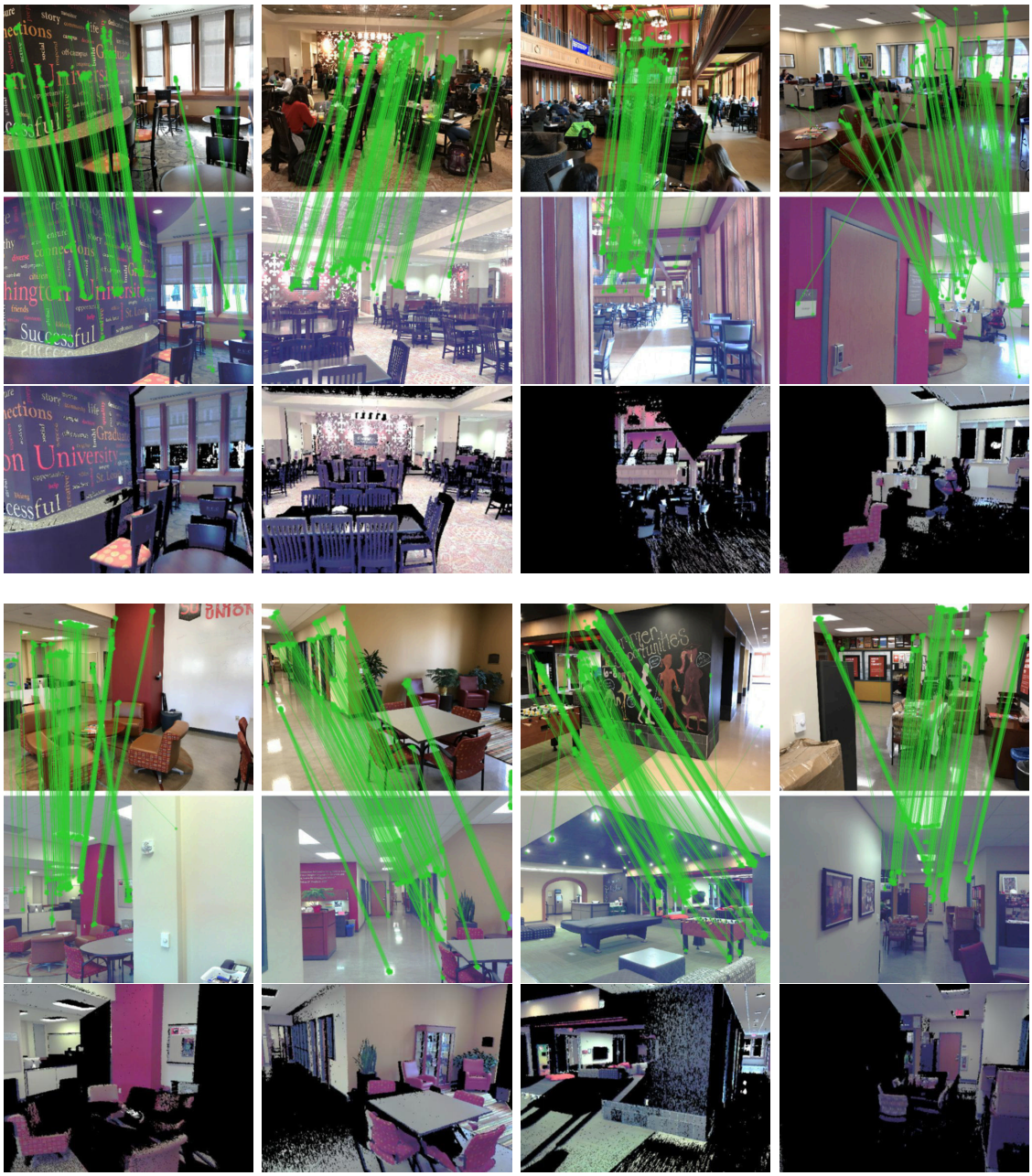


Figure 6-9: **InLoc qualitative results.** For each image pair, we show the top 500 matches produced by Sparse-NCNet between the query image (top row) and database image (middle row). In addition we show the rendered scene from the estimated query 6-dof pose (bottom row), obtained by running RANSAC+PnP[Fischler and Bolles, 1981; Gao et al., 2003] on our matches. Note these rendered images are well aligned with the query images, demonstrating that the estimated poses have low translation and rotation errors.

Method	Correctly localised queries (%) 0.5m, 2°	1.0m, 5°	5.0m, 10°
RootSIFT [Lowe, 2004; Arandjelović and Zisserman, 2012]	36.7	54.1	72.5
DenseSfM [Sattler et al., 2018]	39.8	60.2	84.7
HessAffNet + HN_{++} [Mishchuk et al., 2017; Mishkin et al., 2018]	39.8	61.2	77.6
DELF [Noh et al., 2017]	38.8	62.2	85.7
SuperPoint [DeTone et al., 2018]	42.8	57.1	75.5
D2-Net [Dusmanu et al., 2019]	44.9	66.3	**88.8**
D2-Net (Multi-scale) [Dusmanu et al., 2019]	44.9	64.3	**88.8**
R2D2 (patch = 16) [Revaud et al., 2019]	44.9	67.3	87.8
R2D2 (patch = 8) [Revaud et al., 2019]	**45.9**	66.3	**88.8**
Sparse-NCNet (H, 200 × 150)	44.9	**68.4**	86.7

Table 6.1: **Results on Aachen Day-Night.** Sparse-NCNet is able to localise a similar number of queries to R2D2 and D2-Net.

6.4.3 Aachen Day-Night

The Aachen Day-Night benchmark [Sattler et al., 2018] targets 6-dof outdoor camera localization under challenging illumination conditions. It contains 98 night-time query images from the city of Aachen, and a shortlist of 20 day-time images for each night-time query. Sparse-NCNet is used to obtain matches between the query and images in the short-list. The resulting matches are then processed by the 3D reconstruction software COLMAP [Schönberger and Frahm, 2016] to obtain the estimated query poses.

The results are presented in Table 6.1. Sparse-NCNet presents a similar performance to the state-of-the-art methods D2-Net [Dusmanu et al., 2019] and R2D2 [Revaud et al., 2019]. Note that the results of these three different methods differ by only a few percent, which represents only 1 or 2 additionally localised queries, from the 98 total night-time queries. The proposed Sparse-NCNet obtains state-of-the-art results for the 1m and 5° threshold, being able to localise 68.4% of the queries (67 out of 98). One qualitative example from this benchmark is presented in Fig. 6-1.

Additional qualitative examples are shown in Fig. 6-10. We show several image pairs composed of night query images (top) and their top matching database images (bottom), according to the average matching score of Sparse-NCNet. For each image pair, we overlay the top 500 correspondences obtained with Sparse-NCNet. Note that these correspondences were not geometrically verified by any means. Nevertheless, as seen in Fig. 6-10, most correspondences are coherent and seem to be correct, despite the strong changes in illumination between night and day images.

Figure 6-10: **Aachen day-night results.** We show the top 500 correspondences obtained by Sparse-NCNet between the night query image (top) and the database day image (bottom). Note that the large majority of matches are correct, despite the strong illumination changes.

6.5 Insights about Sparse-NCNet

In this section we provide additional insights about the way Sparse-NCNet operates, which differs from traditional local feature detection and matching methods. In Fig. 6-11 we plot the top N matches produced by Sparse-NCNet for different values of N: 100 (left column), 400 (middle column) and 1600 (right column). By comparing the middle column (showing the top 400 matches) with the left column (showing the top 100), we can observe that many of the additional 300 matches are close to the initial 100 matches. A similar effect is observed when comparing the right column (top 1600 matches) with the middle column (top 400 matches). This could be attributed to the fact that Sparse-NCNet propagates information from the strongest matches to their neighbours. In this sense, strong matches, which are typically non-ambiguous ones, can help in matching their neighbouring features, which might not be so discriminative.

6.6 Conclusion

In this chapter we have presented the Sparse Neighbourhood Consensus Networks for efficiently estimating correspondences between images. Our approach overcomes the main limitations of the original Neighbourhood Consensus Networks that demonstrated promising results on challenging matching problems, making these models practical and widely applicable. The proposed model jointly performs feature extraction, matching and robust match filtering in a computationally efficient manner, outperforming state-of-the-art results on two challenging matching benchmarks. The entire pipeline is end-to-end trainable, which opens-up the possibility for including additional modules for specific downstream problems such as camera pose estimation or 3D reconstruction.

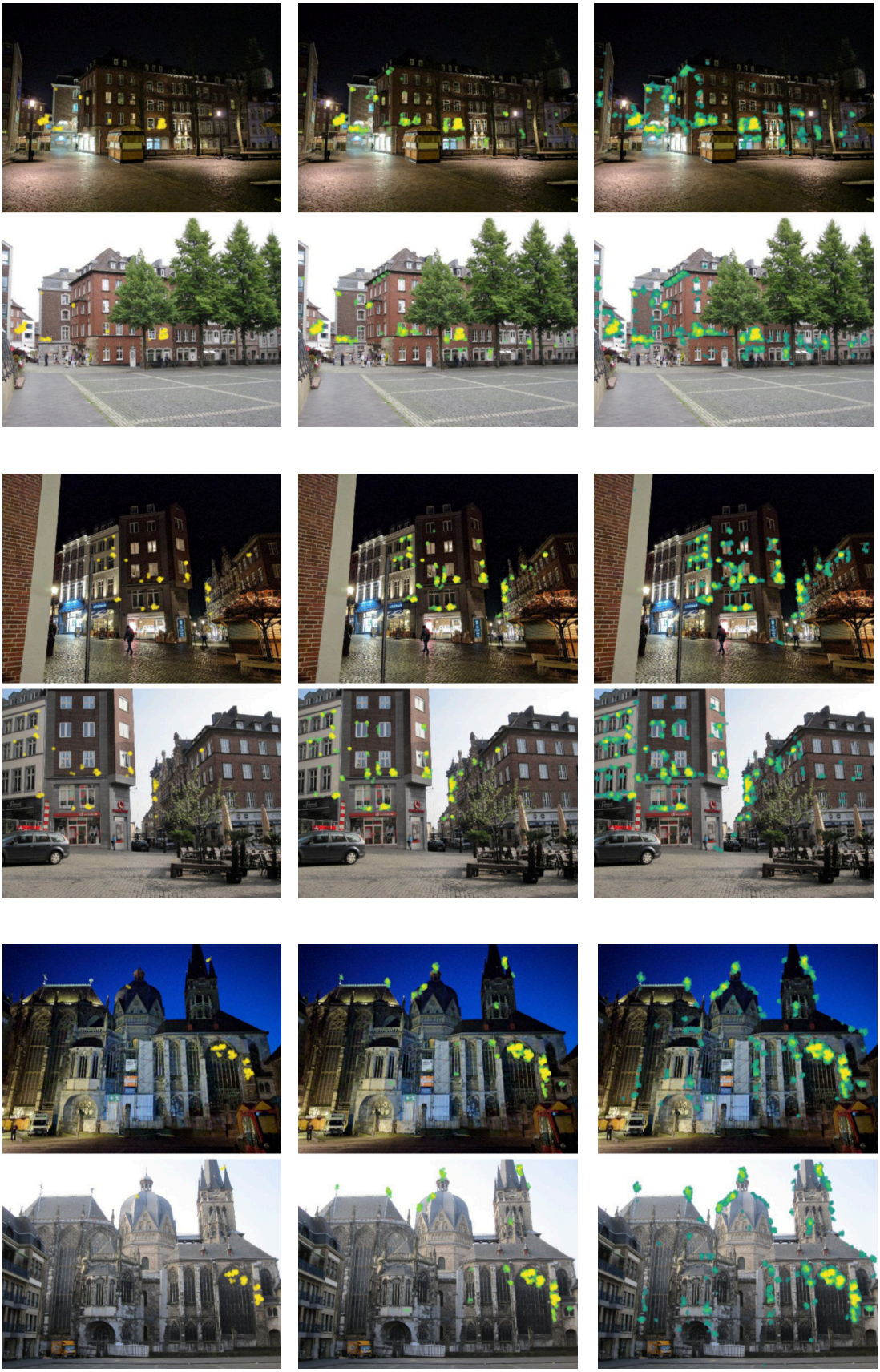


Figure 6-11: **Insights about Sparse-NCNet.** We show the top N matches between each pair of images for different values of N. The strength of the match is shown by color (the more yellow the stronger). Please note how new matches tend to appear close to high scoring matches, demonstrating the propagation of information in Sparse-NCNet.

Chapter 7

Conclusions

In this final chapter, we review the main contributions of this thesis and present some possible research directions for future work.

7.1 Contributions

In this thesis, we have focused on developing trainable models for establishing correspondences between pairs of images. This presented several challenges, such as designing a suitable model which is end-to-end trainable and finding a suitable training scheme. We have investigated several model architectures and training schemes for solving this problem. The contributions developed in each chapter are detailed next.

In Chapter 3, we have developed a Siamese end-to-end trainable CNN architecture for image alignment which outputs parameters of a geometric transformation such as affine, homography, or thin-plate-spline. This architecture has a modular structure which follows the traditional matching pipeline composed of a *feature extraction* stage, followed by a *matching* stage, and a *transformation estimation* stage. In particular, we have proposed to use densely extracted CNN features for the *feature extraction* stage, together with a *correlation operation* which outputs the exhaustive set of matching scores between the densely extracted features. This approach for feature extraction and matching was adopted in all the subsequent chapters of this thesis, as well as by other works, for example, [Novotny et al., 2018; Kim et al., 2018a; Jeon et al., 2018;

Lee et al., 2019; Melekhov et al., 2019; Truong et al., 2020]. In order to train this model, we have developed a suitable training scheme following a self-supervised approach, where training image pairs are generated by synthetically warping a set of natural images with randomly sampled transformations, as well as an appropriate training loss that is agnostic to the type of geometric transformation and its parametrization. Finally, we have shown that the usage of a correlation operation for matching provides good generalization properties, by showing successful results in both instance- and category-level alignment despite the domain gap between training and testing.

In Chapter 4, we have extended the previously proposed model with a weakly-supervised training module, which does not require access to the ground-truth geometric transformation, and thus allows training from corresponding pairs of natural images with no further annotation. This weakly-supervised training module is based on the concept of a soft-inlier count, which borrows inspiration from the inlier count that is used in RANSAC [Fischler and Bolles, 1981]. This method was extended by Wang et al. [2019], who propose to use cycle-consistency across several video frames as a supervisory signal. We have shown that the proposed weakly-supervised training can improve performance over the previous self-supervised training alone, achieving state-of-the-art results in category-level alignment.

In Chapter 5, we have proposed a new end-to-end trainable model for image correspondence. Contrary to the model from previous chapters which outputs parameters of a geometric transformation, this model can output a discrete set of correspondences and is, therefore, better suited for instance-level matching problems. This is achieved by replacing the regressor CNN previously used for transformation estimation, by a 4D Neighbourhood Consensus Network (NCNet) that operates on the space of 4D matching scores. We have shown that NCNet models are useful both for instance and category-level matching obtaining state-of-the-art results on tasks as diverse as indoor localization and semantic keypoint transfer. Our NCNet model has been extended by Li et al. [2020] who propose adaptive neighbourhood sizes and by Yang and Ramanan [2019] who propose alternative separable 4D convolutions and employed by other recent work [Chen et al., 2019; Laskar et al., 2020].

In Chapter 6, we have proposed a more efficient formulation for Neighbourhood Consensus Networks, which seeks to overcome the main drawbacks of the original formulation. For this, the proposed Sparse-NCNet model uses a sparse correlation tensor, that contains only the most promising matches between the dense features of each image, and which is processed by a sparse submanifold CNN. We have shown that the resulting model can run an order of magnitude faster and with fewer memory requirements than the original model while providing equivalent results. Furthermore, we have proposed a two-stage relocalization approach that improves the localization accuracy of matches, which has a direct positive impact on camera pose estimation and 3D reconstruction results.

7.2 Future work

In this section, we analyze some possible future research directions which could extend the work presented in this thesis.

Computational efficiency. Most recent methods for estimating correspondences between images rely on deep CNN descriptors which are computationally intensive to extract, resulting in higher processing times than previous hand-crafted methods. In addition, trainable match filtering methods such as NCNet (chapter 5) have $O(n^2)$ complexity with respect to the number of features n. These factors limit the applicability of both trainable descriptors and trainable match filtering methods for processing a large number of images, such as in the case of large-scale 3D reconstruction, or real-time applications such as SLAM. To obtain more computationally efficient trainable descriptors, distillation approaches [Hinton et al., 2015] could be used for transferring knowledge from deep models towards shallower and more efficient ones. Another alternative is to directly adopt shallower CNN models for image description and train them from scratch for the correspondence task, as done by Revaud et al. [2019]. One additional approach could involve using half-precision floating-point arithmetic to speed-up the computations and reduce memory requirements, as proposed

by Micikevicius et al. [2017]. Finally, to produce fast trainable approaches for match filtering, sparse sub-manifold convolutions can be used, as proposed in Chapter 6, or graph-convolutional networks, as proposed by Sarlin et al. [2019].

Localization accuracy. Due to the high computational requirements discussed above, many recent methods extract CNN features densely along a grid that has a subsampling factor of 4 or 8 with respect to the original image resolution. This subsampling results from the strided-convolutions or pooling operations in the feature extraction CNN, which are often inherited from image classification CNN architectures where such subsampling is desirable for spatially-aggregating the visual information from large image areas. However, for the task of image correspondence estimation, these subsampling operations result in less accurately localized matches, which can be problematic for tasks such as 3D reconstruction. Different approaches have been proposed to address this issue, by either producing a detection map in full image resolution [DeTone et al., 2018], fitting a quadratic function to a 2D keypoint response map [Dusmanu et al., 2019], replacing the subsampling operations by dilated convolutions [Revaud et al., 2019], using guidance from higher-resolution intermediate features [Widya et al., 2018] or using guidance from higher-resolution local matches as we proposed in Chapter 6. However, despite these efforts, 3D reconstruction benchmarks show that the reprojection error of trainable methods is in most cases still above 1px while SIFT can achieve sub-pixel reprojection errors. Therefore, improving the localization accuracy of the correspondences produced by trainable methods to match or surpass that of SIFT is still an ongoing research effort.

Large-scale training. Most trainable methods for correspondence estimation have relied on 3D reconstruction from structure-from-motion (SfM) as a way to generate ground-truth correspondences for training. However, this may introduce a bias for DoG detections if only the sparse 3D point-cloud from SfM methods based on DoG detections are used. An alternative is to obtain dense correspondences between image pairs using optical flow, and use the sparse correspondences and camera poses from

the SfM reconstruction as guidance [Revaud et al., 2019]. However, this still requires a successful 3D reconstruction (typically obtained with DoG keypoints and SIFT features), which limits the difficulty of pairs that can be used during training. Weakly-supervised approaches requiring annotation at the level of image pairs (chapter 4) or at the level of single images [Novotny et al., 2017] have been proposed for the task of category-level matching where object classes are well defined. However, it is not clear how to translate these approaches to instance-level matching, where the notion of "object class" is ill-defined. In the past, weakly-supervised losses have been proposed for the task of visual localization, relying solely on GPS information [Arandjelović et al., 2016]. However, this approach has not yet been extended to the task of learning image correspondences. We believe that this approach has the potential of enabling large-scale training with no manual annotation, possibly allowing to learn richer feature descriptors, and therefore constitutes a possible future research direction.

Identification of non-matching regions. In Chapter 3, we have presented an approach towards identifying non-matching image regions, obtained by thresholding 2D similarity maps. However, this idea was not further explored. Nevertheless, we believe that for a correspondence estimation model to be most useful, both the corresponding and non-corresponding regions should be identified. In the case of large transient objects which appear only in one of the images, additional semantic information could be used to this end. In the case of small regions that only appear in one image due to self-occlusions, reasoning with an estimated 3D scene structure could be helpful. The incorporation of additional semantic and geometric information is discussed next.

Incorporating additional semantic and geometric information. Current methods for estimating correspondences operate on deep CNN features, which are typically pre-trained on an image classification task, and therefore act as high-level descriptors. However, the recent success of CNN methods in tasks such as instance-level semantic segmentation [He et al., 2017], monocular depth estimation [Luo et al., 2020; Lasinger et al., 2019] or surface normal estimation [Zamir et al., 2018] offers the

possibility of incorporating additional semantic and geometric information to the correspondence assignment problem. This could be particularly useful for (i) identifying static and transient objects in the scene, (ii) developing top-down correspondence estimation methods, where high-level correspondences between objects-instances are first established (or not established, for transient objects that appear only in one image), and matches are then searched within the corresponding objects only, (iii) guiding the matching process through geometric reasoning using noisy priors such as depth estimated from each image independently using a monocular method. In the past, Arandjelović and Zisserman [2014] have shown the benefits of incorporating semantic information into SIFT descriptors for matching as well as for image retrieval. Taira et al. [2019] show that semantic information in the form of segmentation masks and geometric information in the form of surface normals can be beneficial for the task of camera pose estimation. However, we believe that further research can be done to incorporate these types of information in more general correspondence estimation methods.

Modern dense 3D reconstruction. Recently, much progress has been made in obtaining dense correspondences in unconstrained settings (contrary to stereo vision or optical flow), with large viewpoint and illumination changes. in Chapters 5 and 6 we have presented a method that operates on densely extracted features and can produce semi-dense correspondences in such difficult matching settings. A different recent method can obtain a regular correspondence field for both instance and category-matching problems [Truong et al., 2020]. It would be desirable if these dense approaches could be used for the task of 3D reconstruction. However, current structure-from-motion pipelines are heavily based on sparse local features with dense estimation only happening at a later stage through multi-view-stereo methods. Using current structure-from-motion pipelines with dense correspondences is not straightforward. Therefore, we believe that more modern 3D reconstruction pipelines and software should be developed, which can fully profit from recent methods for dense correspondence estimation. Note that using dense correspondence estimation

methods for 3D reconstruction has the potential of producing more complete 3D models and to require a smaller set of images for a successful reconstruction.

Bibliography

Agarwal, S., Furukawa, Y., Snavely, N., Simon, I., Curless, B., Seitz, S. M., and Szeliski, R. Building Rome in a day. *Communications of the ACM*, 54(10):105–112, 2011. 46, 102

Alcantarilla, P. F., Nuevo, J., and Bartoli, A. Fast explicit diffusion for accelerated features in nonlinear scale spaces. In *Proc. BMVC.*, 2013. 23

Alcantarilla, P. F., Bartoli, A., and Davison, A. J. KAZE features. In *Proc. ECCV*, 2012. 23

Altwaijry, H., Trulls, E., Hays, J., Fua, P., and Belongie, S. Learning to match aerial images with deep attentive architectures. In *Proc. CVPR*, 2016. 48

Arandjelović, R., Gronat, P., Torii, A., Pajdla, T., and Sivic, J. NetVLAD: CNN architecture for weakly supervised place recognition. In *Proc. CVPR*, 2016. 31, 50, 61, 63, 74, 80, 134, 137, 165

Arandjelović, R. and Zisserman, A. Three things everyone should know to improve object retrieval. In *Proc. CVPR*, pages 2911–2918, 2012. 23, 123, 150, 156

Arandjelović, R. and Zisserman, A. Visual vocabulary with a semantic twist. In *Proc. ACCV*, 2014. 166

Aubry, M., Russell, B., and Sivic, J. Painting-to-3D model alignment via discriminative visual elements. *ACM Transactions on Graphics*, 2013. 80

Azizpour, H., Razavian, A. S., Sullivan, J., Maki, A., and Carlsson, S. Factors of transferability for a generic convnet representation. *IEEE PAMI*, 38(9):1790–1802, 2015. 50

Babenko, A. and Lempitsky, V. Aggregating local deep features for image retrieval. In *Proc. ICCV*, 2015. 50

Badrinarayanan, V., Kendall, A., and Cipolla, R. SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE PAMI*, 39(12):2481–2495, 2017. 116

Balntas, V., Lenc, K., Vedaldi, A., and Mikolajczyk, K. HPatches: A Benchmark and Evaluation of Handcrafted and Learned Local Descriptors. In *Proc. CVPR*, 2017. 119, 121, 136, 148

Balntas, V., Johns, E., Tang, L., and Mikolajczyk, K. PN-Net: Conjoined triple deep network for learning local image descriptors. *arXiv preprint arXiv:1601.05030*, 2016a. 25, 26, 48, 102, 104, 105, 134, 136

Balntas, V., Riba, E., Ponsa, D., and Mikolajczyk, K. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *Proc. BMVC.*, 2016b. 25, 26, 102, 104, 105, 134, 136

Balntas, V., Hammarstrand, L., Heijnen, H., Kahl, F., Maddern, W., Mikolajczyk, K., Pajdla, T., Pollefeys, M., Sattler, T., Schönberger, J. L., Speciale, P., Sivic, J., Toft, C., and Torii, A. Workshop in Long-Term Visual Localization under Changing Conditions, CVPR 2019. https://www.visuallocalization.net/workshop/cvpr/2019/, 2019. 134

Bay, H., Tuytelaars, T., and Van Gool, L. SURF: Speeded up robust features. In *Proc. ECCV*, 2006. 23, 48, 134

Beaudet, P. R. Rotationally invariant image operators. In *Proc. 4th Int. Joint Conf. Pattern Recog, Tokyo, Japan, 1978*, 1978. 20

Berg, A., Berg, T., and Malik, J. Shape matching and object recognition using low distortion correspondence. In *Proc. CVPR*, 2005. 35, 36, 37, 40, 48, 49, 58, 82, 84

Berg, A. C. and Malik, J. Geometric blur for template matching. In *Proc. CVPR*, 2001. 35

Bian, J., Lin, W.-Y., Matsushita, Y., Yeung, S.-K., Nguyen, T.-D., and Cheng, M.-M. GMS: Grid-based motion statistics for fast, ultra-robust feature correspondence. In *Proc. CVPR*, 2017. 34, 103, 106, 107, 123, 140

Bookstein, F. L. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE PAMI*, 1989. 56, 57

Boureau, Y.-L., Bach, F., LeCun, Y., and Ponce, J. Learning mid-level features for recognition. In *Proc. CVPR*, 2010. 37

Brachmann, E., Krull, A., Nowozin, S., Shotton, J., Michel, F., Gumhold, S., and Rother, C. DSAC - Differentiable RANSAC for camera localization. In *Proc. CVPR*, 2017. 83

Brachmann, E. and Rother, C. Neural-guided RANSAC: Learning where to sample model hypotheses. In *Proc. ICCV*, 2019. 34, 106, 137, 138

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., and Shah, R. Signature verification using a siamese time delay neural network. In *Proc. NIPS*, 1994. 25

Brox, T. and Malik, J. Large displacement optical flow: Descriptor matching in variational motion estimation. *IEEE PAMI*, 33(3):500–513, 2010. 107

Chen, J., Wang, L., Li, X., and Fang, Y. Arbicon-net: Arbitrary continuous geometric transformation networks for image registration. In *Proc. NeurIPS*, 2019. 162

Choy, C., Gwak, J., and Savarese, S. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In *Proc. CVPR*, 2019a. 139, 147

Choy, C., Park, J., and Koltun, V. Fully convolutional geometric features. In *Proc. ICCV*, 2019b. 139

Choy, C. B., Gwak, J., Savarese, S., and Chandraker, M. Universal correspondence network. In *Proc. NIPS*, 2016. 105

Chum, O., Matas, J., and Kittler, J. Locally optimized RANSAC. In *Proc. DAGM Symposium*, 2003. 33, 121

Chum, O., Werner, T., and Matas, J. Two-view geometry estimation unaffected by a dominant plane. In *Proc. CVPR*, 2005. 33, 121

Cieslewski, T., Bloesch, M., and Scaramuzza, D. Matching features without descriptors: Implicitly matched interest points. In *Proc. BMVC.*, 2018. 24

Dalal, N. and Triggs, B. Histogram of Oriented Gradients for Human Detection. In *Proc. CVPR*, 2005. 35, 38, 46

Dale, K., Johnson, M., Sunkavalli, K., Matusik, W., and Pfister, H. Image restoration using online photo collections. In *Proc. CVPR*, 2017. 82

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In *Proc. CVPR*, 2009. 50

DeTone, D., Malisiewicz, T., and Rabinovich, A. Deep image homography estimation. *arXiv preprint arXiv:1606.03798*, 2016. 48, 53, 56, 76

DeTone, D., Malisiewicz, T., and Rabinovich, A. SuperPoint: Self-Supervised Interest Point Detection and Description. In *CVPR Workshops*, 2018. 28, 29, 30, 123, 136, 138, 149, 150, 151, 152, 156, 164

Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., and Brox, T. FlowNet: Learning optical flow with convolutional networks. In *Proc. ICCV*, 2015. 107

Duchenne, O., Joulin, A., and Ponce, J. A graph-matching kernel for object categorization. In *Proc. ICCV*, 2011. 36, 37, 49, 65, 66, 71

Dusmanu, M., Rocco, I., Pajdla, T., Pollefeys, M., Sivic, J., Torii, A., and Sattler, T. D2-Net: A Trainable CNN for Joint Detection and Description of Local Features. In *Proc. CVPR*, 2019. 23, 28, 29, 30, 121, 123, 134, 136, 137, 138, 148, 149, 150, 151, 152, 156, 164

Edelman, S., Intrator, N., and Poggio, T. Complex cells and object recognition. In *Proc. NIPS*, 1997. 22, 23

Engel, J., Schöps, T., and Cremers, D. LSD-SLAM: Large-scale direct monocular SLAM. In *Proc. ECCV*, 2014. 9, 82

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The PASCAL Visual Object Classes Challenge 2011 (VOC2011) Results. http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html, 2011. 77

Faugeras, O. D., Luong, Q.-T., and Maybank, S. J. Camera self-calibration: Theory and experiments. In *Proc. ECCV*, pages 321–334, 1992. 21

Fei-Fei, L., Fergus, R., and Perona, P. One-shot learning of object categories. *IEEE PAMI*, 28(4):594–611, 2006. 63, 68, 93, 117

Fischer, P., Dosovitskiy, A., and Brox, T. Descriptor matching with convolutional neural networks: a comparison to SIFT. *arXiv preprint arXiv:1405.5769*, 2014. 102, 104, 105

Fischer, P., Dosovitskiy, A., Ilg, E., Häusser, P., Hazırbaş, C., Golkov, V., van der Smagt, P., Cremers, D., and Brox, T. FlowNet: Learning optical flow with convolutional networks. In *Proc. ICCV*, 2015. 48, 53, 55, 76, 82

Fischler, M. A. and Bolles, R. C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 33, 46, 54, 81, 88, 91, 97, 154, 155, 162

Forsyth, D. A. and Ponce, J. *Computer vision: a modern approach*. Prentice Hall Professional Technical Reference, 2002. 46

Gao, X.-S., Hou, X.-R., Tang, J., and Cheng, H.-F. Complete solution classification for the perspective-three-point problem. *IEEE PAMI*, 25(8):930–943, 2003. 134, 153, 154, 155

Germain, H., Bourmaud, G., and Lepetit, V. Sparse-to-Dense Hypercolumn Matching for Long-Term Visual Localization. In *3DV*, 2019. 31, 134, 137

Girshick, R. Fast R-CNN. In *Proc. ICCV*, 2015. 146

Girshick, R., Donahue, J., Darrell, T., and Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proc. CVPR*, 2014. 38, 41

Gojcic, Z., Zhou, C., Wegner, J. D., Guibas, L. J., and Birdal, T. Learning multiview 3D point cloud registration. In *Proc. CVPR*, 2020. 139

Gong, Y., Wang, L., Guo, R., and Lazebnik, S. Multi-scale orderless pooling of deep convolutional activation features. In *Proc. ECCV*, 2014. 50

Grabner, A., Roth, P. M., and Lepetit, V. 3D Pose Estimation and 3D Model Retrieval for Objects in the Wild. In *Proc. CVPR*, 2018. 9, 134

Graham, B. Sparse 3D convolutional neural networks. *arXiv preprint arXiv:1505.02890*, 2015. 139

Graham, B. Spatially-sparse convolutional neural networks. *arXiv preprint arXiv:1409.6070*, 2014. 139

Graham, B., Engelcke, M., and van der Maaten, L. 3D semantic segmentation with submanifold sparse convolutional networks. In *Proc. CVPR*, 2018. 135, 139, 144

HaCohen, Y., Shechtman, E., Goldman, D. B., and Lischinski, D. Non-rigid dense correspondence with applications for image enhancement. In *Proc. ACM SIGGRAPH*, 2011. 46

Ham, B., Cho, M., Schmid, C., and Ponce, J. Proposal Flow. In *Proc. CVPR*, 2016. 64, 71, 78, 79, 95

Ham, B., Cho, M., Schmid, C., and Ponce, J. Proposal flow: Semantic correspondences from object proposals. *IEEE PAMI*, 40(7):1711–1725, 2017. 36, 38, 39, 41, 46, 49, 63, 64, 65, 66, 67, 68, 69, 84, 92, 93, 96, 97, 117, 118, 119

Han, K., Rezende, R. S., Ham, B., Wong, K.-Y. K., Cho, M., Schmid, C., and Ponce, J. SCNet: Learning semantic correspondence. In *Proc. ICCV*, 2017. 41, 42, 83, 84, 85, 92, 93, 94, 95, 96, 117, 118, 119

Han, X., Hu, X., Huang, W., and Scott, M. R. ClothFlow: A flow-based model for clothed person generation. In *Proc. ICCV*, 2019. 10

Han, X., Leung, T., Jia, Y., Sukthankar, R., and Berg, A. C. MatchNet: Unifying feature and metric learning for patch-based matching. In *Proc. CVPR*, 2015. 26, 48, 102, 105, 134, 136

Harris, C. and Stephens, M. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50, 1988. 18, 19, 20, 28, 48

Hartley, R. and Zisserman, A. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 46, 56, 62, 82

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proc. CVPR*, 2016. 91, 128

He, K., Gkioxari, G., Dollár, P., and Girshick, R. Mask R-CNN. In *Proc. ICCV*, 2017. 165

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 163

Hirschmüller, H. Stereo processing by semiglobal matching and mutual information. *IEEE PAMI*, 30(2):328–341, 2007. 107

Horn, B. K. and Schunck, B. G. Determining optical flow. *Artificial Intelligence*, 17 (1-3):185–203, 1981. 107

Hough, P. V. Method and means for recognizing complex patterns, 1962. US Patent 3,069,654. 27, 46, 54

Ioffe, S. and Szegedy, C. Batch Normalization: Accelerating deep network training by reducing internal covariate shift. In *icml*, 2015. 54

Ishikawa, H. Exact optimization for Markov random fields with convex priors. *IEEE PAMI*, 25(10):1333–1336, 2003. 37

Jaderberg, M., Simonyan, K., Zisserman, A., and Kavukcuoglu, K. Spatial Transformer Networks. In *Proc. NIPS*, 2015. 58, 90

Jahrer, M., Grabner, M., and Bischof, H. Learned local descriptors for recognition and matching. In *Computer Vision Winter Workshop*, 2008. 24, 26, 48, 102, 104, 105

Jeon, S., Kim, S., Min, D., and Sohn, K. Parn: Pyramidal affine regression networks for dense semantic correspondence. In *Proc. ECCV*, 2018. 39, 42, 161

Johnson, J., Douze, M., and Jégou, H. Billion-scale similarity search with GPUs. *arXiv preprint arXiv:1702.08734*, 2017. 147

Julesz, B. Towards the automation of binocular depth perception. In *Proc. IFIP Congress' 62*, pages 439–444, 1962. 133

Kanazawa, A., Jacobs, D. W., and Chandraker, M. WarpNet: Weakly supervised matching for single-view reconstruction. In *Proc. CVPR*, 2016. 39, 40, 48, 53, 76, 83

Kanazawa, A., Tulsiani, S., Efros, A. A., and Malik, J. Learning category-specific mesh reconstruction from image collections. In *Proc. ECCV*, 2018. 131

Kendall, A., Martirosyan, H., Dasgupta, S., Henry, P., Kennedy, R., Bachrach, A., and Bry, A. End-to-end learning of geometry and context for deep stereo regression. In *Proc. ICCV*, 2017. 107

Kim, J., Liu, C., Sha, F., and Grauman, K. Deformable spatial pyramid matching for fast dense correspondences. In *Proc. CVPR*, 2013. 36, 37, 42, 49, 65, 66, 68, 69, 70, 71, 84, 93

Kim, S., Min, D., Lin, S., and Sohn, K. DCTM: Discrete-continuous transformation matching for semantic flow. In *Proc. ICCV*, 2017. 41, 83, 84, 85, 97

Kim, S., Lin, S., Jeon, S., Min, D., and Sohn, K. Recurrent transformer networks for semantic correspondence. In *Proc. NeurIPS*, 2018a. 39, 42, 105, 117, 119, 161

Kim, S., Min, D., Ham, B., Jeon, S., Lin, S., and Sohn, K. FCSS: Fully convolutional self-similarity for dense semantic correspondence. *IEEE PAMI*, pages 581–595, 2018b. 10, 39, 40, 42, 43, 84, 85, 93, 96, 97

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *ICLR*, 2015. 62, 92, 129, 147

Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet classification with deep convolutional neural networks. In *Proc. NIPS*, 2012. 38, 46

Laguna, A. B., Riba, E., Ponsa, D., and Mikolajczyk, K. Key.Net: Keypoint detection by handcrafted and learned CNN filters. In *Proc. ICCV*, 2019. 26, 27, 28, 134, 136

Lamdan, Y., Schwartz, J. T., and Wolfson, H. J. Object recognition by affine invariant matching. In *Proc. CVPR*, 1988. 46, 54

Lasinger, K., Ranftl, R., Schindler, K., and Koltun, V. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *arXiv preprint arXiv:1907.01341*, 2019. 165

Laskar, Z., Melekhov, I., Tavakoli, H. R., Ylioinas, J., and Kannala, J. Geometric image correspondence verification by dense pixel matching. In *Proc. WACV*, 2020. 162

Lazebnik, S., Schmid, C., and Ponce, J. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. CVPR*, 2006. 37

Learned-Miller, E. G. Data driven image models through continuous joint alignment. *IEEE PAMI*, 2006. 49

Lebeda, K., Matas, J., and Chum, O. Fixing the locally optimized RANSAC. In *Proc. BMVC.*, 2012. 33, 73

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 24

Lee, J., Kim, D., Ponce, J., and Ham, B. Sfnet: Learning object-aware semantic correspondence. In *Proc. CVPR*, 2019. 43, 162

Leibe, B., Leonardis, A., and Schiele, B. Robust object detection with interleaved categorization and segmentation. *IJCV*, 2008. 46, 54

Lenc, K. and Vedaldi, A. Learning covariant feature detectors. In *ECCV Workshop on Geometry Meets Deep Learning*, 2016. 26, 27, 134, 136

Lepetit, V., Lagger, P., and Fua, P. Randomized trees for real-time keypoint recognition. In *Proc. CVPR*, 2005. 24

Leutenegger, S., Chli, M., and Siegwart, R. Y. Brisk: Binary robust invariant scalable keypoints. In *Proc. ICCV*, 2011. 23

Li, S., Han, K., Costain, T. W., Howard-Jenkins, H., and Prisacariu, V. Correspondence networks with adaptive neighbourhood consensus. In *Proc. CVPR*, 2020. 162

Lindeberg, T. Feature detection with automatic scale selection. *IJCV*, 30(2):79–116, 1998. 19, 20

Liu, C., Yuen, J., and Torralba, A. SIFT Flow: Dense correspondence across scenes and its applications. *IEEE PAMI*, 2011. 35, 36, 37, 49, 65, 66, 68, 71, 82, 84, 102

Long, J., Zhang, N., and Darrell, T. Do convnets learn correspondence? In *Proc. NIPS*, 2014. 38, 49, 105

Lowe, D. G. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2): 91–110, 2004. 20, 22, 23, 32, 35, 46, 48, 51, 52, 54, 58, 73, 76, 87, 102, 104, 134, 136, 137, 138, 156

Lucas, B. D. and Kanade, T. An iterative image registration technique with an application to stereo vision. In *Proc. IJCAI*, 1981. 107

Luo, X., Huang, J.-B., Szeliski, R., Matzen, K., and Kopf, J. Consistent video depth estimation. In *Proc. ACM SIGGRAPH*, 2020. 165

Ma, J., Zhao, J., Jiang, J., Zhou, H., and Guo, X. Locality preserving matching. *IJCV*, 127(5):512–531, 2019. 106, 107, 123

Maimone, M., Cheng, Y., and Matthies, L. Two years of visual odometry on the mars exploration rovers. *Journal of Field Robotics*, 24(3):169–186, 2007. 9

Marr, D. and Poggio, T. Cooperative computation of stereo disparity. *Science*, 194 (4262):283–287, 1976. 133

Melekhov, I., Tiulpin, A., Sattler, T., Pollefeys, M., Rahtu, E., and Kannala, J. Dgc-net: Dense geometric correspondence network. In *Proc. WACV*, 2019. 162

Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G., et al. Mixed precision training. *arXiv preprint arXiv:1710.03740*, 2017. 164

Mikolajczyk, K. *Interest point detection invariant to affine transformations*. PhD thesis, Institut National Polytechnique de Grenoble, 2002. 20

Mikolajczyk, K. and Schmid, C. An affine invariant interest point detector. In *Proc. ECCV*, 2002. 48, 63, 72, 73, 74, 104, 134, 150

Mikolajczyk, K. and Schmid, C. Scale & affine invariant interest point detectors. *IJCV*, 60(1):63–86, 2004. 19, 28

Mikolajczyk, K. and Schmid, C. A performance evaluation of local descriptors. *IEEE PAMI*, 27(10):1615–1630, 2005. 121

Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., and Van Gool, L. A comparison of affine region detectors. *IJCV*, 65 (1-2):43–72, 2005. 134

Mishchuk, A., Mishkin, D., Radenovic, F., and Matas, J. Working hard to know your neighbor's margins: Local descriptor learning loss. In *Proc. NIPS*, 2017. 25, 26, 123, 134, 136, 150, 156

Mishkin, D., Radenović, F., and Matas, J. Repeatability Is Not Enough: Learning Discriminative Affine Regions via Discriminability. In *Proc. ECCV*, 2018. 26, 28, 123, 134, 136, 150, 156

Moravec, H. P. *Obstacle avoidance and navigation in the real world by a seeing robot rover.* PhD thesis, Stanford University, 1980. 18

Mori, K.-I., Kidode, M., and Asada, H. An iterative prediction and correction method for automatic stereocomparison. *Computer Graphics and Image Processing*, 2(3-4): 393–401, 1973. 133

Newcombe, R. A., Lovegrove, S. J., and Davison, A. J. DTAM: Dense tracking and mapping in real-time. In *Proc. ICCV*, 2011. 82

Nikandrova, E. and Kyrki, V. Category-based task specific grasping. *Robotics and Autonomous Systems*, 70(Supplement C):25–35, 2015. 82

Noh, H., Araujo, A., Sim, J., Weyand, T., and Han, B. Large-scale image retrieval with attentive deep local features. In *Proc. ICCV*, 2017. 31, 48, 104, 105, 123, 134, 137, 150, 156

Novotny, D., Larlus, D., and Vedaldi, A. AnchorNet: A weakly supervised network to learn geometry-sensitive features for semantic matching. In *Proc. CVPR*, 2017. 10, 39, 41, 42, 83, 84, 85, 165

Novotny, D., Albanie, S., Larlus, D., and Vedaldi, A. Self-supervised learning of geometrically stable features through probabilistic introspection. In *Proc. CVPR*, 2018. 42, 161

Ono, Y., Trulls, E., Fua, P., and Yi, K. M. LF-Net: Learning local features from images. In *Proc. NIPS*, 2018. 28, 30, 136

Oron, S., Dekel, T., Xue, T., Freeman, W. T., and Avidan, S. Best-buddies similarity—robust template matching using mutual nearest neighbors. *IEEE PAMI*, 40(8): 1799–1813, 2017. 137

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in PyTorch, 2017. 62, 92, 128, 147

Persson, M. and Nordberg, K. Lambda twist: An accurate fast robust perspective three point (P3P) solver. In *Proc. ECCV*, 2018. 134

Philbin, J., Chum, O., Isard, M., Sivic, J., and Zisserman, A. Object retrieval with large vocabularies and fast spatial matching. In *Proc. CVPR*, 2007. 49, 58

Revaud, J., Weinzaepfel, P., Harchaoui, Z., and Schmid, C. DeepMatching: Hierarchical deformable dense matching. *IJCV*, 2015. 65, 66, 71

Revaud, J., Weinzaepfel, P., de Souza, C. R., and Humenberger, M. R2D2: Repeatable and reliable detector and descriptor. In *Proc. NeurIPS*, 2019. 28, 30, 136, 150, 151, 152, 156, 163, 164, 165

Rocco, I., Arandjelović, R., and Sivic, J. Convolutional neural network architecture for geometric matching. In *Proc. CVPR*, 2017. 16, 83, 84, 85, 91, 92, 93, 95, 96, 97, 105, 117, 119

Rocco, I., Arandjelović, R., and Sivic, J. End-to-end weakly-supervised semantic alignment. In *Proc. CVPR*, 2018a. 16, 79, 105, 117, 119

Rocco, I., Arandjelović, R., and Sivic, J. Convolutional neural network architecture for geometric matching. *IEEE PAMI*, 2018b. 16

Rocco, I., Cimpoi, M., Arandjelović, R., Torii, A., Pajdla, T., and Sivic, J. Neighbourhood consensus networks. In *Proc. NeurIPS*, 2018c. 16, 122, 134, 138

Rocco, I., Arandjelović, R., and Sivic, J. Efficient neighbourhood consensus networks via submanifold sparse convolutions. In *Proc. ECCV*, 2020. 16

Rocco, I., Cimpoi, M., Arandjelović, R., Torii, A., Pajdla, T., and Sivic, J. NCNet: Neighbourhood consensus networks for estimating image correspondences. *IEEE PAMI*, Accepted. 16

Rubinstein, M., Joulin, A., Kopf, J., and Liu, C. Unsupervised joint object discovery and segmentation in internet images. In *Proc. CVPR*, 2013. 46

Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. ORB: An efficient alternative to SIFT or SURF. In *Proc. ICCV*, 2011. 23, 134

Sakurada, K. and Okatani, T. Change Detection from a Street Image Pair using CNN Features and Superpixel Segmentation. In *Proc. BMVC.*, 2015. 9

Sarlin, P.-E., DeTone, D., Malisiewicz, T., and Rabinovich, A. SuperGlue: Learning feature matching with graph neural networks. In *Proc. CVPR*, 2019. 34, 138, 164

Sattler, T., Leibe, B., and Kobbelt, L. SCRAMSAC: Improving RANSAC's Efficiency with a Spatial Consistency Filter. In *Proc. ICCV*, 2009. 33, 34, 106

Sattler, T., Maddern, W., Toft, C., Torii, A., Hammarstrand, L., Stenborg, E., Safari, D., Okutomi, M., Pollefeys, M., Sivic, J., et al. Benchmarking 6DOF outdoor visual localization in changing conditions. In *Proc. CVPR*, 2018. 9, 23, 31, 102, 131, 134, 136, 137, 156

Savinov, N., Ladicky, L., and Pollefeys, M. Matching neural paths: Transfer from recognition to correspondence search. In *Proc. NIPS*, 2017. 104

Schaffalitzky, F. and Zisserman, A. Automated scene matching in movies. In *International Conference on Image and Video Retrieval*, 2002a. 34, 103, 106, 140

Schaffalitzky, F. and Zisserman, A. Multi-view matching for unordered image sets, or "How do i organize my holiday snaps?". In *Proc. ECCV*, 2002b. 19

Schmid, C. *Appariement d'images par invariants locaux de niveaux de gris. Application à l'indexation d'une base d'objets*. PhD thesis, Institut National Polytechnique de Grenoble - INPG, 1996. 32

Schmid, C. and Mohr, R. Local grayvalue invariants for image retrieval. *IEEE PAMI*, 19(5):530–535, 1997. 21, 22, 23, 31, 34, 46, 48, 54, 103, 106, 140

Schonberger, J. L., Hardmeier, H., Sattler, T., and Pollefeys, M. Comparative evaluation of hand-crafted and learned local features. In *Proc. CVPR*, 2017. 102

Schönberger, J. L. and Frahm, J.-M. Structure-from-motion revisited. In *Proc. CVPR*, 2016. 134, 156

Schönberger, J. L., Zheng, E., Pollefeys, M., and Frahm, J.-M. Pixelwise view selection for unstructured multi-view stereo. In *Proc. ECCV*, 2016. 134

Shokrollahi Yancheshmeh, F., Chen, K., and Kamarainen, J.-K. Unsupervised visual alignment with similarity graphs. In *Proc. CVPR*, 2015. 49

Simo-Serra, E., Trulls, E., Ferraz, L., Kokkinos, I., Fua, P., and Moreno-Noguer, F. Discriminative learning of deep convolutional feature point descriptors. In *Proc. ICCV*, 2015. 25, 26, 48, 102, 104, 105

Simonyan, K. and Zisserman, A. Very deep convolutional networks for large-scale image recognition. In *Proc. ICLR*, 2015. 46, 50, 91

Simonyan, K., Vedaldi, A., and Zisserman, A. Learning local feature descriptors using convex optimisation. *IEEE PAMI*, 36(8):1573–1585, 2014. 24, 26, 102, 104, 105

Sivic, J. and Zisserman, A. Video Google: A text retrieval approach to object matching in videos. In *Proc. ICCV*, 2003. 34, 46, 54, 103, 106, 140

Snavely, N., Seitz, S. M., and Szeliski, R. Photo tourism: Exploring photo collections in 3D. *ACM transactions on graphics (TOG)*, 25(3):835–846, 2006. 9

Sun, D., Roth, S., and Black, M. J. Secrets of optical flow estimation and their principles. In *Proc. CVPR*, 2010. 107

Sun, D., Yang, X., Liu, M.-Y., and Kautz, J. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *Proc. CVPR*, 2018. 107

Szeliski, R. Image alignment and stitching: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 2(1):1–104, 2006. 82

Taira, H., Okutomi, M., Sattler, T., Cimpoi, M., Pollefeys, M., Sivic, J., Pajdla, T., and Torii, A. InLoc: Indoor visual localization with dense matching and view synthesis. In *Proc. CVPR*, 2018. 31, 102, 119, 122, 126, 127, 134, 136, 137, 153

Taira, H., Rocco, I., Sedlar, J., Okutomi, M., Sivic, J., Pajdla, T., Sattler, T., and Torii, A. Is this the right place? Geometric-semantic pose verification for indoor visual localization. In *Proc. ICCV*, 2019. 166

Taniai, T., Sinha, S. N., and Sato, Y. Joint recovery of dense correspondence and cosegmentation in two images. In *Proc. CVPR*, 2016. 63, 66, 67, 68, 84, 93, 97, 118

Thacker, N. A. and Courtney, P. Statistical analysis of a stereo matching algorithm. In *Proc. BMVC.*, 1992. 21

Thewlis, J., Zheng, S., Torr, P., and Vedaldi, A. Fully-trainable deep matching. In *Proc. BMVC.*, 2016. 48

Tian, Y., Fan, B., and Wu, F. L2-net: Deep learning of discriminative patch descriptor in euclidean space. In *Proc. CVPR*, 2017. 25, 26, 134, 136

Torii, A., Arandjelović, R., Sivic, J., Okutomi, M., and Pajdla, T. 24/7 place recognition by view synthesis. In *Proc. CVPR*, 2015. 31, 134, 137

Truong, P., Danelljan, M., and Timofte, R. GLU-Net: Global-local universal network for dense flow and correspondences. In *Proc. CVPR*, 2020. 162, 166

Tuytelaars, T. and Mikolajczyk, K. Local invariant feature detectors: A survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3):177–280, 2008. 104

Ufer, N. and Ommer, B. Deep semantic feature matching. In *Proc. CVPR*, 2017. 38, 39, 40, 84

Vedaldi, A. and Fulkerson, B. VLFeat: An open and portable library of computer vision algorithms. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1469–1472, 2010. 22

Verdie, Y., Yi, K., Fua, P., and Lepetit, V. TILDE: A temporally invariant learned detector. In *Proc. CVPR*, 2015. 26, 27, 134, 136

Wang, X., Jabri, A., and Efros, A. A. Learning correspondence from the cycle-consistency of time. In *Proc. CVPR*, 2019. 10, 162

Weinzaepfel, P., Revaud, J., Harchaoui, Z., and Schmid, C. DeepFlow: Large displacement optical flow with deep matching. In *Proc. ICCV*, 2013. 48, 53, 82

Widya, A. R., Torii, A., and Okutomi, M. Structure from motion using dense cnn features with keypoint relocalization. *IPSJ Transactions on Computer Vision and Applications*, 10(1):6, 2018. 31, 116, 134, 137, 138, 139, 164

Winder, S. A. and Brown, M. Learning local image descriptors. In *Proc. CVPR*, 2007. 23, 24, 26

Yang, F., Li, X., Cheng, H., Li, J., and Chen, L. Object-aware dense semantic correspondence. In *Proc. CVPR*, 2017. 84, 96, 97

Yang, G. and Ramanan, D. Volumetric correspondence networks for optical flow. In *Proc. NeurIPS*, 2019. 162

Yang, Y. and Ramanan, D. Articulated human detection with flexible mixtures of parts. *IEEE PAMI*, 2013. 65, 92

Yi, K. M., Trulls, E., Lepetit, V., and Fua, P. LIFT: Learned invariant feature transform. In *Proc. ECCV*, 2016. 28, 29, 30, 102, 105, 136, 147

Yi, K. M., Trulls, E., Ono, Y., Lepetit, V., Salzmann, M., and Fua, P. Learning to find good correspondences. In *Proc. CVPR*, 2018. 34, 106, 107, 123, 137, 138

Yu, G. and Morel, J.-M. ASIFT: An Algorithm for Fully Affine Invariant Comparison. *Image Processing On Line*, 2011. doi: 10.5201/ipol.2011.my-asift. 73, 74

Zagoruyko, S. and Komodakis, N. Learning to compare image patches via convolutional neural networks. In *Proc. CVPR*, 2015. 25, 26, 48, 102, 105, 134, 136

Zamir, A. R., Sax, A., Shen, W., Guibas, L. J., Malik, J., and Savarese, S. Taskonomy: Disentangling task transfer learning. In *Proc. CVPR*, 2018. 165

Zeiler, M. D. and Fergus, R. Visualizing and understanding convolutional networks. In *Proc. ECCV*, 2014. 110

Zhang, J., Sun, D., Luo, Z., Yao, A., Zhou, L., Shen, T., Chen, Y., Quan, L., and Liao, H. Learning two-view correspondences and geometry using order-aware network. *Proc. ICCV*, 2019. 34, 106, 138

Zhang, Z., Deriche, R., Faugeras, O., and Luong, Q.-T. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial intelligence*, 78(1-2):87–119, 1995. 34, 103, 106, 140

Zhao, W.-L., Jégou, H., and Gravier, G. Oriented pooling for dense and non-dense rotation-invariant features. In *Proc. BMVC.*, 2013. 137

Zhou, H., Sattler, T., and Jacobs, D. W. Evaluating local features for day-night matching. In *Proc. ECCV*, 2016a. 134

Zhou, T., Lee, Y. J., Yu, S. X., and Efros, A. A. FlowWeb: Joint image set alignment by weaving consistent, pixel-wise correspondences. In *Proc. CVPR*, 2015. 49

Zhou, T., Krähenbühl, P., Aubry, M., Huang, Q., and Efros, A. A. Learning dense correspondence via 3d-guided cycle consistency. In *Proc. CVPR*, 2016b. 80

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networkss. In *Proc. ICCV*, 2017. 80

RÉSUMÉ

L'objectif de cette thèse est de développer des méthodes pour la mise en correspondance entre de paires d'images dans des situations difficiles, telles que des changements extrêmes d'éclairage, des scènes avec peu de texture ou comprenant des structures répétitives, ou la mise en correspondance entre parties d'objets qui appartiennent à la même classe mais qui peuvent présenter de grandes différences d'apparence intra-classe. Nos contributions sont les suivantes : (i) nous développons une approche entraînable pour l'alignement paramétrique d'images en utilisant un modèle de réseau siamois, (ii) nous concevons une approche d'entraînement faiblement supervisée, qui permet l'entraînement à partir de paires d'images réelles annotées seulement au niveau des paires d'images, (iii) nous proposons les Réseaux de Consensus de Voisinage qui peuvent être utilisés pour estimer de manière robuste les correspondances pour des tâches où des correspondances discrètes sont requises et (iv) nous développons une variante plus efficace qui peut réduire les besoins en mémoire et le temps d'exécution des Réseaux de Consensus de Voisinage par un facteur dix.

MOTS CLÉS

Vision par ordinateur, estimation de correspondances, apprentissage profond, apprentissage faiblement supérvisé.

ABSTRACT

The goal of this thesis is to develop methods for establishing correspondences between pairs of images in challenging situations, such as extreme illumination changes, scenes with little texture or with repetitive structures, and matching parts of objects which belong to the same class, but which may have large intra-class appearance differences. In summary, our contributions are the following: (i) we develop a trainable approach for parametric image alignment by means of a siamese network model, (ii) we devise a weakly-supervised training approach, which allow training from real image pairs having only annotation at the level of image-pairs, (iii) we propose the Neighbourhood Consensus Networks which can be used to robustly estimate correspondences in tasks where discrete correspondences are required, and (iv) because the dense formulation of the Neighbourhood Consensus Networks is memory and computationally intensive, we develop a more efficient variant that can reduce the memory requirements and run-time by more than ten times.

KEYWORDS

Computer vision, correspondence estimation, deep Learning, weakly-supervised learning.