



CENTER FOR
MACHINE PERCEPTION



CZECH TECHNICAL
UNIVERSITY IN PRAGUE

PhD Thesis

Learning and Crafting for the Wide Multiple Baseline Stereo

A Doctoral Thesis presented to the Faculty of the Electrical Engineering of the Czech Technical University in Prague in fulfillment of the requirements for the Ph.D. Degree in Study Programme No. P2612 - Electrical Engineering and Information Technology, Branch No. 3902V035 - Artificial Intelligence and Biocybernetics, by

Dmytro Mishkin

mishkdmy@cmp.felk.cvut.cz

Thesis Advisor: prof. Jiří Matas

March 4, 2021

Available at

http://cmp.felk.cvut.cz/~mishkdmy/dissertation_DM.pdf

Supported by Czech Science Foundation Project GACR P103/12/G084, Austrian Ministry for Transport, Innovation and Technology, the Federal Ministry of Science, Research and Economy, and the Province of Upper Austria in the frame of the COMET center SCCH, the CTU student grant SGS17/185/OHK3/3T/13, and OP VVV funded project CZ.02.1.01/0.0/0.0/16 019/0000765 "Research Center for Informatics"

Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University
Technická 2, 166 27 Prague 6, Czech Republic
fax +420 2 2435 7385, phone +420 2 2435 7637, www: <http://cmp.felk.cvut.cz>

Abstract

This thesis introduces the wide multiple baseline stereo (WxBS) problem. WxBS, a generalization of the standard wide baseline stereo problem, considers the matching of images that simultaneously differ in more than one image acquisition factor such as viewpoint, illumination, sensor type, or where object appearance changes significantly, e.g., over time. A new dataset with the ground truth, evaluation metric and baselines has been introduced.

The thesis presents the following improvements of the WxBS pipeline. (i) A loss function, called HardNeg, for learning a local image descriptor that relies on hard negative mining within a mini-batch and on the maximization of the distance between the closest positive and the closest negative patches. (ii) The descriptor trained with the HardNeg loss, called HardNet, is compact and shows state-of-the-art performance in standard matching, patch verification and retrieval benchmarks. (iii) A method for learning the affine shape, orientation, and potentially other parameters related to geometric and appearance properties of local features. (iv) A tentative correspondences generation strategy which generalizes the standard first to second closest distance ratio is presented. The selection strategy, which shows performance superior to the standard method, is applicable to either hard-engineered descriptors like SIFT, LIOP, and MROGH or deeply learned like HardNet. (v) A feedback loop is introduced for the two-view matching problem, resulting in MODS – matching with on-demand view synthesis – algorithm. MODS is an algorithm that handles a viewing angle difference even larger than the previous state-of-the-art ASIFT algorithm, without a significant increase of computational cost over “standard” wide and narrow baseline approaches.

Last, but not least, a comprehensive benchmark for local features and robust estimation algorithms is introduced. The modular structure of its pipeline allows easy integration, configuration, and combination of methods and heuristics.

Abstrakt

Tato práce představuje problém wide multiple baseline stereo (WxBS), který je zobecněním standardního problému wide baseline stereo. WxBS se zabývá párováním obrázků, které se současně liší více než v jednom faktoru získání obrázku, jako je stanoviště a zorný úhel, osvětlení, typ senzoru, nebo kde se vzhled objektu zásadně mění, např. v průběhu času. Byl představen nový dataset s anotacemi, evaluační metrikou a výchozími metodami.

Disertační práce představuje následující vylepšení algoritmu WxBS: Ztrátovou funkci (i) pro učení lokálního deskriptoru obrázku založeného na výběru obtížných negativních příkladů v mini-batchi a na maximalizaci rozdílu vzdálenosti od nejbližšího pozitivního a nejbližšího negativního příkladu. Výsledný deskriptor (ii), nazvaný HardNet, který je kompaktní a dosahuje vynikajících výsledků v standardním párování, ověřování malých částí obrázků a vyhledávání obrázků. Metodu pro učení afinního tvaru, orientace a potenciálně dalších parametrů souvisejících s geometrií a vzhledem lokálních příznaků (iii). Strategii pro generování tentativních korespondencí (iv), která zobecňuje standardní poměr prvních dvou nejbližších vzdáleností. Strategie výběru, která dosahuje lepších výsledků než standardní metoda, je aplikovatelná jak pro manuálně navržené deskriptory jako SIFT, LIOP či MROGH, tak pro hluboce učené deskriptory jako HardNet. Zavedením zpětnovazebních smyček pro problém párování ze dvou pohledů vznikl algoritmus MODS – matching with on-demand view synthesis (v). Algoritmus MODS zvládá i větší rozdíly v úhlu pohledu než ASIFT algoritmus, a to bez výrazného navýšení výpočetní ceny proti standardním wide-baseline a narrow-baseline přístupům.

V neposlední řadě je představen komplexní benchmark pro porovnávání lokálních příznaků a algoritmů pro robustní odhady (vi). Jeho modulární struktura umožňuje jednoduchou integraci, konfiguraci a kombinaci metod a heuristik.

Анотація

У даній дисертації досліджується задача стереозору з широкою мультибазою (англ. Wide Multiple Baseline Stereo, WxBS), яка є узагальненням стандартної задачі стереозору із широкою базою (англ. Wide Baseline Stereo, WBS). Задача WxBS розглядає співставлення зображень однієї сцени, які одночасно відрізняються більш ніж в одному аспекті. Такими аспектами можуть бути позиція камери, освітленість, тип датчика або істотна зміна зовнішнього вигляду об'єкта, наприклад, з часом. Для тестування алгоритмів розв'язування задачі WxBS представлено новий датасет із еталонними даними та метриками.

У дисертації запропоновано такі вдосконалення алгоритму для розв'язання задачі стереозору з широкою мультибазою. Розроблено функцію втрат (i) для тренування локального дескриптора, яка покладається на пошук найскладнішого негативного прикладу у міні-батчі та максимізації відстані між найближчим позитивним та найближчим негативним фрагментами зображення. Отриманий дескриптор (ii), який одержав назву HardNet, є компактним та демонструє найкращі результати у тестах на порівняння локальних дескрипторів. Запропоновано метод навчання детекторів еліптичної форми, орієнтації та потенційно інших параметрів локальних ознак зображення (iii). Удосконалено стратегію генерації початкових відповідностей (iv) шляхом узагальнення стандартного співвідношення першої та другої найближчих відстаней дескрипторів. Нова стратегія генерації покращує якість отриманих відповідностей порівняно зі стандартним методом та може застосовуватися для будь-якого типу дескрипторів, від спроектованих вручну, таких як SIFT, LIOP і MROGH, до навчених за допомогою глибокого навчання, таких як HardNet. Запропоновано включення зворотного зв'язку до алгоритму співставлення двох зображень однієї сцени, що дозволило побудувати алгоритм співставлення шляхом генерування синтетичних зображень на вимогу (v), який одержав назву MODS (від англ. Matching with On-Demand View Synthesis). MODS дозволяє знаходити відповідності на зображеннях, одержаних у ширшому діапазоні різниць в позиціях камер, ніж попередній найкращий алгоритм розв'язання даної задачі, ASIFT. Окрім того, на відміну від ASIFT, MODS спроможний швидко опрацьовувати пари зображень невисокої складності, наприклад, отриманих з камер, які розташовані близько одна до одної.

Також у дисертації представлено бенчмарк для детекторів та дескрипторів локальних ознак, алгоритмів стійкої оцінки геометрії сцени. Його модульна структура дозволяє легко додавати, налаштовувати, поєднувати та порівнювати різноманітні методи та евристики.

Acknowledgements

Pursuing a PhD has been the longest and hardest endeavour in my life so far. It started with an internship at Center for Machine Perception in 2012 – 2013. That internship literally changed my life and not least because of the wonderful people I met and admired since then. Jan, Michal, Kostya, Sasha, Nataliya, Hongping, Matej, Jana, Javier, Jimmy, Michal, Milan, Filip – our lunches together will always be in my heart. My first close supervisor – Michal Perdoch spent a lot of time answering my stupid questions about callbacks, coding, and geometry.

Professor Jiří Matas – was and is much more than just a supervisor to me. He always has time for help and advice – be it the deadline paper writing session at midnight, fight for the open-science on social media, discussion about scientific reforms, or not kicking me out for leaving the note "I am going for Euromaidan tomorrow" on his table and disappearing to Kyiv. He – together with all other people of the CMP – have shown me a bright side of academia – free, supportive, and inspiring. It is a blessing to work with you.

Eva Matyskova – thank you for making Prague feel like a home and treating us – PhD students – as your kids in the best sense of this word. My life in Czech Republic would be much more difficult without your help. Sergey Shelpuk – your talk in Odesa, 2014 about deep learning was pivotal for me. Thank you.

My co-authors – Michal, Karel, Anastasiia, Filip, Edgar, Eduard, Milan, Milan, Kwang, Yuhe, Javier, Alexey, Nikolaj, Vladlen, Orest, Mykola, Volodymyr, Gary, Daniel, Daniel, Ivan, Ethan, Iliia, Amy – it was a pleasure to work together and I hope to do it once again.

Amy Tabb, James Pritts, and Karel Lenc – thank you for reading the early drafts of this thesis and helping to make it better.

Finally, nothing above would happen without the endless support of my family – my parents Sergiy and Tetyana, my wife Olya, and my daughter Sofia. Whatever I do, you believe in me. Hugs :)

Contents

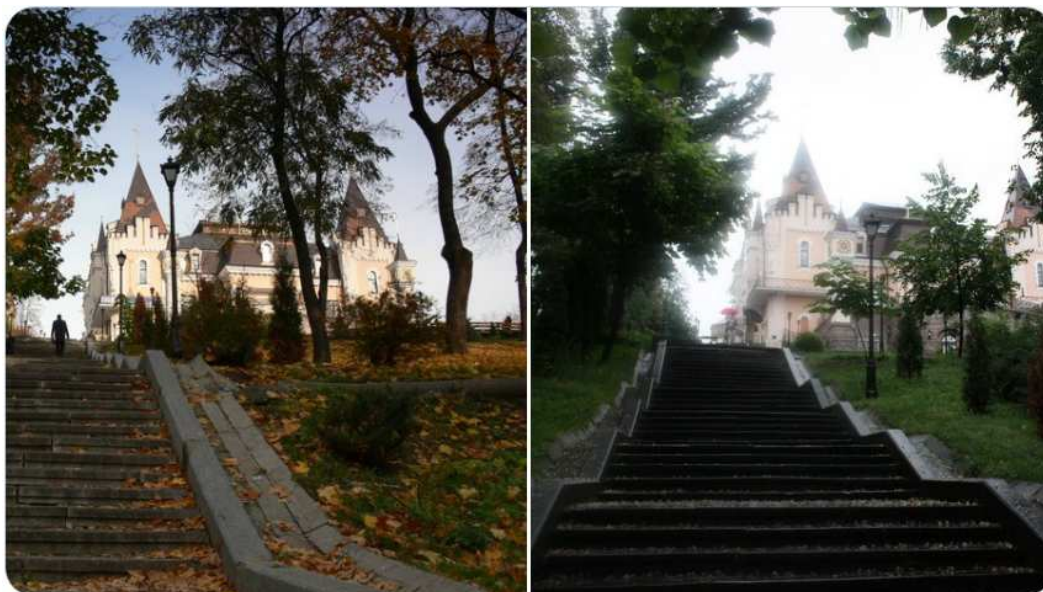
| | | |
|----------|--|-----------|
| 1 | Introduction | 13 |
| 1.1 | What is wide multiple baseline stereo? | 13 |
| 1.2 | Areas related to the wide multiple baseline stereo | 14 |
| 1.3 | How to solve the wide multiple baseline stereo problem? | 14 |
| 1.4 | History of the wide multiple baseline stereo and its role in the deep learning world | 16 |
| 1.5 | Contributions | 20 |
| 1.6 | Publications | 20 |
| 2 | WxBS: Wide multiple baseline stereo problem | 23 |
| 2.1 | Definition | 23 |
| 2.2 | Wide multiple baseline stereo: evaluation | 24 |
| 2.3 | Evaluation of local descriptors on single baseline datasets | 26 |
| 3 | HardNet local feature descriptor | 29 |
| 3.1 | Related work | 29 |
| 3.2 | The HardNet descriptor | 30 |
| 3.2.1 | Sampling and loss | 30 |
| 3.2.2 | HardNet architecture | 31 |
| 3.2.3 | Model training | 31 |
| 3.3 | Empirical evaluation | 32 |
| 3.3.1 | Patch descriptor evaluation | 32 |
| 3.3.2 | Ablation study | 33 |
| 3.3.3 | Wide baseline stereo | 34 |
| 3.3.4 | Image retrieval | 35 |
| 3.4 | Exploring HardNet design choices [199]. | 37 |
| 3.4.1 | Architecture | 37 |
| 3.4.4 | Batch size | 39 |
| 3.4.5 | Margin value | 40 |
| 3.4.7 | Compression of Embeddings | 41 |
| 3.4.8 | Combining all together | 42 |
| 3.5 | Summary | 43 |
| 4 | Local affine features | 45 |
| 4.1 | Introduction | 45 |
| 4.2 | Keypoints are not just points | 46 |
| 4.3 | Benefits of local affine features | 47 |
| 4.3.1 | Making descriptor job easier | 47 |
| 4.3.2 | Making RANSAC job easier | 47 |
| 4.3.3 | Application-specific benefits | 49 |
| 4.3.4 | Related work | 49 |
| 4.4 | Learning affine shape and orientation | 51 |
| 4.4.1 | Affine shape parametrization | 51 |
| 4.4.2 | Hard negative-constant loss | 51 |
| 4.4.3 | Descriptor losses for shape registration | 52 |
| 4.4.4 | AffNet training procedure | 53 |
| 4.4.5 | Training dataset and data preprocessing | 53 |
| 4.4.6 | Implementation details | 54 |

| | | |
|----------|---|-----------|
| 4.5 | Empirical evaluation | 55 |
| 4.5.1 | Loss functions and descriptors for learning measurement region | 55 |
| 4.5.2 | Repeatability | 56 |
| 4.5.3 | Wide baseline stereo | 57 |
| 4.5.4 | Image retrieval | 58 |
| 4.6 | Summary | 60 |
| 5 | MODS: Matching with On-Demand View Synthesis | 61 |
| 5.1 | Introduction | 61 |
| 5.2 | The MODS algorithm | 63 |
| 5.2.1 | Synthetic views generation | 63 |
| 5.2.2 | Local feature detection and description | 64 |
| 5.2.3 | Tentative correspondence generation | 65 |
| 5.2.4 | Geometric verification | 66 |
| 5.3 | Implementation and parameter setup | 67 |
| 5.3.1 | View synthesis for different detectors and descriptors | 67 |
| 5.3.2 | First geometrically inconsistent nearest neighbor ratio correspondence selection strategy | 69 |
| 5.4 | Experiments | 70 |
| 5.4.1 | MODS variants testing on Extreme Viewpoint and Oxford Dataset | 70 |
| 5.4.2 | MODS testing on other datasets | 72 |
| 5.4.3 | MODS testing on a non-planar dataset | 75 |
| 5.5 | Why does affine synthesis help? | 78 |
| 5.6 | Summary | 79 |
| 6 | Image Matching Benchmark | 83 |
| 6.1 | Introduction | 83 |
| 6.2 | Related Work | 85 |
| 6.2.1 | Local features | 86 |
| 6.2.2 | Robust matching | 86 |
| 6.2.3 | Structure from Motion (SfM) | 86 |
| 6.2.4 | Datasets and benchmarks | 86 |
| 6.3 | The Image Matching Challenge PhotoTourism Dataset | 87 |
| 6.3.1 | Dataset details | 88 |
| 6.3.2 | Estimating the co-visibility between two images | 88 |
| 6.3.3 | On the quality of the “ground-truth” | 89 |
| 6.4 | Pipeline | 92 |
| 6.4.1 | Feature extraction | 93 |
| 6.4.2 | Feature matching | 93 |
| 6.4.3 | Outlier pre-filtering | 93 |
| 6.4.4 | Stereo task | 94 |
| 6.4.5 | Multiview task | 94 |
| 6.4.6 | Error metrics | 94 |
| 6.4.7 | Implementation | 95 |
| 6.5 | Details are Important | 95 |
| 6.5.1 | RANSAC: Leveling the field | 95 |
| 6.5.2 | RANSAC: One method at a time | 96 |
| 6.5.3 | Ratio test: One feature at a time | 96 |
| 6.5.4 | Choosing the number of features | 99 |
| 6.5.5 | Additional experiments | 99 |
| 6.6 | Establishing the State of the Art | 101 |
| 6.6.1 | Results with 8k features | 102 |
| 6.6.2 | Results with 2k features | 104 |
| 6.6.3 | 2k features vs 8k features | 104 |
| 6.6.4 | Outlier pre-filtering with deep networks | 104 |
| 6.6.5 | On the effect of local feature orientation | 104 |
| 6.6.6 | On the effect of approximate nearest neighbor matching | 108 |
| 6.6.7 | Pose mAA vs. traditional metrics | 108 |
| 6.6.8 | Breakdown by scene | 109 |

| | | |
|----------|---|------------|
| 6.6.9 | Breakdown by co-visibility | 109 |
| 6.6.10 | Classical metrics vs. pixel threshold | 112 |
| 6.6.11 | Breakdown by angular threshold | 112 |
| 6.6.12 | Qualitative results | 113 |
| 6.7 | Further results and considerations | 113 |
| 6.7.1 | Number of inliers per step | 115 |
| 6.7.2 | Feature matching with FGINN | 115 |
| 6.7.3 | Image pre-processing | 115 |
| 6.7.4 | Optimal settings breakdown | 116 |
| 6.8 | Summary | 116 |
| 7 | Questions from Past and Ideas for Future Research | 123 |
| 7.1 | Questions | 123 |
| 7.1.1 | Should one make the training-time setup as close as possible to the test time setup? | 123 |
| 7.1.2 | End-to-end or stop gradients? | 123 |
| 7.1.3 | Evaluation: typical use-case sampling or focus on corner cases? | 124 |
| 7.2 | Future research directions | 124 |
| 7.2.1 | Application-specific local features | 124 |
| 7.2.2 | Rethinking the overall wide baseline stereo pipeline, optimized for specific applications | 124 |
| 7.2.3 | Rethinking and improving the process of training data generation for WxBS | 125 |
| 7.2.4 | Matching with On-Demand View Synthesis revisited | 125 |
| 7.2.5 | More inputs! | 125 |

CHAPTER 1

Introduction



1.1 What is wide multiple baseline stereo?

Imagine you have a nice photo you took in autumn and would like to take one in summer, from the same spot. How would you achieve that? You go to the place and you start to compare what you see on the camera screen and on the printed photo. Specifically, you would probably try to locate the same objects, e.g., “that high lamppost” or “this wall clock”. Then one would estimate how differently they are arranged on the old photo and camera screen. For example, by checking whether the lamppost is occluding the clock on the tower or not. That would give an idea of how you should move your camera.

Now, what if you are not allowed to take that photo with you, because it is a museum photo and taking pictures is prohibited there. Instead you can create a description of it. In that case, it is likely that you would try to make a list of features and objects in the photo together with the descriptions, which are sufficient to distinguish the objects. For example, “a long staircase on the left side”, “The nice building with a dark roof and two towers” or “the top of the lamppost”. It would be useful to also describe where these objects and features are pictured in the photo, “The lamp posts are on the left, the closest to the viewer is in front of the left tower with a clock. The clock tower is not the part of the building and stands on its own”. Then when arriving, you would try to find those objects, match them to the description you have, and try to estimate where you should go. You repeat the procedure until the camera screen shows a picture which is fitting the description you have and the image you have in your memory.

Congratulations! You just have successfully registered the two images, which have a significant difference in viewpoint, appearance, and illumination. In the process of doing so, you were solving multiple times the wide multiple-baseline stereo problem (WxBS) – estimating the relative camera pose from a pair of images, different in many aspects, yet depicting the same scene.

A bit more formally¹, the *wide multiple baseline stereo (WxBS)* is a process of establishing a sufficient number of pixel or region correspondences from two or more images depicting the same scene to estimate the geometric relationship between cameras, which produced these images. Typically, WxBS relies on the scene rigidity – the assumption that there is no motion in the scene except the motion of the camera itself. The stereo problem is called wide multiple baseline if the images are significantly different in more than one aspect: viewpoint, illumination, time of acquisition, and so on. Historically, people were focused on the simpler problem with a single baseline, which was geometrical, i.e., viewpoint difference between cameras, and the area was known as wide baseline stereo. Nowadays, the field is mature and research is focused on solving more challenging multi-baseline problems. WxBS is a building block of many popular computer vision applications, where spatial localization or 3D world understanding is required – panorama stitching, 3D reconstruction, image retrieval, SLAM, etc.

1.2 Areas related to the wide multiple baseline stereo

Let us position the WxBS problem in computer vision research by showing its similarities and differences to related areas and concepts.

Image registration is a general term for the task of establishing dense correspondences between images of the same or related scenes, surfaces of volumes, often in 2D. Unlike WxBS, image registration typically assumes a small geometric baseline and no occlusions, but does not assume the rigidity of the scene. Image registration often assumes some underlying model, e.g., elastic deformation, whose parameters need to be estimated. Image registration also allows images of a different modality, e.g., registering the X-ray image to the ultrasound image, or aligning different but consecutive cuts of some tissue, each of those is coloured with a different chemical pigment.

A related concept – *optical flow* – is a process of establishing dense correspondences between images, which are taken in consecutive moments of time. Optical flow does not have the rigidity assumption unlike the WxBS but instead relies on the narrow temporal baseline and often on the (soft) brightness constancy assumption. Unlike image registration, optical flow typically deals with occlusions, but does not estimate camera movement or scene geometry.

Narrow baseline stereo or – simply – *stereo* estimates the disparity between the left and right frames. The disparity is a difference in horizontal coordinates between the corresponding points in the left and right images. The disparity definition suggests that the camera setup is a calibrated camera pair, or, at least, that the geometric relationship between cameras is known beforehand. Moreover, as the name suggests, the stereo algorithm assumes a small geometrical baseline. Gradient-based methods often can be successfully applied to solve the stereo, image registration and optical flow problems, unlike the wide baseline stereo.

The *3D reconstruction* or *Structure-from-Motion (SfM)* aims to reconstruct a 3D scene structure, e.g., in the form of a point cloud, from a set of 2D images. WxBS, unlike SfM, does not estimate the underlying point cloud. Moreover, the SfM, be it sparse or dense, benefits from having as many correspondences as possible to obtain a good 3D model. WxBS, in its turn, tends to have just enough correspondences needed for estimating the geometry model.

Visual localization is a task of estimating the camera pose from which a query image was acquired, given the database of training images with a known pose. WxBS can be seen as one component of the general system, which is used for solving the visual localization. For example, one can first perform an image search to produce the list of candidate images most similar to the query image. Then the WxBS is applied to estimate the query image camera pose. However, that is only one of the possible ways of solving the visual localization problem. For example, one could train a model, which directly estimates the camera pose given the image.

1.3 How to solve the wide multiple baseline stereo problem?

The wide baseline stereo problem is commonly addressed by a family of algorithms, the general structure of which is shown in Figure 1.1. We will be referring to it as the WxBS pipeline or the WxBS algorithm. Let us describe it in more detail and discuss the reasoning behind each block.

¹The formal definition of the WxBS will be given in Section 2.1.

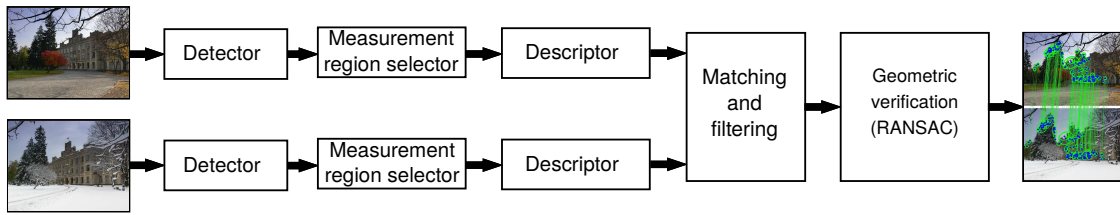


Figure 1.1: WxBS: Wide multiple baseline stereo pipeline.

1. A set of the local features² is detected in each image independently. In automated systems the local features are usually low level structures like corners, blobs and so on. However, they can also be more high level semantic structures, as we used in the example in a Section 1.1: “a long staircase on the left side”, “the top of the lamppost” and so on. An important detail is that detection is typically done in each image separately. Why is it the case? If the task is to match only a single image pair, that would be an unnecessary restriction. It is even beneficial to process the images jointly, as a human would do, by placing both images side-by-side and looking at them back and forth. However, the wide baseline stereo task rarely arises by itself, more often it is only a part of a bigger system, e.g. visual localization or 3D reconstruction from the collection of images. Therefore, one needs to match an image to not the one, but multiple other images. That is why it is beneficial to perform feature extraction only once per image and then load the stored results. Moreover, independent feature extraction is a task which is easy to parallelize and that is typically done in most of libraries and frameworks for the WxBS. One could be wondering if the local feature detection process is necessary at all? Indeed, it is possible to avoid feature detection and consider all the pixels as “detections”. The problem with such approach is the high computational and memory complexity – even a small 800×600 image contains half a million pixels, which need to be matched to half a million pixels in another image.
2. A region around the local feature to be described is selected. If one considers a keypoint to be literally a point, it is impractical to distinguish between them based only on coordinates and, maybe, the single RGB value of the pixel. On the other extreme, part of the image, which really far from the current keypoint helps little to nothing in terms of finding a correspondence. Thus, a reasonable trade-off needs to be made. Keypoint therefore can be think of as the “reference point of the distinguished region”, e.g. a center of the blob. It worth mention that some detectors return a region by default, so this step is omitted, or, to be precise, included into step 1 “local features detection”. However, it is useful to have it discussed separately .
3. A patch around each local feature is described with a local feature descriptor, i.e. converted to a compact format. Such procedure also should be robust to changes in acquisition conditions so that descriptors related to the same 3D points are similar and dissimilar otherwise. The local feature descriptors are then used for the efficient generation of tentative correspondences. Could one skip this stage? Yes, but as with the local feature detection, the skipping is not desirable from a computational point of view – the benefits are discussed in the next stage – matching. Local feature detection, measurement region selection and description together convert an image into a sparse representation, which is suitable for the correspondence search. Such representation is more robust to the acquisition conditions and can be further indexed if used for image retrieval.
4. Tentative correspondences between detected features are established and then filtered. The simplest and common way to generate tentative correspondences is to perform a nearest neighbor search in the descriptor space. The commonly used descriptors are the binary or float point vectors, which allows to employ various algorithms for approximate nearest neighbor search and trade a small amount of accuracy for orders of magnitude speed-up. Such correspondences need to be filtered, that is why they are called “tentative” or “putative” – a significant percentage of them is incorrect. There are many reasons for that – imperfection

²Also known as keypoints, local regions, distinguished regions, salient regions, salient points, etc.

of the local feature detector, descriptor, matching process and simply the fact that some parts of the scene are visible only on one image, but not another.

5. The geometric relationship between the two images is recovered, which is the final goal of the whole process. In addition, the tentative correspondences, which are not consistent with the found geometry, are called outliers and are discarded. The most common way of robust model estimation in the presence of outliers is called RANSAC – random sample consensus. There are other methods as well, e.g. re-weighted least squares, but RANSAC predominantly used in practice.

The class of algorithms that we have just described significantly changed the computer vision landscape after its introduction in middle 1990-s³ and is still at the core of many computer vision tasks in 2021, such as 3D reconstruction [240, 3, 226], SLAM (Simultaneous localization and mapping) [230, 117, 176], panorama stitching [38], feature-based image registration [242], visual localization [190] and image retrieval [179].

This thesis is devoted to the wide multiple baseline stereo problem and to improving the WxBS algorithm in the form shown in Figure 1.1 in particular. We present ways of improving individual components of the WxBS algorithm, as well as adding new parts to it⁴.

1.4 History of the wide multiple baseline stereo and its role in the deep learning world

As often happens, a new problem arises from the old – narrow or short baseline stereo. In the narrow baseline stereo, images are taken from nearby positions, often exactly at the same time. One could find correspondence for the point (x, y) from the image I_1 in the image I_2 by simply searching in some small window around (x, y) [86, 168] or, assuming that a camera pair is calibrated and the images are rectified – by searching along the known epipolar line [90].

One of the first, if not the first, approaches to the wide baseline stereo problem was proposed by Schmid and Mohr [224] in 1995. Given the difficulty of the wide multiple baseline stereo task at the moment, only a single – geometrical – baseline was considered, thus the name – wide baseline stereo. The idea of Schmid and Mohr was to equip each keypoint with an invariant descriptor. This allowed establishing tentative correspondences between keypoints under viewpoint and illumination changes, as well as occlusions. One of the stepping stones was the corner detector by Harris and Stevens [88], initially used for the application of tracking. It is worth a mention, that there were other good choices for the local feature detector at the time, starting with the Forstner [77], Moravec [168] and Beaudet feature detectors [28].

The Schmid and Mohr approach was later extended by Beardsley, Torr and Zisserman [27] by adding RANSAC [75] robust geometry estimation and later refined by Pritchett and Zisserman [192, 191] in 1998. The general pipeline remains mostly the same until now [264, 55, 109], which is shown in Figure 1.1.

Soon after the introduction of the WBS algorithm, it became clear that its quality significantly depends on the quality of each component, i.e., local feature detector, descriptor, and geometry estimation. Local feature detectors were designed to be as invariant as possible, backed up by the scale-space theory, most notable developed by Lindenberg [135, 136, 137]. A plethora of new detectors and descriptors were proposed in that time. We refer the interested reader to these two surveys: by Tuytelaars and Mikolajczyk [270] (2008) and by Csurka *et al.* [55] (2018). Among the proposed local features is one of the most cited computer vision papers ever – SIFT local feature [141, 140]. Besides the SIFT descriptor itself, Lowe’s paper incorporated several important steps, proposed earlier with his co-authors, to the matching pipeline. Specifically, they are quadratic fitting of the feature responses for precise keypoint localization [37], using the Best-Bin-First kd-tree [29] as an approximate nearest neighbor search engine to speed-up the tentative correspondences generation, and using second-nearest neighbor (SNN) ratio to filter the tentative matches. It is worth noting that SIFT feature became popular only after Mikolajczyk benchmark paper [151, 152] that showed its superiority to the rest of alternatives.

Robust geometry estimation was also a hot topic: a lot of improvements over vanilla RANSAC were proposed. For example, LO-RANSAC [50] proposed an additional local optimization step

³The brief history of the wide (single) baseline stereo will be given in Section 1.4

⁴See Section 1.5 for the detailed list of contributions

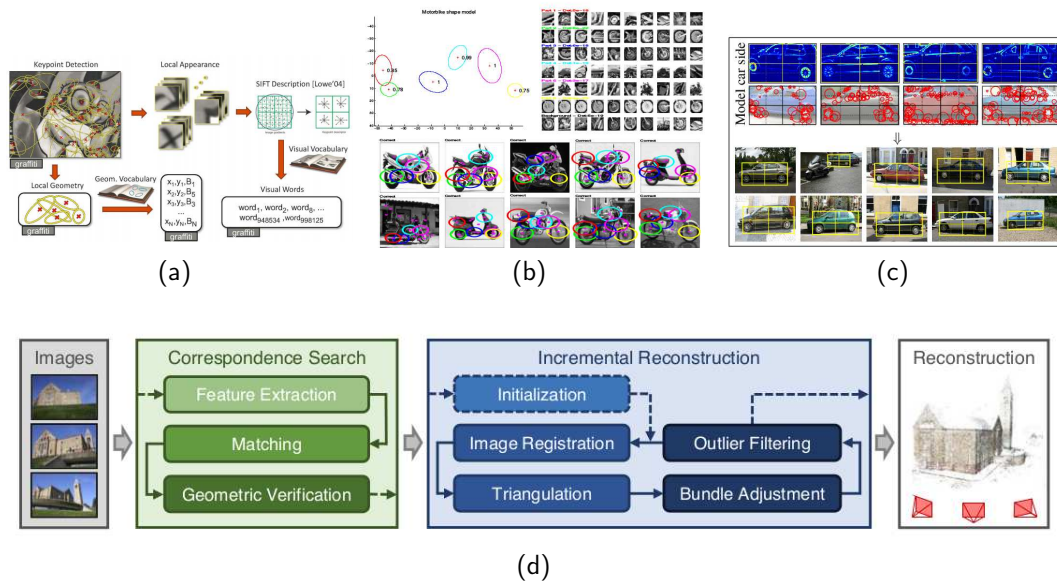


Figure 1.2: Application of the wide baseline stereo in the different computer vision tasks. All the images are taken from the cited papers respectively. (a) Bag of words image search. Image credit: Ondra Chum and Filip Radenović. (b) Bag of local features representation for classification [73] (c) Exemplar-representation of the classes using local features [48], (d) SfM pipeline from COLMAP [226]

into RANSAC to significantly decrease the number of required steps. PROSAC [49] takes into account the tentative correspondences matching score during sampling to speed up the procedure. DEGENSAC [51] improved the quality of the geometry estimation in the presence of a dominant plane in the images, which is the typical case for urban images. We refer the interested reader to the survey by Choi *et al.* [46].

Wide baseline stereo framework became the main part of such applications as 3D reconstruction, SLAM, and image localization. The success of the wide baseline stereo pipeline with SIFT features led to application of its components to other computer vision tasks, which were reformulated through the wide baseline stereo lens. We took the illustration figures from the iconic papers of that time in Figure 1.2. To name a few:

- **Scalable image search.** Sivic and Zisserman in the famous "Video Google" paper [237] proposed to treat local features as "visual words" and use ideas from text processing for searching in image collections. Later, even more WBS elements were reintroduced to image search, most notably – **spatial verification** [188]: a simplified RANSAC procedure to verify if visual word matches were spatially consistent.
- **Image classification** was performed by placing a classifier (SVM, random forest, etc) on top of some encoding of the SIFT-like descriptors, extracted sparsely – by Fergus *et al.* [73], Dance *et al.* [78] or densely by Lazebnik *et al.* [125].
- **Object detection** was formulated as a relaxed wide baseline stereo problem – by Chum and Zisserman [48] or as a classification of SIFT-like features inside a sliding window – by Dalal and Triggs [58].
- **Semantic segmentation** was performed by a classification of the local region descriptors, typically SIFT and color features, and postprocessing afterwards – by Tighe and Lazebnik [257].

Deep learning invasion: Retreat to the geometrical fortress. In 2012 the deep learning-based AlexNet [121] approach beat all methods in image classification at the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). Soon after, Razavian *et al.* [206] have shown that convolutional neural networks (CNNs) pre-trained on the Imagenet outperform more complex

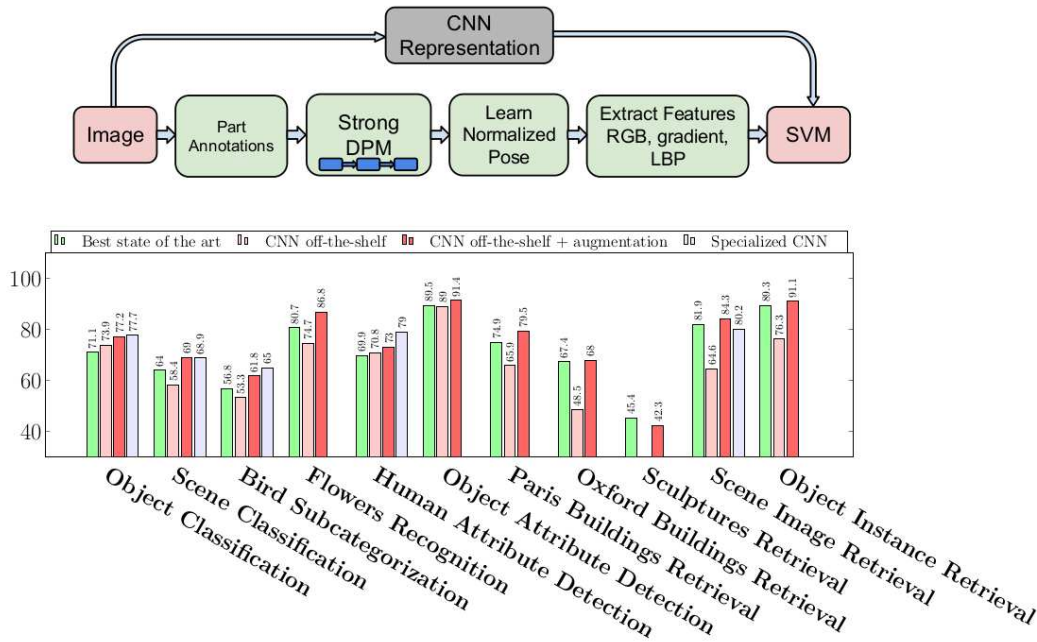


Figure 1.3: CNN representation beats complex traditional pipelines. Reds are CNN-based and greens are the handcrafted. From [206].

traditional solutions in image and scene classification, object detection and image search, see Figure 1.3. The performance gap between deep learning and “classical” solutions was large and quickly increasing. In addition, deep learning pipelines, be it off-the-shelf pretrained, fine-tuned or the end-to-end learned networks, are simple from the engineering perspective. That is why the deep learning algorithms quickly become the default option for lots of computer vision problems.

However, there was still a domain, where deep learned solutions failed, sometimes spectacularly: geometry-related tasks. Wide baseline stereo [150], visual localization [115] and SLAM are still areas, where the classical wide baseline stereo dominates [220, 298, 190]. The full reasons why convolution neural network pipelines are struggling to perform tasks that are related to geometry, and how to fix that, are yet to be understood. The observations from the recent papers are following:

- CNN-based pose predictions are roughly equivalent to the retrieval of the most similar image from the training set and outputting its pose [220]. This kind of behaviour is also observed in a related area: single-view 3D reconstruction performed by deep networks is essentially a retrieval of the most similar 3D model from the training set [252].
- Geometric and arithmetic operations are hard to represent via vanilla neural networks (i.e., matrix multiplication followed by non-linearity) and they may require specialized building blocks, approximating operations of algorithmic or geometric methods, e.g. spatial transformers [105] and arithmetic units [267, 146]. Even with such special-purpose components, the deep learning solutions require "careful initialization, restricting parameter space, and regularizing for sparsity" [146].
- Vanilla CNNs suffer from sensitivity to geometric transformations like scaling and rotation [53] or even translation [292]. The sensitivity to translations might sound counter-intuitive, because the convolution operation by definition is translation-covariant. However, a typical CNN contains also zero-padding and downscaling operations, which break the covariance [292, 101]. Unlike them, classical local feature detectors are grounded on scale-space [137] and image processing theories. Some of the classical methods deal with the issue by explicit geometric normalization of the patches before description.
- CNNs predictions can be altered by a change in a small localized area [40] or even a

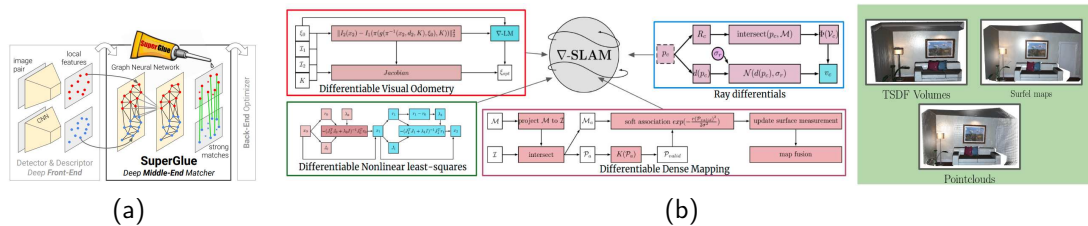


Figure 1.4: Modern approach to wide multiple baseline stereo: use learned components in a classical structure pipeline or make the whole pipeline differentiable. Images taken from the cited papers. (a) SuperGlue [216]: separate matching module for handcrafted and learned features. (b) gradSLAM: differentiable formulation of SLAM pipeline [120]

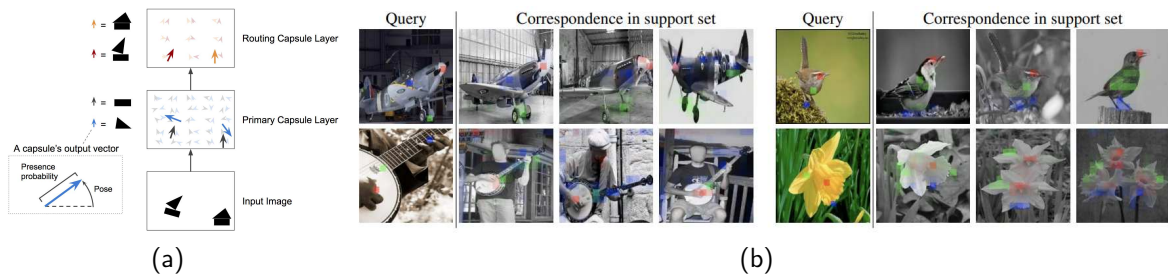


Figure 1.5: Wide baseline stereo ideas in the modern neural network architectures. (a) Capsule networks [214]: revisiting the WBS idea. Each feature response is accompanied with its pose. Feature poses should be in agreement, otherwise object would not be recognized. Image by Aurélien Géron (b) Cross-transformers: spatial attention helps to select relevant feature for few-shot learning [63].

single pixel [246], while the wide baseline stereo methods require the consensus of different independent regions.

This leads us to the following question – **is deep learning helping WxBS today?** The answer is yes. After the quick interest in the black-box-style models, the current trend is to design deep learning solutions for the wide baseline stereo in a modular fashion [297], resembling the one in Figure 1.1. Such modules are learned separately. For example, the HardNet [158] descriptor replaces SIFT local descriptor. The Hessian detector can be replaced by deep learned detectors like KeyNet [23] or the joint detector-descriptor [62, 207, 69]. The matching and filtering are performed by the SuperGlue [216] matching network, etc. There have been attempts to formulate the full pipeline solving problem like SLAM [120] in a differentiable way, combining the advantages of structured and learning-based approaches. Examples of such structured and modular architectures are shown in Figure 1.4.

Is WxBS helping deep learning? The WxBS is helping the progress of deep learning-based computer vision research in three main ways. First is providing supervision and labeling of the data as a result of running structure-from-motion pipelines like COLMAP. Produced 3D models of the scenes are used for training image retrieval [203, 204], monocular depth estimation [134], local features [181, 70], correspondence filtering [284, 248] and other models. The second way is to incorporate geometry-based constraints into the learning [82, 283, 278]. The third is the influence of the ideas from WxBS on the design of new model types. Specifically, capsule networks [97, 214] are based on two ideas, rooted in wide baseline stereo: an image should be represented as a set of local parts, robust to occlusion. The second idea is that the decision should be based on the spatial consensus of local feature correspondences. Unlike vanilla CNNs, capsule networks encode not only the intensity of feature response, but also its location. Geometric agreement between “object parts” is a requirement for outputting a confident prediction. Similar ideas are explored for improving the adversarial robustness of CNNs [133] or for semi-supervised learning [63], where the spatial consistency is employed to provide supervision in the few-shot learning setup. Examples of such consensus are shown in Figure 1.5.

While wide multiple baseline stereo is a mature field now and does not attract even nearly as much attention as before, it continues to play an important role in computer vision.

1.5 Contributions

The contributions of this thesis are:

1. The introduction of the concept of the wide multiple baseline stereo (WxBS) [163] problem. WxBS is a generalization of the standard wide baseline stereo problem that considers matching of images that simultaneously differ in more than one image acquisition factor such as viewpoint, illumination, sensor type or where object appearance changes significantly, e.g. over time. A new dataset with the ground truth, evaluation metric and baselines was introduced.
2. A loss function [158] for learning a local image descriptor that relies on hard negative mining within a mini-batch and the maximization of the distance between the closest positive and closest negative patches. The resulting descriptor called HardNet is compact – it has the same dimensionality as SIFT (128), it shows state-of-art performance in standard matching, patch verification and retrieval benchmarks. HardNet is fast to compute on a GPU. This work was done in collaboration with Anastasiia Mishchuk, who is first author of the paper [158].
3. A loss function for descriptor-based registration and learning, named the hard negative-constant loss [161]. It combines the advantages of the triplet and contrastive positive losses.
4. We experimentally show that geometric repeatability of the local feature is not a sufficient condition for successful feature matching [161]. The learning of affine shape increases the number of corrected matches if it steers the estimators towards discriminative regions and therefore must involve optimization of descriptor-related loss.
5. A method for learning the affine shape, orientation and potentially other parameters related to geometric and appearance properties of local features [161]. The learning method does not require a precise ground truth which reduces the need for manual annotation.
6. A tentative correspondences generation strategy is presented which generalizes the standard first to second closest distance ratio [162]. The selection strategy which shows performance [129] superior to the standard method is applicable to either hard-engineered descriptor like SIFT, LIOP and MROGH or deeply learned like HardNet.
7. MODS – matching with on-demand view synthesis. MODS is a two-view matching algorithm that handles viewing angle difference even larger than the previous state-of-the-art ASIFT algorithm, without a significant increase of computational costs over “standard” wide and narrow baseline approaches [164, 162].
8. EVD – extreme view dataset for benchmarking the two-view matching algorithms under extreme viewpoint changes [164, 162].
9. A comprehensive benchmark [109] for local features and robust estimation algorithms. The modular structure of its pipeline allows to easily integrate, configure, and combine methods and heuristics. This work is done in a wide collaboration across four labs. I focused on the benchmark methodology, specifically importance of RANSAC and matching tuning in order to properly evaluate local feature methods.

1.6 Publications

The presented work is published in the following papers:

[162] **Dmytro Mishkin**, Jiří Matas, and Michal Perdoch. MODS: Fast and robust method for two-view matching. *Computer Vision and Image Understanding*, 141:81 – 93, 2015.

- [163] **Dmytro Mishkin**, Jiří Matas, Michal Perdoch, and Karel Lenc. WxBS: Wide baseline stereo generalizations. In BMVC, 2015.
- [158] Anastasiia Mishchuk, **Dmytro Mishkin**, Filip Radenović and Jiří Matas. Working Hard to Know Your Neighbor’s Margins: Local Descriptor Learning Loss. In NeurIPS, 2017.
- [161] **Dmytro Mishkin**, Filip Radenović, and Jiří Matas. Repeatability is Not Enough: Learning Affine Regions via Discriminability. In ECCV, 2018.
- [109] Yuhe Jin, **Dmytro Mishkin**, Anastasiia Mishchuk, Jiří Matas, Pascal Fua, Kwang Moo Yi, and Eduard Trulls. Image matching across wide baselines: From paper to practice. IJCV, 2020.

The differentiable implementation of some of the abovementioned contributions is included into an open source library named **kornia** [208]. The paper describing the library is however, of little scientific interest and we do not include it into the thesis.

- [208] Edgar Riba, **Dmytro Mishkin**, Daniel Ponsa, Ethan Rublee, Gary Bradski. Kornia: an open source differentiable computer vision library for PyTorch. In WACV, 2020.

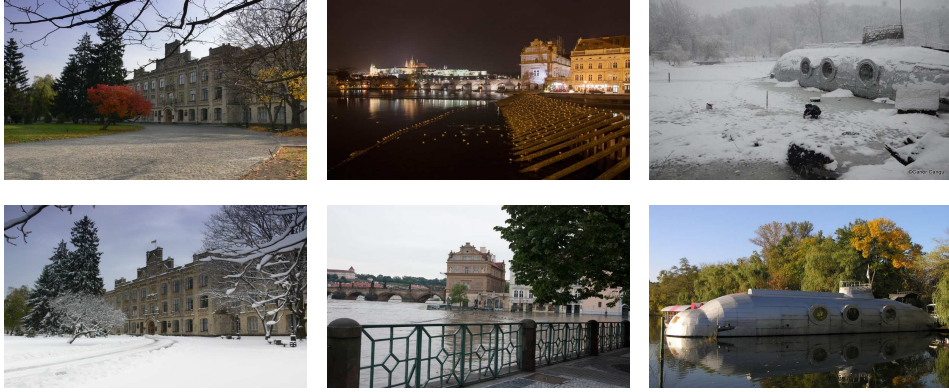
Some of the results presented in this thesis and related to HardNet descriptor, were obtained with Milan Pultar, who completed the master thesis under my supervision. They are presented in full detail in his master thesis [199].

Finally, the following publications have not been included into the thesis, either because they are less relevant to the thesis main topic ([160],[159], [165], [123], [21]), or because the author contribution is small([8], [123], [198], [9]):

- [164] **Dmytro Mishkin**, Michal Perdoch, and Jiří Matas. Two-view matching with view synthesis revisited. In Proc of Image and Vision Computing New Zealand (IVCNZ), 2013.
- [129] Karel Lenc, Jiří Matas, and **Dmytro Mishkin**. A few things one should know about feature extraction. Computer Vision Winter Workshop, pages 67 – 74, 2014
- [160] **Dmytro Mishkin**, Michal Perdoch and Jiří Matas. Place Recognition with WxBS Retrieval. CVPR 2015 Workshop on Visual Place Recognition in Changing Environments.
- [159] **Dmytro Mishkin** and Jiří Matas. All you need is a good init. In ICLR, 2016.
- [165] **Dmytro Mishkin**, Nikolay Sergievskiy and Jiří Matas. Systematic evaluation of convolution neural network advances on the imagenet. Computer Vision and Image Understanding, 161:11 – 19, 2017.
- [8] Javier Aldana-Iuit, **Dmytro Mishkin**, Ondra Chum and Jiří Matas. In the Saddle: Chasing Fast and Repeatable Features. In ICPR, 2016.
- [123] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, **Dmytro Mishkin** and Jiří Matas. DeblurGAN: Blind Motion Deblurring Using Conditional Adversarial Networks. In CVPR 2018
- [198] Milan Pultar, **Dmytro Mishkin** and Jiří Matas. Leveraging Outdoor Webcams for Local Descriptor Learning. In Computer Vision Winter Workshop, 2019.
- [9] Javier Aldana-Iuit, **Dmytro Mishkin**, Ondrej Chum and Jiří Matas. Saddle: Fast and repeatable features with good coverage. Image and Vision Computing, 2019.
- [21] Daniel Barath, **Dmytro Mishkin**, Ivan Eichhardt and Jiří Matas. Efficient Initial Pose-graph Generation for Global SfM. In CVPR, 2021 (to appear).

CHAPTER 2

Wide multiple baseline stereo



2.1 Definition

Let us denote observations $O_{j,i}$, $i = 1..n$, each of which belongs to one of the views V_j , $j = 1..m$, $m \leq n$. Observation $O_{j,i} = \{x; d\}$ consist of spatial information x and the descriptor d ; index i is a unique index among all observations. View $V_j = \{\bigcup O_{j,i}; \theta_j\}$ consists of union of the observations and the additional information θ , which is shared for all observations, which belong to view V_j .

For example, a single observation can be an RGB pixel. Its spatial information is the pixel coordinates $x = (u, v) \in R^2$ and the descriptor $d = (r, g, b) \in \{(0..255)^3\}$ is its color intensity value. The view V then is the image, with information θ containing the camera pose, camera intrinsics, sensor type, illumination conditions, timestamp of the acquisition, etc. Some of this information can be unknown to the user, i.e., a hidden variable. Another example could be an event camera [79]. In that case, the observation O contains the pixel coordinates $x = (u, v) \in R^2$ and the descriptor $d \in \{+, -, 0\}$ is the sign of the intensity change. The view V will contain the information about the sensor, camera pose, timestamp and the single observation O , because every event has a unique timestamp. Observations and views can be of different nature and dimensionality. E.g., views V_1, V_2 might be RGB images, V_3 – point cloud from a laser scanner, V_4 – an image from a thermal camera, and so on.

An unordered pair of observations $(O_{j,i}, O_{k,l})$ forms a correspondence c_{jikl} if they are belong to different views $V_j \neq V_k$. A set of observations is called multiview correspondence C_o , when there is no more than one observation $O_{j,i}$ per view $V_j \forall j$. Some of observations $O_{j,i}$ can be empty \emptyset , i.e. not observed in the specific view V_j .

The world model M is a system of constraints on views, observations and correspondences $M : f(V, O, C, \Omega) = 0$, where Ω stands for the world model parameters. For example, one of the popular models are epipolar geometry and rigid motion assumption: $x'^T F x = 0$, where Ω is a fundamental matrix [90] F for this case. In practice, however, the exact satisfaction of the ground truth model is unlikely due to the measurement errors and noise. That is why we also incorporate an acceptable noise level ϵ into our model $M : f(V, O, C, \Omega) < \epsilon$. The correspondence is called ground truth or veridical if it satisfies the constraints posed by the ground truth world model.

We can now define a wide baseline stereo. By the wide baseline stereo we understand the estimation of the unknown parameters Ω of the world model M^* by establishing the correspondences

which satisfy this world model, given the views and observations.

By the baseline we understand the difference between views information θ_j . If views V_j are significantly different in a single aspect, e.g., camera pose, we will call the task wide single baseline stereo. If more than one aspect is different, e.g., both camera pose and sensor type are different, we will call the task wide multiple baseline stereo, or WxBS.

Moreover, we limit our definition of the WxBS by setting the restriction on the possible world models. Most often in this thesis we will be using the the following world model. The scene is rigid and static. The observations are 2D projections of the 3D scene to the camera plane by the pinhole or rectilinear camera. The relationship between observations in different views is either epipolar geometry or projective transform [90]. Any moving object does not satisfy the world model and therefore is considered an occlusion.

Let us define the areas similar to WxBS, mentioned in Chapter 1 using the framework we have just developed. The wide baseline stereo process, which in addition estimates the (latent) scene structure, e.g., in the form of 3d point world coordinates we would call Structure-from-Motion (SfM) or 3D reconstruction. We do not consider SfM in this thesis.

The optical flow is the process of establishing correspondences c_{jikt} between the short temporal baseline views without recovering the world model parameters Ω . The image registration problem is identical to the wide multiple baseline stereo in the definition, but the family of considered world models is different (wider). The visual localization is the estimation of the camera pose θ from the given view V .

Since the paper publication in 2015. The rest of the chapter below is presented as was originally published in 2015. Since then, the interest to the topic, in the form of day-night [207, 202, 69] image matching, visual relocalization in changing environments [218, 144, 149, 263] has been significant. A series of datasets and benchmarks [16, 218, 33] and local features [286, 207, 69, 62] have been proposed. We believe that we have only started to explore the area of wide multiple baseline stereo.

2.2 Wide multiple baseline stereo: evaluation

We consider the generalization of Wide (geometric) Baseline Stereo to WxBS, a multiview image matching problem where two or more of the image formation and acquisition properties *significantly* change, i.e. they have a *wide baseline*. The "significant change" distinguishes the problem from image registration, where a dense correspondence is routinely established between multi-modal images and various complex transformations have been considered, see Zitová and Flusser [301]. For example, the standard physical models of image formation and acquisition consider, besides geometry, the effects of illumination, the properties of the transparent medium where light rays pass through in the scene, the surface properties of objects and the properties of the imaging sensors.

The following single wide baseline stereo, or correspondence, problems and their combinations are considered: illumination (WLBS) – differences in position, direction, number, intensity, and wavelength of light sources; geometry (WGBS) – difference in camera and object pose, scale and resolution - the “classical” WBS; sensor (WsBS) – change in sensor type: visible, IR, MR; noise, image preprocessing algorithms inside the camera, etc; appearance (WABS) – difference in the object appearance because of time or seasonal changes, occlusions, turbulent air, etc. We denote matching problems, or, equivalently, image pairs, with a significant change in only one of the groups listed as W1BS; if a combination of effects is present, as WxBS. To our knowledge, the most published to date image datasets and algorithms are in the W1BS class [155], [170], [277], [4], [93], [103].

We present a new public dataset with ground truth, which combines the above-mentioned challenges and contains both W2BS image pairs including viewpoint and appearance, viewpoint and illumination, viewpoint and sensor, illumination and appearance change and W3BS – problems where viewpoint, appearance and lighting differ significantly.

WxBS dataset and evaluation protocol. A set of 37 image pairs has been collected from Flickr, taken by the authors themselves and other sources. The dataset is divided into 6 categories based on the combinations of nuisance factors present, see Table 2.1. For every image, a set of approximately 20 ground-truth correspondences has been annotated. Selected examples are presented in Figure 2.1. The resolution of the majority of the images is 800×600 with the

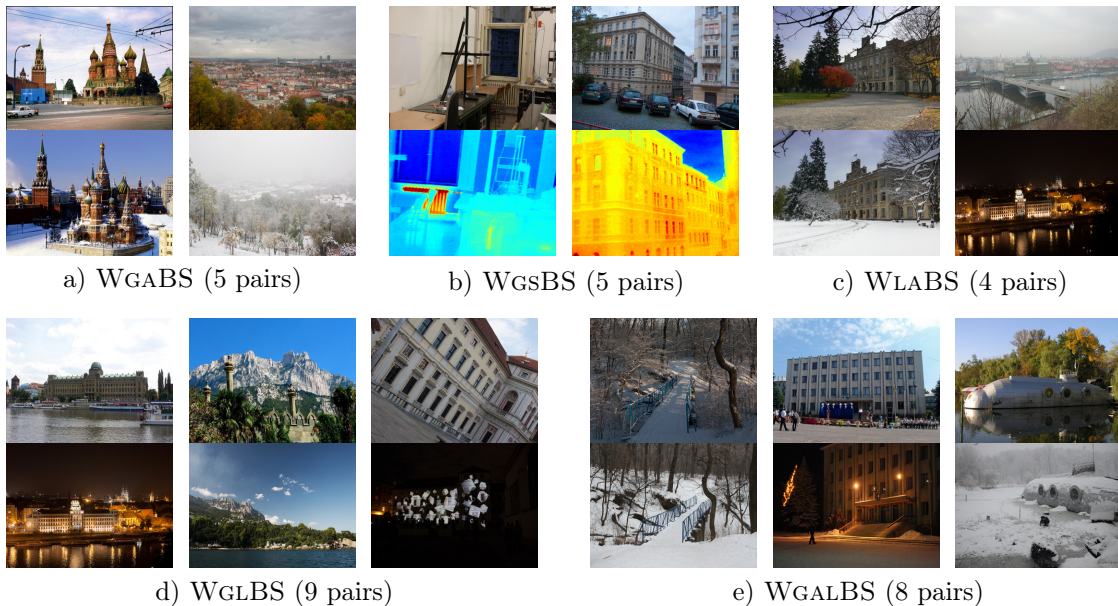


Figure 2.1: Examples of image pairs from the WxBS dataset.

exception of LWIR images from the WGSBS dataset which were captured by a thermal camera with a resolution of 250×250 pixels. The selected image pairs contain both urban and natural scenes.

Table 2.1: The WxBS datasets categories

| Short name | Nuisance | #images | Avg. # GT Corr. |
|------------|---------------------------------|---------|---------------------|
| MAP2PH | appearance (map to photo) | 6 pairs | homography provided |
| WGABS | viewpoint, appearance | 5 pairs | 22 per img. |
| WGLBS | viewpoint, lighting | 9 pairs | 21 per img. |
| WGSBS | viewpoint, modality | 5 pairs | 18 per img. |
| WLABS | lighting, appearance | 4 pairs | 25 per img. |
| WGALBS | viewpoint, appearance, lighting | 8 pairs | 17 per img. |

Ground truth and evaluation protocol. In image registration tasks, it is often sufficient to define the ground truth as a homography between an image pair. However, the WxBS dataset contains significant viewpoint changes. In the case of a non-planar scene, a homography can, at best, cover the dominant plane.

We assume that an ideal algorithm matches the majority of the scene content, thus our ground truth is a set of manually selected correspondences which evenly cover the part of the scene visible in both images. The average number of correspondences per image pair is shown in Table 2.1.

The evaluation protocol for the WxBS dataset. For each image pair indexed with $i \in \mathbb{Z}$ we have manually annotated a set of correspondences $(\mathbf{u}_i, \mathbf{v}_i) \in C_i$ where \mathbf{u} and \mathbf{v} are positions in the 1st and the 2nd image respectively. For epipolar geometry we use the *symmetric epipolar distance* and the *symmetric reprojection error* for homography [90].

Recall on ground truth correspondences C_i of image pair i and for geometry model \mathbf{M}_i is computed as a function of a threshold θ

$$r_{i, \mathbf{M}_i}(\theta) = \frac{|(\mathbf{u}_i, \mathbf{v}_i) : (\mathbf{u}_i, \mathbf{v}_i) \in C_i, e(\mathbf{M}_i, \mathbf{u}, \mathbf{v}) < \theta|}{|C_i|} \quad (2.1)$$

using appropriate error functions. For all pairs of each category W we define an overall recall per category as:

$$r_W(\theta) = \frac{1}{|W| \sum_{i \in W} r_{i, \mathbf{M}_i}(\theta)} \quad (2.2)$$

This measure is as the fraction of the confirmed annotated correspondences for a given threshold in a nuisance category.

2.3 Evaluation of local descriptors on single baseline datasets

Nowadays, there is a series of benchmarks for local feature descriptors and detectors, most notably HPatches[16], Image Matching Challenge [109], Brown dataset [38] and so on. However, in 2015 many of them were not existed and we have proposed W1BS local patch dataset for the evaluation of the descriptors and detectors. It is also used in Chapters 3, 4, 5, so we will describe it here together with the initial results obtained with it.

Datasets used in the experiments are listed in Table 2.2. When evaluating detectors (Section 2.3) all dataset images are used. However, descriptor evaluation is performed only on a subset of the most challenging and prominent pairs (i.e., only pairs 1-6 from OxfordAffine) with provided homography of each WxBS category.

Most of the published datasets (with exception of the LostInPast dataset [74]) include only a single nuisance factor per image pair.

This is suitable for the evaluation of the robustness to a particular nuisance factor but fails to predict performance in more complex environments. One of the motivations of the proposed WxBS dataset is to address this issue.

Table 2.2: Datasets used for evaluation

| Short name | Proposed by | #images | Type |
|------------|---|------------|--|
| GDB | Kelman <i>et al.</i> [113], 2007 | 22 pairs | WLBS, WsBS |
| SymB | Hauage and Snavely [93], 2012 | 46 pairs | WABS, WLBS |
| MMS | Aguilera <i>et al.</i> [4], 2012 | 100 pairs | WsBS |
| EVD | Mishkin <i>et al.</i> [164], 2013 | 15 pairs | WGBS |
| OxAff | Mikolajczyk <i>et al.</i> [151], [155], 2013 | 8 sixplets | WGBS |
| EF | Zitnick and Ramnath <i>et al.</i> [300], 2011 | 8 sixplets | WGBS, WLBS |
| Amos | Jacobs <i>et al.</i> [103], 2007 | > 100K | WLBS, WABS |
| VPRICE | VPRICE Challenge 2015 [247] | 3K pairs | WGABS, WGLBS, WGsBS, |
| Past | Fernando <i>et al.</i> [74], 2014 | 502 images | WGABS |
| WxBS | here | 37 pairs | WABS, WGABS, WGLBS, WGsBS, WLABS, WGALBS |

Descriptor evaluation The evaluation protocol is as follows. The dataset consists of 40 image pairs from the datasets listed in Table 2.2 divided into 5 parts by the nuisance factor. For all pairs, homography is the appropriate two-view relationship – the images are either without significant relative depth, or taken from virtually identical viewpoints. To minimize bias towards a specific detector, affine-covariant regions by Hessian-Affine [153], MSER [148] and FOCI [300] in the first – least challenging image of the pair are used (visible in the case of IR-vis, day for day-night, frontal when view point changes, etc.). The affine-covariant regions have been detected with dominant orientation and then reprojected to the second image by the ground truth homography. Features which are not visible in the second image have been discarded. Therefore, the geometric repeatability of affine regions on the selected regions is always 100% and the maximum possible recall is 1. Color-to-grayscale image transformation has been done via channel averaging, which gives the best matching performance [111].

Then the affine regions were normalized to patch size 41x41 (scale $\sigma = 3\sqrt{3}$) and described with given descriptors. An affine-normalization procedure is performed even for the fast binary descriptors, which is rarely used because of the significant additional processing time. However, the goal of our experiment is to explore the descriptor performance in challenging conditions, not their speed. The procedure helps – the typical threshold of the Hamming distance for binary descriptors on unnormalized patch is around 60-80, while on affine normalized patches similar performance is obtained with a threshold around 10-30. All descriptors clearly benefit from the affine-normalized process, e.g., the graffiti 1-6 pair from the OxfordAffine dataset could be matched with FREAK descriptor only when using a normalized patch.

The tested descriptors are: SIFT [140], rSIFT [14], hrSIFT (gradients in interval $[0; \pi)$) [113], In-vSIFT (SIFT with reordered cells as for inverted image) [87], LIOP[296], AKAZE [7], MROGH [72], FREAK [6], ORB [212], SymFeat [93], SSIM [231] (implementation [45]), DAISY [259] and L_2 -normalized raw grayscale pixel intensities. Floating point descriptors have been compared using L_2 distance, binary using Hamming distance. The Recall-Precision curves are shown in Figure 2.2. The second nearest distance ratio is used to parameter the curve for floating point descriptors, the Hamming distance for binary ones.

Note that most of the descriptors gain significantly from photometric normalization, cf. the first two rows of Figure 2.2. The published implementations are clearly sensitivity to contrast variations.

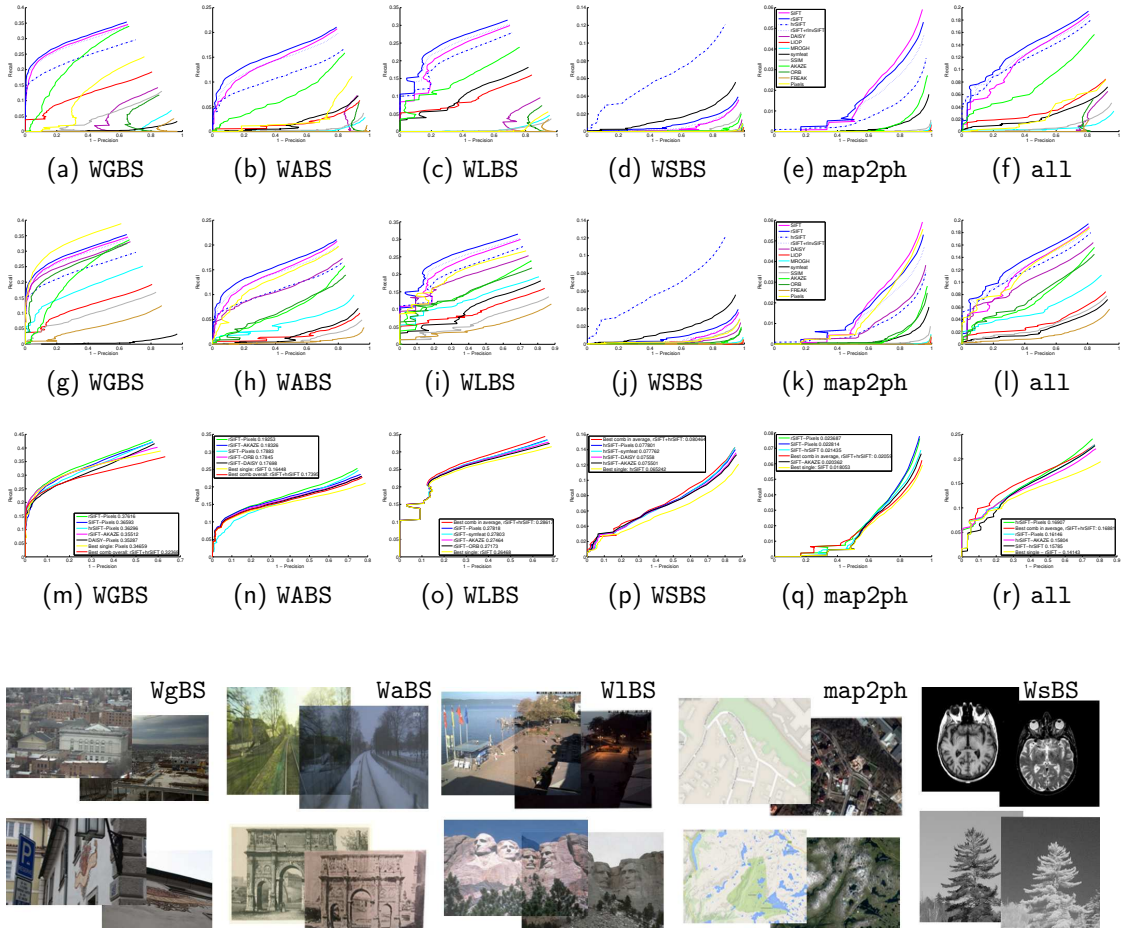


Figure 2.2: First row: descriptors computed using authors’ implementation, second row - descriptors computed on photometrically normalized patches (mean = 0.5, var = 0.2) patches as done in SIFT. Third row: top 5 complementary pairs of descriptors (photometrically normalized). The numbers in legend are mean average precision. Bottom row: examples of the image pairs from each subset. Note that axis scales differs in each column, i.e. for different WxBS problems.

The results show that gradient-histogram based SIFT and its variants including DAISY are the best performing descriptors by a big margin in the presence of any (geometric, illumination, etc) nuisance factors despite the fact that some of the competitors – LIOP, MROGH – have been specifically designed to deal with illumination changes. The second best descriptor is – surprisingly – the patch with contrast- L_2 -normalized pixels, which beats all other descriptors. It has huge memory footprint – 1681 floats, but the affine-photo- L_2 -normed grayscale pixel intensities are a strong descriptor baseline.

Most of the descriptors, despite their different underlying assumptions and algorithmic structure, successfully match almost the same patches (see the third row in Figure 2.2) – and the most complementary descriptor to the leading rSIFT is its gradient-reversal-insensitive version – hrSIFT.

The results confirming the domination of SIFT-based methods are in agreement with [245] and [74] despite the fact that they adopted a rather different evaluation methodology. However, we could not confirm the clear superiority of the SSIM over SymFeat descriptors, which could be explained by the fact that the SSIM descriptor was designed for use only with the SSIM detector.

Tentative correspondences are generated using kD-tree [171] and the 1st geometrically inconsistent rule with radius equal 10 pixels as a threshold is applied [164]. Descriptors from different detector types (Hessian, MSER+, MSER-) as well as for different descriptors are put in separate kD-trees. After matching, all tentative correspondences are put into a single list and duplicates, which appears due to view synthesis, are filtered if the features in both images are within a 3 pixel radius.

Detectors evaluation. The following detectors are compared: MSER [148], DoG [140],

Table 2.3: Detector evaluation results. The number of matched image pairs (left) and the average running time (right). The FOCI detector is run through MS Windows simulator wine, the time includes a big overhead.

| Alg. | EF | | EVD | | MMS | | WGABS | | W GALBS | | WGLBS | | WGsBS | | W LABS | | Past | | OxAff | | SymB | | GDB | |
|----------------------|----|------|-----|------|-----|------|-------|------|---------|------|-------|------|-------|------|--------|------|------|------|-------|------|------|------|-----|------|
| | # | time | # | time | # | time | # | time | # | time | # | time | # | time | # | time | # | time | # | time | # | time | # | time |
| | 33 | [s] | 15 | [s] | 100 | [s] | 5 | [s] | 8 | [s] | 9 | [s] | 5 | [s] | 4 | [s] | 172 | [s] | 40 | [s] | 46 | [s] | 22 | [s] |
| Threshold adaptation | | | | | | | | | | | | | | | | | | | | | | | | |
| MSER | 16 | 1.4 | 3 | 1.4 | 1 | 0.3 | 0 | 2.0 | 0 | 1.3 | 0 | 1.3 | 0 | 0.8 | 1 | 1.2 | 8 | 1.3 | 40 | 3.5 | 23 | 2.4 | 9 | 2.4 |
| AdMSER | 25 | 3.4 | 8 | 4.0 | 6 | 1.0 | 0 | 4.0 | 0 | 3.2 | 0 | 3.3 | 0 | 1.4 | 1 | 2.6 | 11 | 2.9 | 40 | 5.7 | 26 | 4.6 | 13 | 6.9 |
| DoG | 29 | 2.3 | 0 | 2.8 | 10 | 0.8 | 0 | 2.7 | 0 | 2.3 | 0 | 2.1 | 0 | 1.0 | 1 | 2.4 | 13 | 2.0 | 38 | 4.8 | 29 | 2.7 | 12 | 4.7 |
| iiDoG | 29 | 3.1 | 0 | 3.0 | 11 | 1.2 | 0 | 3.2 | 0 | 2.9 | 0 | 2.8 | 0 | 1.2 | 1 | 2.5 | 13 | 2.2 | 38 | 8.0 | 29 | 2.9 | 12 | 6.1 |
| AdDoG | 29 | 2.6 | 0 | 3.4 | 11 | 1.2 | 0 | 3.3 | 0 | 3.0 | 0 | 3.0 | 0 | 1.5 | 1 | 2.7 | 13 | 2.7 | 38 | 4.1 | 30 | 3.0 | 12 | 4.8 |
| HesAf | 32 | 4.6 | 1 | 5.2 | 15 | 1.2 | 0 | 5.5 | 0 | 3.8 | 0 | 4.2 | 0 | 2.0 | 1 | 3.6 | 24 | 4.0 | 40 | 11. | 35 | 5.8 | 17 | 9.1 |
| AdHesAf | 33 | 5.7 | 2 | 7.6 | 35 | 2.9 | 0 | 7.2 | 1 | 6.5 | 0 | 6.0 | 0 | 3.2 | 1 | 4.9 | 25 | 5.4 | 40 | 10. | 35 | 7.2 | 18 | 13. |
| Other detectors | | | | | | | | | | | | | | | | | | | | | | | | |
| W α SH | 0 | 1.8 | 0 | 5.4 | 0 | 0.6 | 0 | 2.8 | 0 | 2.5 | 0 | 1.4 | 0 | 1.8 | 0 | 1.2 | 0 | 1.9 | 24 | 4.1 | 3 | 2.8 | 3 | 6.9 |
| ORB | 3 | 4.1 | 0 | 3.6 | 1 | 0.8 | 0 | 2.8 | 0 | 2.7 | 0 | 3.6 | 0 | 1.6 | 0 | 2.8 | 1 | 2.3 | 28 | 8.7 | 5 | 3.0 | 3 | 6.1 |
| SURF | 27 | 2.3 | 0 | 2.4 | 7 | 1.0 | 0 | 2.5 | 0 | 1.9 | 0 | 2.1 | 0 | 0.9 | 1 | 1.4 | 10 | 1.9 | 38 | 5.8 | 31 | 2.9 | 15 | 4.0 |
| AKAZE | 28 | 4.3 | 0 | 3.6 | 10 | 0.8 | 1 | 4.7 | 0 | 3.4 | 0 | 4.0 | 0 | 1.3 | 1 | 2.7 | 25 | 3.6 | 38 | 13. | 35 | 5.6 | 17 | 6.4 |
| FOCI | 29 | 12. | 0 | 39. | 14 | 11. | 1 | 32. | 0 | 29. | 0 | 29. | 0 | 20. | 1 | 29. | 21 | 13. | 38 | 35. | 35 | 27. | 17 | 45. |
| SFOP | 25 | 11. | 0 | 16. | 12 | 4.7 | 0 | 12. | 0 | 10. | 0 | 10. | 0 | 9.2 | 0 | 7.5 | 11 | 12. | 36 | 15. | 24 | 11. | 8 | 17. |
| WADE | 16 | 14. | 0 | 20. | 0 | 3.4 | 0 | 58. | 0 | 11. | 0 | 14. | 0 | 7.9 | 1 | 8.3 | 20 | 23. | 34 | 60. | 34 | 46. | 13 | 77. |

Hessian-Affine [153] (implementation [185]), FOCI [300], IIDOG [277], WADE [215], W α SH [272], SURF [26], SFOP [76], AKAZE[7]. We focus on getting a reliable answer to the "match/non-match" question in real image pairs. Therefore, the performance criterion is the number of successfully matched pairs using the best combination of descriptors (see Section **Descriptor evaluation**) – rSIFT and hrSIFT. Image pairs are considered matched if ≥ 15 correct inliers to a homography are found. Since the Lost-in-past dataset contains 2300 matchable image pairs, which is unfeasible for direct matching, we have selected a subset of 172 medium-challenging image pairs. Other datasets are used fully.

Adaptive threshold of the detector response. One of the main problems in the matching of day to night and infrared images is the low number of detected features. The problem is acute in dark low contrast images in the WGsBS and MMS [4] datasets. A possible approach addressing the problem is iiDoG [277] where the difference of Gaussians is normalized by the sum of Gaussians. It works well, but cannot be easily applied for other types of detectors, e.g., MSER.

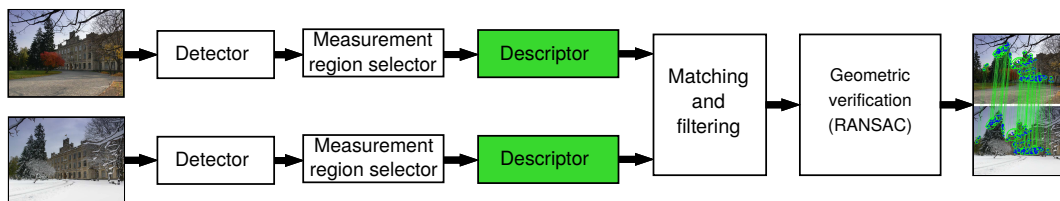
Instead, we propose to use the following adaptive thresholding for all feature detectors. First, all local extrema of the response function are detected (i.e., no thresholding occurs). Next, the detected features are sorted according to the response magnitude. If the number of detected features with response magnitude $\geq \Theta$ is greater than a given threshold R_{\min} , these are output and the algorithm terminates (this is the standard approach). If there are not enough features above the threshold, top R_{\min} features are outputted.

The results are summarized in Table 2.3. Note that none of the methods was able to match almost any image pair which combines more nuisance factors.

Results in Table 2.3 confirm that the proposed adaptive thresholding strategy works as well as, or even better, than iiDoG for DoG, but it is 1.5 times faster. It also significantly improves the results of the MSER and Hessian-Affine, even when the main nuisance is in the viewing geometry (EVD dataset).

CHAPTER 3

HardNet local feature descriptor



In this Chapter we focus on descriptor learning and, using a novel method, train a convolutional neural network (CNN), called HardNet. We additionally show that our learned descriptor outperforms both hand-crafted and learned descriptors in real-world tasks like image retrieval and two-view matching under extreme conditions. We explore possible architectural, training and data choices in Section 3.4 further improving HardNet descriptor.

3.1 Related work

Classical local feature descriptors. A local feature descriptor is a mapping from a (small) image to a metric space, which is robust to changes in acquisition conditions so that descriptors related to the same 3D points are similar and dissimilar otherwise [199].

Early local descriptors include color pixel histograms[249], eigenvalues of the patch [269], grayscale histograms [223], later replaced by grayscale invariants [225] which were inspired by geometry representation in biological systems [119].

The next big step in local patch description was the SIFT [140] – a histogram of gradient orientations. Most of the hand-crafted descriptors after this exploited and extended the same idea of aggregating image gradients – RootSIFT [14], PCA-SIFT [112], HoG [58], RootSIFT-PCA [42], DSP-SIFT [65], ConvOpt [235]. Another, less popular family is the intensity-order based descriptors – LIOP [296], MROGH [72], etc. For real-time applications, a series of binary descriptors – local binary pattern [180] – based on pairwise intensity comparisons was developed. The most popular are BRIEF [43], its version ORB [212], FREAK [6] and so on. They are orders of magnitude faster than SIFT, but worse in terms of accuracy and robustness.

Learned local feature descriptors. Simonyan and Zisserman [235] proposed a simple filter plus pooling scheme learned with convex optimization to replace the hand-crafted filters and poolings in SIFT. Han *et al.* [85] proposed a two-stage Siamese architecture – for embedding and for two-patch similarity. The latter network improved the matching performance, but prevented the use of fast approximate nearest neighbor algorithms like kd-tree [171]. Zagoruyko and Komodakis [289] have independently presented a similar Siamese-based method which explored different convolutional architectures. Simo-Serra *et al.* [234] harnessed hard-negative mining with a relative shallow architecture that exploited pair-based similarity.

The next three papers have followed the classical SIFT matching scheme in the training procedure. Balntas *et al.* [18] used a triplet margin loss and a triplet distance loss, with random sampling of the patch triplets. They show the superiority of the triplet-based architecture over a pair-based. Although, unlike SIFT matching or our work, they sampled negatives randomly. Choy *et al.* [47] calculate the distance matrix for mining positive as well as negative examples, followed by pairwise contrastive loss.

Tian et al [288] proposed a descriptor called L2Net. L2Net uses n matching pairs in batch for generating $n^2 - n$ negative samples and requires that the distance to the ground truth matches is minimum in each row and column. No other constraint on the distance or distance ratio is enforced. Instead, they propose a penalty for the correlation of the descriptor dimensions and adopt deep supervision [127] by using intermediate feature maps for matching. Given the good performance of the L2Net, we have adopted its architecture as a base for our descriptor. We show that it is possible to learn an even more powerful descriptor with a significantly simpler learning objective without the need of two auxiliary loss terms.

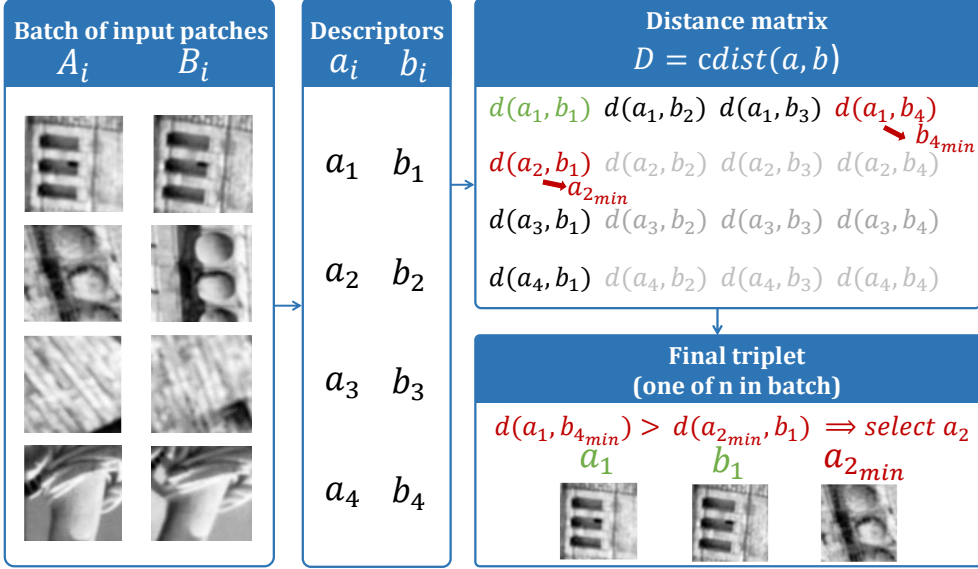


Figure 3.1: HardNet sampling procedure. First, patches are described by the descriptor network, then a distance matrix is calculated. The closest non-matching descriptor – shown in red – is selected for each A_i and B_i patch from positive pair (green) respectively. Finally, among two negative candidates the hardest one is chosen. All operations are done in a single forward pass.

3.2 The HardNet descriptor

3.2.1 Sampling and loss

Our learning objective mimics SIFT matching criterion. The process is shown in Figure 3.1.

First, a batch $\mathcal{X} = (A_i, B_i)_{i=1..n}$ of matching local patch pairs is generated. Both patches from a pair (A_i, B_i) correspond to the same point on the 3D surface. One could sample not pairs, but bigger tuples of the matching patches, however, we have not observed the benefits of doing so. We make sure that in batch \mathcal{X} , there is exactly one pair originating from a given 3D point.

Second, the $2n$ patches in \mathcal{X} are passed through the network as shown in Figure 3.2.

L2 pairwise distance matrix $D = \text{cdist}(a, b)$, where $d(a_i, b_j) = 2\sqrt{1 - a_i b_j}$, $i = 1..n, j = 1..n$ of size $n \times n$ is calculated, where a_i and b_j denote the descriptors of patches A_i and B_j respectively.

Next, for each descriptor from the matching pair (a_i, b_i) the closest non-matching descriptors are found.

$$\begin{aligned}
 j_{min} &= \arg \min_{j=1..n, j \neq i} d(a_i, b_j), \\
 k_{min} &= \arg \min_{k=1..n, k \neq i} d(a_k, b_i)
 \end{aligned} \tag{3.1}$$

Then from each quadruplet of descriptors $(a_i, b_i, b_{j_{min}}, a_{k_{min}})$, a triplet is formed: $(a_i, b_i, b_{j_{min}})$, if $d(a_i, b_{j_{min}}) < d(a_{k_{min}}, b_i)$ and $(b_i, a_i, a_{k_{min}})$ otherwise.

Our goal is to minimize the distance between the matching descriptor and the closest non-matching descriptor. These n triplet distances are fed into the triplet margin loss:

$$L = \frac{1}{n} \sum_{i=1,n} \max(0, 1 + d(a_i, b_i) - \min(d(a_i, b_{j_{min}}), d(a_{k_{min}}, b_i))) \quad (3.2)$$

where $\min(d(a_i, b_{j_{min}}), d(a_{k_{min}}, b_i))$ is precomputed during the triplet construction.

The distance matrix calculation is done on GPU and the only overhead compared to the random triplet sampling is the distance matrix calculation and calculating the minimum over rows and columns. Moreover, compared to usual learning with triplets, our scheme needs only two-stream CNN, not three, which results in 30% less memory consumption and computations.

We experienced no significant overfitting with such procedure, unlike L2Net [288]. This allowed us to get rid of and a constraint on the correlation of descriptor dimensions and simplify the training and implementation.

3.2.2 HardNet architecture

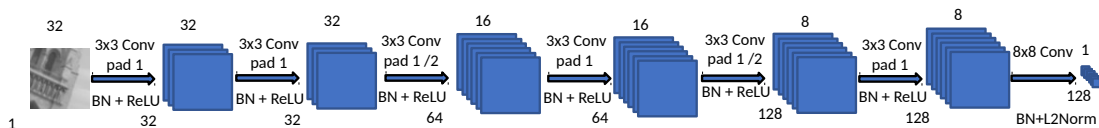


Figure 3.2: HardNet architecture, adopted from L2Net [288]. Each convolutional layer is followed by batch normalization + ReLU, except the last one, which is followed by batch normalization and L2 normalization. Dropout regularization is used before the last convolution layer. Pad 1 – symmetrical zero padding, /2 stands for stride 2. Numbers on the top denote number of channels, in the bottom – spatial size.

The HardNet architecture, shown in Figure 3.2, is identical to L2Net [288]. Padding with zeros is applied to all convolutional layers, to preserve the spatial size, except for the final one. There are no pooling layers, since we found that they decrease performance of the descriptor. That is why the spatial size is reduced by strided convolution. Batch normalization [99] layer followed by ReLU [177] non-linearity is added after each layer, except the last one, where ReLU is replaced with L2 normalization. Thus, the output of the network produces 128-D descriptor with a unit length. Dropout [241] regularization with 0.3 dropout rate is applied before the last convolution layer. Grayscale input patches with size 32×32 pixels are normalized by subtracting the per-patch mean and dividing by the per-patch standard deviation.

Optimization is done by stochastic gradient descent with a learning rate of 10.0 per batch size 1024, momentum of 0.9, and weight decay of 0.0001. Learning rate was linearly decayed to zero within 10 epochs for most of the experiments in this Chapter, as recommended in [165]. Weights were initialized to orthogonally [222] with a gain equal to 0.6, biases set to 0.01. Training is done with PyTorch library [183].

The original version of HardNet was trained with slightly different hyperparameters: learning rate 0.1 and dropout rate 0.1. We present the results for both versions in Figure 3.3 and Table 3.1.

3.2.3 Model training

UBC Phototour [39], also known as Brown dataset, is used for the training of the base HardNet. UBC Phototour consists of three subsets: *Liberty*, *Notre Dame* and *Yosemite* with about 400k normalized 64×64 patches in each. Keypoints were detected by DoG detector and verified by the 3D model.

Test set consists of 100k matching and non-matching pairs for each sequence. Common setup is to train the descriptor on one subset and test on two others. Metric is the false positive rate (FPR) at a point of 0.95 true positive recall. It was found out by Michel Keller that [85] and [18] evaluation procedure reports FDR (false discovery rate) instead of FPR (false positive rate). To avoid the incomprehension of results we’ve decided to provide both FPR and FDR rates and re-estimated the scores for straight comparison. Results are shown in Table 3.1. The proposed descriptor outperforms competitors, with training augmentation, or without it.

In the rest of the Chapter we use a descriptor trained on *Liberty* sequence, unless stated otherwise.

Table 3.1: Patch correspondence verification performance on the Brown dataset. We report a false positive rate at the true positive rate equal to 95% (FPR95). **Some papers report false discovery rate (FDR) instead of FPR due to a bug in the source code.** For consistency, we provide FPR, either obtained from the original article or re-estimated from the given FDR (marked with *). The best results are in **bold**.

| Training | Notredame | Yosemite | Liberty | Yosemite | Liberty | Notredame | Mean | |
|---|-------------|-------------|-------------|-------------|-------------|-------------|------|-------------|
| Test | Liberty | | Notredame | | Yosemite | | FDR | FPR |
| SIFT [140] | 29.84 | | 22.53 | | 27.29 | | | 26.55 |
| MatchNet*[85] | 7.04 | 11.47 | 3.82 | 5.65 | 11.6 | 8.7 | 7.74 | 8.05 |
| TFeat-M* [18] | 7.39 | 10.31 | 3.06 | 3.8 | 8.06 | 7.24 | 6.47 | 6.64 |
| PCW [173] | 7.44 | 9.84 | 3.48 | 3.54 | 6.56 | 5.02 | | 5.98 |
| L2Net [288] | 3.64 | 5.29 | 1.15 | 1.62 | 4.43 | 3.30 | | 3.24 |
| HardNetNeurIPS | 3.06 | 4.27 | 0.96 | 1.4 | 3.04 | 2.53 | 3.00 | 2.54 |
| HardNet | 1.47 | 2.67 | 0.62 | 0.88 | 2.14 | 1.65 | | 1.57 |
| Augmentation: flip, 90° random rotation | | | | | | | | |
| GLoss+[122] | 3.69 | 4.91 | 0.77 | 1.14 | 3.09 | 2.67 | | 2.71 |
| DC2ch2st+[289] | 4.85 | 7.2 | 1.9 | 2.11 | 5.00 | 4.10 | | 4.19 |
| L2Net+ [288] + | 2.36 | 4.7 | 0.72 | 1.29 | 2.57 | 1.71 | | 2.23 |
| HardNet+NeurIPS | 2.28 | 3.25 | 0.57 | 0.96 | 2.13 | 2.22 | 1.97 | 1.9 |
| HardNet+ | 1.49 | 2.51 | 0.53 | 0.78 | 1.96 | 1.84 | | 1.51 |

3.3 Empirical evaluation

We have extensively evaluated the learned descriptors on real-world tasks like two-view matching and image retrieval, as it is known [18] that good performance on the patch verification task on Brown dataset does not always mean good performance in the nearest neighbor setup and vice versa.

We have selected RootSIFT [14], PCW [173], TFeat-M* [18], and L2Net [288] for direct comparison with our descriptor, as they show the best results on a variety of datasets.

3.3.1 Patch descriptor evaluation

HPatches [16] is a dataset for local patch descriptor evaluation. It consists of 116 sequences of 6 images. The dataset is split into two parts: *viewpoint* – 59 sequences with significant viewpoint change and *illumination* – 57 sequences with significant illumination changes, both natural and artificial. Keypoints are detected by DoG, Hessian, and Harris detectors in the reference image and reprojected to the rest of the images in each sequence with 3 levels of geometric noise: *Easy*, *Hard*, and *Tough* variants. The HPatches benchmark defines three tasks: patch correspondence verification, image matching, and small-scale patch retrieval. Patch correspondence verification checks how good descriptors in determination if two (random) patches are in correspondence or not. Image matching and patch retrieval are closer to the real-world applications and check how good is the descriptor in the retrieval of the correct match as a nearest neighbor. The difference between image matching and patch retrieval protocols is that the matching protocol considers only the patches from the single pair of images, while retrieval – from multiple images. We refer the reader to the HPatches paper [16] for a detailed protocol for each task.

Results are shown in Figure 3.3. L2Net and HardNet have shown similar performance on the patch verification task with a small advantage of HardNet. On the matching task, even the non-augmented version of HardNet outperforms the augmented version of L2Net+ by a noticeable margin. The difference is larger in the TOUGH and HARD setups. Illumination sequences are more challenging than the geometric ones for all descriptors. We have trained a network with

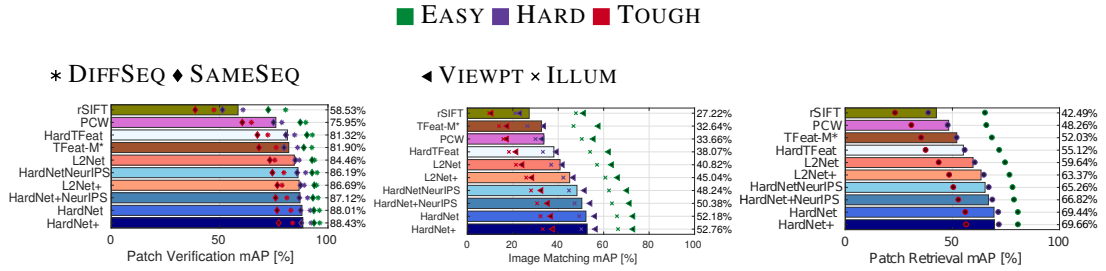


Figure 3.3: Left to right: Verification, matching and retrieval results on HPatches dataset. All the descriptors are trained on *Liberty* subset of Brown [39] dataset, or are handcrafted. Comparison of descriptors trained on other datasets are in Figure 3.4. Marker color indicates the level of geometrical noise in: EASY, HARD and TOUGH. Marker type indicates the experimental setup. DIFFSEQ and SAMESEQ shows the source of negative examples for the verification task. VIEWPT and ILLUM indicate the type of sequences for matching. The HardNet and HardNet+ were trained after the HardNet paper [158] published with optimized hyperparameters learning rate.

TFeat architecture, but with the proposed HardNet loss function – it is denoted as HardTFeat. It outperforms the original version in matching and retrieval, while being on par with it on the patch verification task.

In patch retrieval, the relative performance of the descriptors is similar to the matching problem: HardNet beats L2Net+. Both descriptors significantly outperform the previous state-of-the-art, showing the superiority of the selected deep CNN architecture over the shallow TFeat model.

We have studied the impact of the training dataset on descriptor performance. In particular, we compared the commonly used *Liberty* subset with patches extracted by DoG detector, the full Brown dataset – subsets *Liberty*, *Notredame* and *Yosemite* with patches, extracted by DoG and Harris detectors. We also included results by Mitra *et al.*, who trained HardNet on a new large-scale PhotoSync [166] dataset. Results are shown in Figure 3.4.

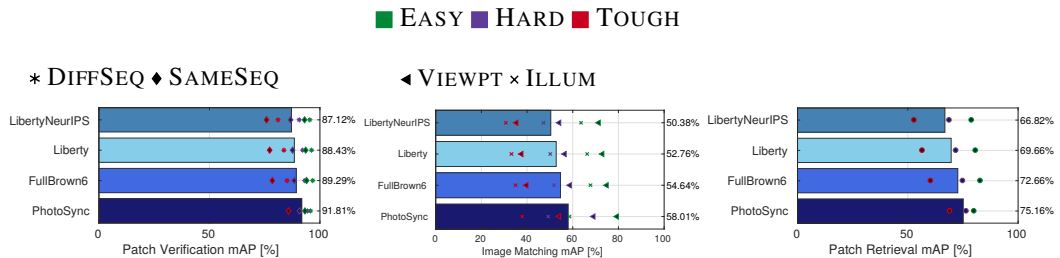


Figure 3.4: Comparison of HardNet versions, trained on different datasets. LibertyNeurIPS, Liberty – *Liberty* [39], FullBrown6 – all six subsets of Brown dataset [39], PhotoSync – dataset from Mitra *et al.* [166]. Left to right: Verification, matching and retrieval results on HPatches dataset. Marker color indicates the level of geometrical noise in: EASY, HARD and TOUGH. Marker type indicates the experimental setup. DIFFSEQ and SAMESEQ shows the source of negative examples for the verification task. VIEWPT and ILLUM indicate the type of sequences for matching.

3.3.2 Ablation study

For a better understanding of the significance of the sampling strategy and the loss function, we conduct the experiments summarized in Table 3.2. We train our HardNet model (architecture is the same as L2Net model), change one parameter at a time, and evaluate its impact.

The following sampling strategies are compared: random, the proposed “hardest-in-batch”, and “classical” hard negative mining, i.e., selecting in each epoch the closest negatives from the full training set. The following loss functions are tested: softmax on distances, triplet margin with margin $m = 1$, contrastive with margins $m = 1, m = 2$. The last is the maximum possible distance for unit-normed descriptors. Mean mAP for HPatches Matching task is shown in Table 3.2.

Table 3.2: Comparison of the loss functions and sampling strategies on the HPatches matching task, the mean mAP is reported. CPR stands for the regularization penalty of the correlation between descriptor channels, as proposed in [288]. Hard negative mining is performed once per epoch. Best results are in **bold**. HardNet uses the hardest-in-batch sampling and the triplet margin loss.

| Sampling / Loss | Softmin | Triplet margin | | |
|-------------------------|---------|----------------|---------|--------------|
| | | $m = 1$ | $m = 1$ | $m = 2$ |
| Random | | | overfit | |
| Hard negative | | | overfit | |
| Random + CPR | 0.349 | 0.286 | 0.007 | 0.083 |
| Hard negative + CPR | 0.391 | 0.346 | 0.055 | 0.279 |
| Hardest in batch (ours) | 0.474 | 0.482 | 0.444 | 0.482 |

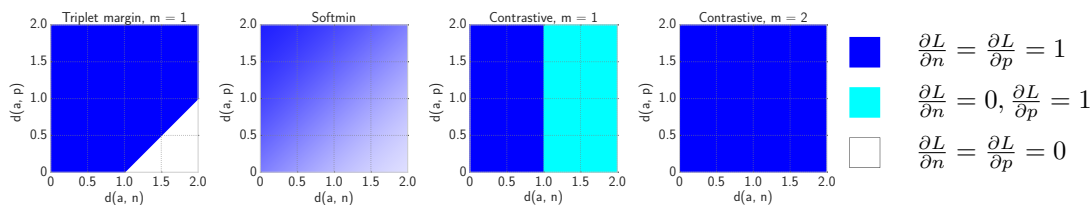


Figure 3.5: Contribution to the gradient magnitude from the positive and negative examples. Horizontal and vertical axes show the distance from the anchor (a) to the negative (n) and positive (p) examples respectively. Softmin loss gradient quickly decreases when $d(a, n) > d(a, p)$, unlike the triplet margin loss. For the contrastive loss, negative examples with $d(a, n) > m$ contribute zero to the gradient. The triplet margin loss and the contrastive loss with a big margin behave very similarly.

The proposed “hardest-in-batch” clearly outperforms all other sampling strategies for all loss functions and it is the main reason for HardNet good performance. The random sampling and “classical” hard negative mining led to huge overfit, when the training loss was high, but the test performance was low and varied several times from run to run. This behavior was observed with all loss functions. Similar results for random sampling were reported in [288].

The poor results of hard negative mining (“hardest-in-the-training-set”) are surprising. We guess that this is due to dataset label noise, the mined “hard negatives” are actually positives. Visual inspection confirms this. We were able to get reasonable results with random and hard negative mining sampling only with an additional correlation penalty on descriptor channels (CPR), as proposed in [288].

Regarding the loss functions, softmin gave the most stable results across all sampling strategies, but it is marginally outperformed by contrastive and triplet margin loss for our strategy. One possible explanation is that the triplet margin loss and contrastive loss with a large margin have constant non-zero derivative w.r.t both positive and negative samples, see Figure 3.5. In the case of contrastive loss with a small margin, many negative examples are not used in the optimization (zero derivatives), while the softmin derivatives become small once the distance to the positive example is smaller than to the negative one.

3.3.3 Wide baseline stereo

To validate the descriptors generalization and their ability to operate in extreme conditions, we tested them on the W1BS dataset [163], which is described in Chapter 2. It consists of 40 image pairs with one particular extreme change between the images: **Appearance (A)**: difference in appearance due to seasonal or weather change, occlusions, etc; **Geometry (G)**: difference in scale, camera and object position; **Illumination (L)**: significant difference in intensity, wavelength of light source; **Sensor (S)**: difference in sensor data (IR, MRI).

Local features in W1BS dataset are detected with MSER [148], Hessian-Affine [153] (in implementation from [185]) and FOCI [300] detectors. They fire on different local structures than

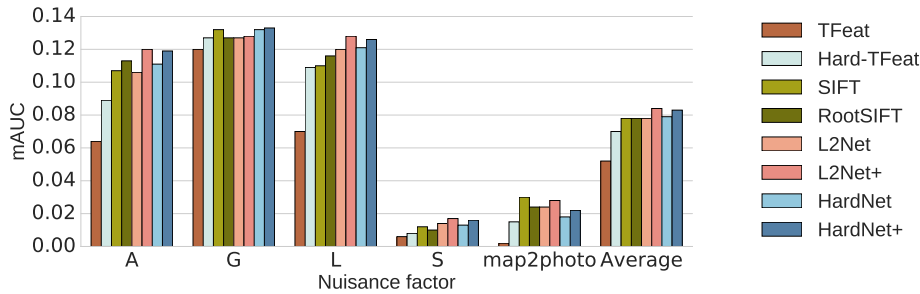


Figure 3.6: Descriptor evaluation on the W1BS patch dataset, mean area under precision-recall curve is reported. Letters denote nuisance factor, A: appearance; G: viewpoint/geometry; L: illumination; S: sensor; map2photo: satellite photo vs. map.

Table 3.3: Comparison of the descriptors on wide baseline stereo within MODS matcher[163] on wide baseline stereo datasets. Number of matched image pairs and average number of inliers are reported. Numbers in the header corresponds to the number of image pairs in dataset.

| Descriptor | EF | | EVD | | OxAff | | SymB | | GDB | | WxBS | | LTLL | |
|------------|-----------|-----------|-----------|-----------|-----------|------------|-----------|-----------|-----------|-----------|-----------|-----------|------------|-----------|
| | 33 | inl. | 15 | inl. | 40 | inl. | 46 | inl. | 22 | inl. | 37 | inl. | 172 | inl. |
| RootSIFT | 33 | 32 | 15 | 34 | 40 | 169 | 45 | 43 | 21 | 52 | 11 | 93 | 123 | 27 |
| TFeat-M* | 32 | 30 | 15 | 37 | 40 | 265 | 40 | 45 | 16 | 72 | 10 | 62 | 96 | 29 |
| L2Net+ | 33 | 34 | 15 | 34 | 40 | 304 | 43 | 46 | 19 | 78 | 9 | 51 | 127 | 26 |
| HardNet+ | 33 | 35 | 15 | 41 | 40 | 316 | 44 | 47 | 21 | 75 | 11 | 54 | 127 | 31 |

DoG. Note that DoG patches were used for the training of the descriptors. Another significant difference to the HPatches setup is the absence of geometrical noise: all patches are perfectly reprojected to the target image in pair. The testing protocol is the same as for the HPatches matching task.

Results are shown in Figure 3.6. HardNet and L2Net perform comparably, the former is performing better on images with geometrical and appearance changes, while the latter works a bit better in map2photo and visible-vs-infrared pairs. Both outperform SIFT, but only by a small margin. However, considering the significant amount of domain shift, the descriptors perform very well, while TFeat loses badly to SIFT. HardTFeat significantly outperforms the original TFeat descriptor on the W1BS dataset, showing the superiority of the proposed loss.

Good performance on the patch matching and verification task does not automatically lead to the better performance in practice, e.g., to more images registered. Therefore, we also compared the descriptors in the wide baseline stereo setup with two metrics: the number of successfully matched image pairs and the average number of inliers per matched pair, following the matcher comparison protocol from [163]. The only change to the original protocol is that the first fast matching step with ORB detector and descriptor was removed, as we are comparing “SIFT-replacement” descriptors.

The results are shown in Table 4.3. Results on Edge Foci (EF) [300], Extreme view [162] and Oxford Affine [154] datasets are saturated and all descriptors are good enough for matching all image pairs. HardNet has a slight advantage in a number of inliers per image. The rest of datasets: SymB [93], GDB [282], WxBS [163] and LTLL [74] have one thing in common: image pairs are or from a different domain than the photo (e.g. drawing to drawing) or cross-domain (e.g., drawing to photo). Here HardNet outperforms learned descriptors and is on par with hand-crafted RootSIFT. We would like to note that HardNet was not learned to match in different domains, nor cross-domain scenarios, therefore such results show the generalization ability.

3.3.4 Image retrieval

We evaluate our method and compare it against the related ones, on the practical application of image retrieval with local features. Standard image retrieval datasets are used for the evaluation, *i.e.*, Oxford5k [188] and Paris6k [189] datasets. Both datasets contain a set of images (5062 for Oxford5k and 6300 for Paris6k) depicting 11 different landmarks together with distractors. For each of the 11 landmarks, there are 5 different query regions defined by a bounding box,

Table 3.4: Performance (mAP) evaluation on bag-of-words (BoW) image retrieval. Vocabulary consisting of 1M visual words is learned on independent dataset, that is, when evaluating on Oxford5k, the vocabulary is learned with features of Paris6k and *vice versa*. SV: spatial verification. QE: query expansion. The best results are highlighted in **bold**. All the descriptors except SIFT and HardNet++ were learned on *Liberty* sequence of Brown dataset [39]. HardNet++ is trained on union of Brown and HPatches [16] datasets.

| Descriptor | Oxford5k | | | Paris6k | | |
|---------------|-------------|-------------|-------------|-------------|-------------|-------------|
| | BoW | +SV | +QE | BoW | +SV | +QE |
| TFeat-M* [18] | 46.7 | 55.6 | 72.2 | 43.8 | 51.8 | 65.3 |
| RootSIFT [14] | 55.1 | 63.0 | 78.4 | 59.3 | 63.7 | 76.4 |
| L2Net+ [288] | 59.8 | 67.7 | 80.4 | 63.0 | 66.6 | 77.2 |
| HardNet | 59.0 | 67.6 | 83.2 | 61.4 | 67.4 | 77.5 |
| HardNet+ | 59.8 | 68.8 | 83.0 | 61.0 | 67.0 | 77.5 |
| HardNet++ | 60.8 | 69.6 | 84.5 | 65.0 | 70.3 | 79.1 |

Table 3.5: Performance (mAP) comparison with the state-of-the-art image retrieval with local features. Vocabulary is learned on independent dataset, that is, when evaluating on Oxford5k, the vocabulary is learned with features of Paris6k and *vice versa*. All presented results are with spatial verification and query expansion. VS: vocabulary size. SA: single assignment. MA: multiple assignments. The best results are highlighted in **bold**.

| Method | VS | Oxford5k | | Paris6k | |
|-----------------------|-----|-------------|-------------|-------------|-------------|
| | | SA | MA | SA | MA |
| SIFT-BoW [185] | 1M | 78.4 | 82.2 | – | – |
| SIFT-BoW-fVocab [156] | 16M | 74.0 | 84.9 | 73.6 | 82.4 |
| RootSIFT-HQE [261] | 65k | 85.3 | 88.0 | 81.3 | 82.8 |
| HardNet++-HQE | 65k | 86.8 | 88.3 | 82.8 | 84.9 |

constituting 55 query regions per dataset. The performance is reported as mean average precision (mAP) [188].

In the first experiment, for each image in the dataset, multi-scale Hessian-affine features [155] are extracted. Exactly the same features are described by ours and all related methods, each of them producing a 128-D descriptor per feature. Then, k-means with approximate nearest neighbor [171] is used to learn a 1 million visual vocabulary on an independent dataset, that is, when evaluated on Oxford5k, the vocabulary is learned with descriptors of Paris6k and *vice versa*. All descriptors of the testing dataset are assigned to the corresponding vocabulary, so finally, an image is represented by the histogram of visual word occurrences, *i.e.*, the bag-of-words (BoW) [238] representation, and an inverted file is used for an efficient search. Additionally, spatial verification (SV) [188], and standard query expansion (QE) [189] are used to re-rank and refine the search results. Comparison with the related work on patch description is presented in Table 4.4. HardNet+ and L2Net+ perform comparably across both datasets and all settings, with slightly better performance of HardNet+ on average across all results (average mAP 69.5 vs. 69.1). RootSIFT, which was the best performing descriptor in image retrieval for a long time, falls behind with average mAP 66.0 across all results. We also trained HardNet++ version – with all available training data at the moment: the union of Brown and HPatches datasets, instead of just *Liberty* sequence from Brown for the HardNet+. It shows the benefits of having more training data and is performing best for all setups.

Finally, we compare our descriptor with the state-of-the-art image retrieval approaches that use local features. For fairness, all methods presented in Table 4.5 use the same local feature detector as described before, learn the vocabulary on an independent dataset, and use spatial verification (SV) and query expansion (QE). In our case (HardNet++-HQE), a visual vocabulary of 65k visual words is learned with an additional Hamming embedding (HE) [107] technique that further refines descriptor assignments with a 128 bit binary signature. We follow the same procedure as RootSIFT-HQE [261] method, by replacing RootSIFT with our learned HardNet++

Table 3.6: Evaluation of models found by the DARTS algorithm, one model per row. Several trainings were run due to the recommendation in [138]. HPatches and AMOS Patches – Mean average precision (mAP). IMC – mean average accuracy (mAA) at 10°.

| Architecture | HPatches | AMOS Patches | IMW PT |
|--------------|--------------|--------------|--------------|
| DARTS 1 | 36.56 | 26.42 | 57.38 |
| DARTS 2 | 28.48 | 17.41 | 46.53 |
| DARTS 3 | 24.54 | 13.36 | 40.76 |
| HardNet | 52.96 | 44.21 | 68.17 |

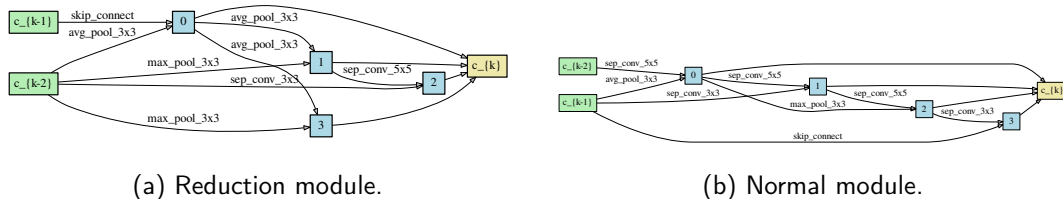


Figure 3.7: Modules found by the DARTS algorithm visualized using the provided tool [138].

descriptor. Specifically, we use: (i) weighting of the votes as a decreasing function of the Hamming distance [106]; (ii) burstiness suppression [106]; (iii) multiple assignments of features to visual words [189, 108]; and (iv) QE with feature aggregation [261]. All parameters are set as in [261]. The performance of our method is the best reported on both Oxford5k and Paris6k when learning the vocabulary on an independent dataset (mAP 89.1 was reported [14] on Oxford5k by learning it on the same dataset comprising the relevant images), and using the same amount of features (mAP 89.4 was reported [261] on Oxford5k when using twice as many local features, *i.e.*, 22M compared to 12.5M used here).

3.4 Exploring HardNet design choices [199]

This Section contains various experiments with HardNet design choices, performed by Milan Pultar during his master studies [199] under my supervision. The IMC benchmark, extensively used here, is described in Chapter 6. AMOS patches dataset is described in [198].

3.4.1 Architecture

Architecture of a model is one of the paramount factors influencing the performance of the descriptor. A lot of work in the field of computer vision is focused on discovering architectures that achieve higher performance, are faster or more compact. However, there is a paucity of literature available describing new architectures for a local feature descriptor. Here we explore different architectural choices beyond original VGG-style L2Net architecture.

3.4.2 Automated Search

Differentiable Architecture Search (DARTS) [138] is an algorithm which searches a space of architectures formulated in a continuous manner by using gradient descent. The bilevel optimization problem is expressed in [138] as follows:

$$\begin{aligned} \min_{\alpha} \mathcal{L}_{val}(w^*(\alpha), \alpha), \\ \text{s.t. } w^*(\alpha) = \arg \min_w \mathcal{L}_{train}(w, \alpha), \end{aligned} \quad (3.3)$$

where \mathcal{L}_{train} and \mathcal{L}_{val} is the loss function on the training and validation set, α is the architecture and w are the associated weights. The search space is represented by a sequence of cells, which are composed of k ordered nodes. The input to each node (called a connection or edge) is equal to the sum of its two predecessors. The output of a cell is defined as a concatenation of all its

Table 3.7: Variants of the HardNet. Each convolutional layer is followed by batch normalization and ReLU.

| HardNet | HardNet7x2 | HardNet8 | HardNet8x2 | HardNet9 |
|-----------------------------|----------------|----------------|----------------|----------------|
| 1.3M param | 5.3M param | 4.7M param | 19M param | 5.3 param |
| Block 1, spatial size 32x32 | | | | |
| Conv 3x3x32 | Conv 3x3x64 | Conv 3x3x32 | Conv 3x3x64 | Conv 3x3x32 |
| Conv 3x3x32 | Conv 3x3x64 | Conv 3x3x32 | Conv 3x3x64 | Conv 3x3x32 |
| Block 2, spatial size 16x16 | | | | |
| Conv 3x3x64/2 | Conv 3x3x128/2 | Conv 3x3x64/2 | Conv 3x3x128/2 | Conv 3x3x64/2 |
| Conv 3x3x64 | Conv 3x3x128 | Conv 3x3x64 | Conv 3x3x128 | Conv 3x3x64 |
| - | - | - | - | Conv 3x3x128 |
| Block 3, spatial size 8x8 | | | | |
| Conv 3x3x128/2 | Conv 3x3x256/2 | Conv 3x3x128/2 | Conv 3x3x256/2 | Conv 3x3x128/2 |
| Conv 3x3x128 | Conv 3x3x256 | Conv 3x3x128 | Conv 3x3x256 | Conv 3x3x256 |
| - | - | Conv 3x3x256 | Conv 3x3x512 | Conv 3x3x256 |
| Global pooling block | | | | |
| Dropout(0.3) | | | | |
| Conv 8x8x128 | Conv 8x8x256 | Conv 8x8x256 | Conv 8x8x256 | Conv 8x8x256 |
| Flatten(), L2Norm() | | | | |

Table 3.8: Evaluation of the variants of the HardNet architecture. HPatches and AMOS Patches – Mean average precision (mAP). IMC – mean average accuracy (mAA) at 10°.

| Architecture | HPatches | AMOS Patches | IMC |
|--------------|--------------|--------------|--------------|
| HardNet | 52.96 | 44.21 | 68.17 |
| HardNet7x2 | 54.56 | 44.39 | 68.67 |
| HardNet8 | 53.67 | 43.77 | 69.43 |
| HardNet8x2 | 54.55 | 42.96 | 69.18 |
| HardNet9 | 54.21 | 42.85 | 69.34 |

nodes. The edges between the nodes correspond to the operations that are to be learned. During the architecture search each of these are assigned parameters that are weighted using the softmax operator.

After the search procedure is finished, the operation corresponding to the highest weight is assigned to each edge. It is recommended to use a larger number of cells at this point due to smaller resource requirements.

In our experiments we use the publicly available implementation¹ and plug in our data loaders, so that Liberty and Notredame from UBC Phototour [213] is the training and validation dataset respectively. We run the architecture search for 40 epochs, 100 000 samples each, learning rate = 2.5, batch size = 64. Other settings are kept default.

Subsequent training is performed for 40 epochs, 500 000 tuples each, with batch size = 128 and learning rate = 0.025. In Table 3.6 we list the results for several runs and compare them with HardNet. In Figure 3.7 we present the found cells constituting the architecture DARTS 1. Reduction module is used in two positions in the architecture (1/3 and 2/3 of the length), all other cells are normal. The results of the DARTS models are significantly worse than those of HardNet. It might be, for example, due to the small batch size – higher is not feasible due to GPU memory. Also, the method could be improved by changing the search space. We leave such modifications for future work.

3.4.3 Manual Search

VGG [236] Style HardNet is a VGG style network with blocks of convolutional layers of decreasing spatial size and an increasing number of channels. Here we introduce and evaluate several modifications to the architecture. With respect to the original version, HardNet7x2

¹Available at <https://github.com/quark0/darts>

Table 3.9: ResNet architectures. Each convolutional layer is followed by batch normalization and ReLU.

| ResNet2 | ResNet3 | ResNet3x |
|-----------------------------|----------------|----------------|
| 1.7 param | 2M params | 2.1M params |
| Block 1, spatial size 32x32 | | |
| Conv 3x3x64 | Conv 3x3x32 | Conv 3x3x32 |
| Conv 3x3x64 | Conv 3x3x32 | Conv 3x3x64 |
| Block 2, spatial size 16x16 | | |
| ResBlock2/2 32 | ResBlock3/2 16 | ResBlock3/2 32 |
| ResBlock2 64 | ResBlock3 32 | ResBlock3 64 |
| Block 3, spatial size 8x8 | | |
| ResBlock2/2 64 | ResBlock3/2 64 | ResBlock3/2 64 |
| ResBlock2 128 | ResBlock3 128 | ResBlock3 128 |
| Global pooling block | | |
| Dropout(0.3) | | |
| Conv 8x8x256 | Conv 8x8x256 | Conv 8x8x128 |
| Flatten(), L2Norm() | | |

Table 3.10: Evaluation of the ResNet architectures together with the baseline HardNet. HPatches and AMOS Patches – Mean average precision (mAP). IMC – mean average accuracy (mAA) at 10°.

| Architecture | HPatches | AMOS Patches | IMW PT |
|--------------|--------------|--------------|--------------|
| ResNet2 | 52.63 | 40.57 | 68.31 |
| ResNet3 | 52.96 | 40.72 | 69.01 |
| ResNet3x | 52.25 | 38.50 | 68.27 |
| HardNet8 | 53.67 | 43.77 | 69.43 |

contains two times more channels in each layer, HardNet8 has one convolutional layer added in the third block, and HardNet9 contains two more layers, one in the second and one on the third block. HardNet8x2 has two times more channels in each layer than HardNet8. See Table 3.7 for a detailed architecture description. In Table 3.8 we list the results for each architecture.

ResNet [95] Style Another popular architecture type is ResNet. It consists of blocks of convolutional layers called ResBlock. The output from each block is summed with its input, so the network learns an incremental change instead of a direct transformation. In Table 3.9 we list the architectures we tested. Evaluation of these models is shown in Table 3.10. Each model was trained on the Liberty dataset for 20 epochs, 5 million samples each. The results are almost on par with those of HardNet models. However, in none of our experiments it was superior and it has a higher GPU memory requirement, so we can not advise to use this architecture type for the learning of a local feature descriptor.

3.4.4 Batch size

Batch size is an important factor in the setup of the training procedure. See in Figure 3.10 the influence of the batch size on the performance of the trained descriptor. Notice that there is an increase in performance, measured by the mAP score on AMOS Patches and HPatches benchmarks, if we increase the batch size. It is likely that a higher batch size would be even more beneficial, but we are limited by the GPU memory. On a 32 GB machine, the maximum batch size is approximately 8192. The best performance on the IMC benchmark is achieved for the batch size around 4096. The IMC benchmark is described in detail in Chapter 6. Interestingly, a further increase worsens the results on both HPatches and IMC.

Input Size The input to the HardNet architecture is of size 32×32 pixels. Due to the observation that enlarging the model architecture further from HardNet8 does not lead to better performance, we perform the following experiment. We make minor adjustments to the HardNet8 architecture

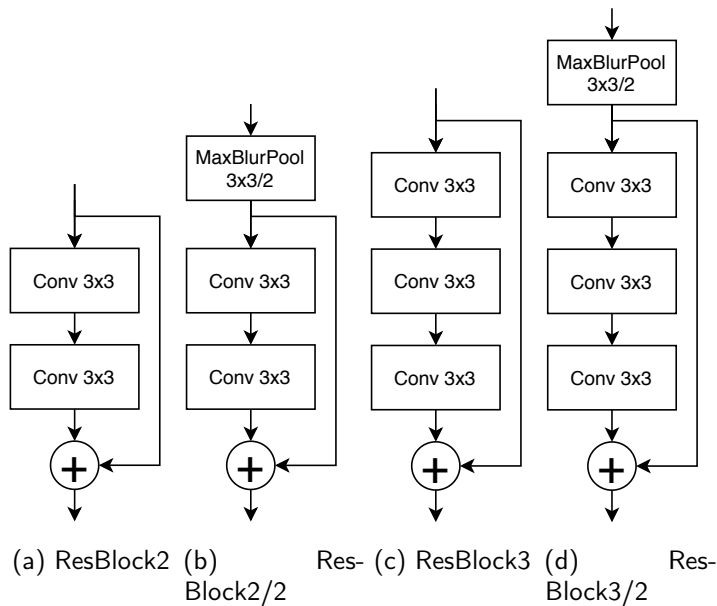


Figure 3.8: ResNet modules.

Table 3.11: Evaluation of HardNet variants with different input size. HPatches and AMOS Patches – Mean average precision (mAP). IMC benchmark – mean average accuracy (mAA) at 10° .

| Model | HPatches | AMOS Patches | IMC |
|---------------------------|--------------|--------------|--------------|
| HardNet8 | 52.37 | 42.42 | 69.06 |
| HardNet8, input size = 48 | 53.34 | 43.12 | 68.41 |
| HardNet8, input size = 64 | 54.19 | 43.98 | 68.25 |

so that it expects inputs of a bigger size. First, we change the stride to 2 in the penultimate layer, then the input size is 64×64 pixels. If we also set the kernel size of the last convolutional layer to 6×6 pixels, the input size is then 48×48 pixels. We compare these three models in Table 3.11. Notice the two opposite trends: bigger input size leads to better performance on HPatches and AMOS Patches, but such a model gives inferior results on the IMC benchmark.

3.4.5 Margin value

The purpose of this experiment is to determine the best value for the margin in the loss function. In Figure 3.9 we can see that the performance roughly increases with the margin on both the CVPR and HPatches benchmarks. We can observe that the optimal value for the margin is around 0.5 in case of the IMC benchmark. In HPatches the difference in evaluation between margins above 0.4 is almost negligible.

3.4.6 Final Pooling

The HardNet architecture uses global pooling implemented by a $8 \times 8 \times 128$ convolution as the last layer in the model – i.e. in fact a fully connected linear layer – unlike SIFT, which uses local pooling, where the output vector is composed of 16 blocks based on the relative position in the input patch. In this section we compare different architectures and try several pooling approaches. All models are trained on the Liberty dataset for 20 epochs, 5 million samples each.

Local pooling In this experiment we try to mimic the mechanism of SIFT and replace the last convolutional layer in order to increase its stride so that the convolution is performed over separate blocks. In Table 3.12 we can observe that the performance of these models is worse than of the baseline.

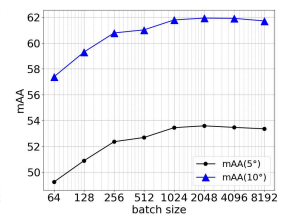
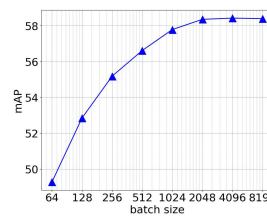
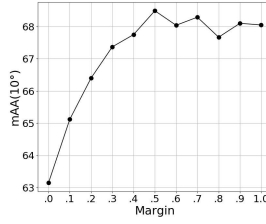
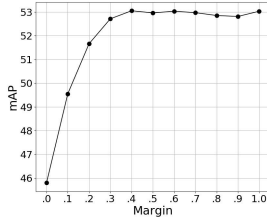


Figure 3.9: HardNet performance depending on the margin applied during training. Left: the training batch size. Left: mAP on HPatches, right: mean average accuracy (mAA) with the threshold of 10° on IMC. Figure 3.10: HardNet performance depending on the margin applied during training. Left: the training batch size. Left: mAP on HPatches, right: mean average accuracy (mAA) with the threshold of 10° on IMC.

Table 3.12: Final pooling variants. Modifications to the HardNet8 architecture by substituting the last convolutional layer are listed. The last row represents no modification. X^y means the layer X is stacked y times. We write n/c for models which failed to learn. HPatches and AMOS Patches – Mean average precision (mAP). IMC – mean average accuracy (mAA) at 10° .

| Final Layer(s) | HPatches | AMOS Patches | IMC |
|--|--------------|--------------|--------------|
| Local Pooling | | | |
| Conv 4x4x16/4 | 53.28 | 44.50 | 68.50 |
| Conv 3x3x16/2, p=1 | 52.76 | 44.78 | 68.36 |
| Increased Receptive Field | | | |
| (Conv 3x3/2, p=1) ¹ → Conv 4x4, p=0 | 52.31 | 42.00 | 68.29 |
| (Conv 3x3/2, p=1) ² → Conv 2x2, p=0 | 50.25 | 39.14 | 67.62 |
| (Conv 3x3/1, p=0) ³ → Conv 2x2, p=0 | n/c | n/c | n/c |
| Non-learned | | | |
| MaxPool 8x8 | 44.23 | 38.58 | 67.11 |
| AvgPool 8x8 | n/c | n/c | n/c |
| Global Pooling (original) | | | |
| Conv 8x8x256 | 53.67 | 43.77 | 69.43 |

Increased Receptive Field We also test modifications to the HardNet8 architecture which increase its receptive field. In Table 3.12 we can observe that such architectures do not outperform the baseline. Enlarging the architecture by 3 layers even fails to learn.

Non-learned Pooling Another possibility is to compute the maximum or average per input channel. MaxPool achieves a worse result and AvgPool fails to learn, see Table 3.12.

3.4.7 Compression of Embeddings

During the manual architecture search, we found that some of the larger models, e.g. HardNet8, achieve better performance. These architectures, however, also output a longer vector, which brings disadvantages such as higher memory usage and slower nearest-neighbour search. If we reduce the output size of HardNet8 to 128 to match the vector length of SIFT, we get inferior results - 68.75 mAA(10°) vs 69.43 mAA(10°) on the IMC benchmark. Another way to decrease the output size is to use a dimensionality reduction technique. Here we use principal component analysis (PCA) to compress the feature embeddings. See in Figure 3.11 that dimensionality reduction is beneficial. We can observe that the best combination is HardNet8 with output size 512 followed by compression to size 128. In this experiment, we train on the Liberty dataset from UBC Phototour and we run PCA for the model outputs from the training data.

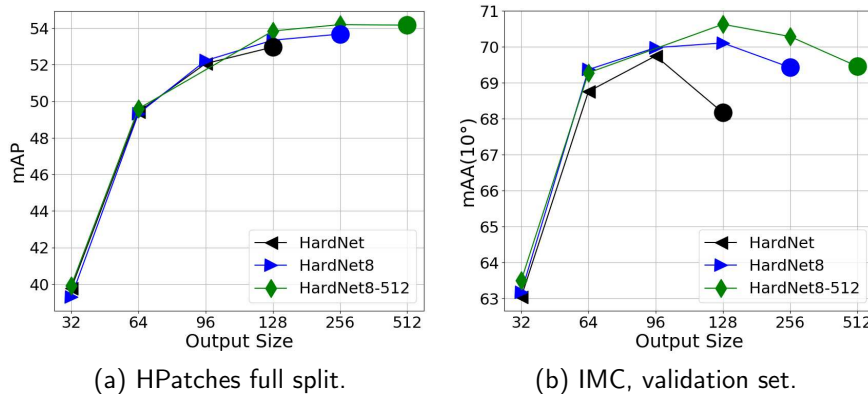


Figure 3.11: Impact of the embeddings compression with PCA. HardNet8-512 denotes HardNet8 with the number of channels in the last layer changed to 512. HPatches and AMOS Patches – Mean average precision (mAP). IMC – mean average accuracy (mAA) at 10°. Original, non-compressed dimension is marked with a circle.

3.4.8 Combining all together

We evaluate two HardNet8 variants which use the improvements described above. The first, HardNet8-Univ, was trained on Liberty and AMOS Patches [198] datasets for 20 epochs, 5 million samples each, batch size = 3072.

The second model, HardNet8-PT, has 512 output channels and was trained on Liberty, Notre Dame and Colosseum Exterior [109] datasets for 40 epochs, 5 million samples each, batch size = 9000. The embeddings from the model are compressed by PCA – which is fitted on Liberty – to dimensionality of 128.

Evaluation is performed on the test set of IMC – the previous experiments were evaluated on the validation set. HPatches and AMOS Patches comprise of a single test split which is used both in the experiments and final evaluation.

Results are shown in Table 3.13. HardNet8-Univ works reasonably well on a wide range of conditions, be it viewpoint or illumination changes, and improves state-of-the-art on standard benchmarks. It is only outperformed by GeoDesc in the viewpoint split of HPatches. HardNet8-PT further improves the performance on the IMC benchmark in the Stereo 8k task.

Apart from benchmarking, we also conjecture that significantly better results can be achieved for a real-world task if HardNet8 is retrained on a specific combination of datasets, suitable for the use case – e.g. AMOS Patches for illumination changes with no viewpoint change, while Liberty, Notre Dame, Colosseum Exterior and others for purely viewpoint changes.

Table 3.13: Final evaluation of the proposed HardNet8 descriptors. HPatches and AMOS Patches – Mean average precision (mAP). IMC – mean average accuracy (mAA) at 10°.

| Model | Trained on | HPatches subset | | | AMOS Patches | IMC Stereo 8k |
|---------------|------------------|-----------------|--------------|--------------|--------------|---------------|
| | | illum | view | full | | |
| SIFT | – | 23.33 | 28.88 | 26.15 | 37.08 | 45.84 |
| HardNet | Liberty | 49.86 | 55.63 | 52.79 | 43.32 | 55.43 |
| SOSNet | Liberty | 50.66 | 56.73 | 53.75 | 43.26 | 55.87 |
| HardNetPS | PS Dataset | 48.55 | 67.43 | 58.16 | 31.83 | 50.51 |
| GeoDesc | GL3d [232] | 50.52 | 67.48 | 59.15 | 25.50 | 51.11 |
| HardNet8-Univ | AMOS+Liberty | 58.20 | 63.60 | 60.95 | 47.20 | 56.22 |
| HardNet8-PT | Lib+Coloss+Notre | 51.04 | 55.81 | 53.46 | 45.01 | 57.58 |

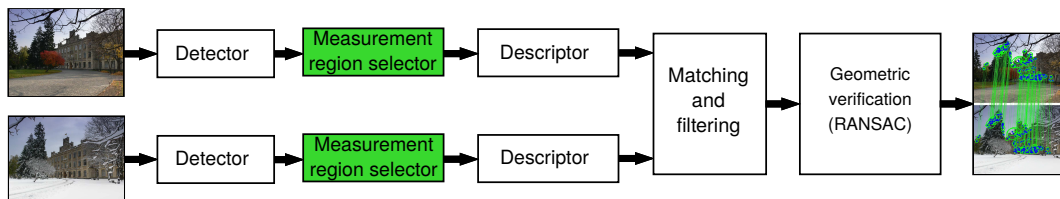
3.5 Summary

We proposed a novel loss function for learning a local image descriptor that relies on the hard negative mining within a mini-batch and the maximization of the distance between the closest positive and closest negative patches. The proposed sampling strategy outperforms classical hard-negative mining and random sampling for softmin, triplet margin, and contrastive losses. We have also evaluated a series of design choices influencing the performance of the HardNet.

The resulting descriptor is compact – it has the same dimensionality as SIFT (128), it shows state-of-art performance on standard matching, patch verification and retrieval benchmarks, and it is fast to compute on a GPU. The training source code and the trained convnets are available at <https://github.com/DagnyT/hardnet>.

CHAPTER 4

Local affine features



This Chapter makes four contributions towards the robust estimation of the local feature affine shape. First, we experimentally show that the geometric repeatability of a local feature is not a sufficient condition for successful matching. The learning of the affine shape increases the number of corrected matches if it steers the estimators towards discriminative regions and therefore must involve the optimization of a descriptor-related loss. Second, we propose a novel loss function for descriptor-based registration and learning, named the *hard negative-constant loss*. It combines the advantages of the triplet and contrastive positive losses. Third, we propose a method for learning the affine shape, orientation, and potentially other parameters related to geometric and appearance properties of local features. The learning method does not require a precise ground truth, which reduces the need for manual annotation. Finally, the learned AffNet itself significantly outperforms prior methods for affine shape estimation and improves the state-of-the-art in BoW-based image retrieval by a large margin. Importantly, unlike the de facto standard [25], AffNet does not significantly reduce the number of detected features, it is thus suitable even for pipelines where affine invariance is needed only occasionally.

4.1 Introduction

Local feature detector finds "image patterns which differ from its immediate neighborhoods" [270]. Local detectors, compared to dense image sampling, serve two purposes: reduce the amount of computation needed for descriptor matching and increase the robustness to occlusions. Similar to the local descriptors, detectors can be divided into handcrafted-vs-learned groups and float-vs-binary kinds.

The earliest detectors are still in use: the Hessian [28] blob detector and the Harris [88] corner detector. Both detectors are based on image gradients. Later, they were extended to multi-scale [135, 136, 68] and affine-covariant [25, 153] versions. Other popular blob detector is Difference-of-Gaussians(DoG), proposed in SIFT paper [140].

Binary detectors that rely on intensity comparisons for finding corner-like structures started with the SUSAN [239] detector. It was then accelerated by the learned order of comparisons for fast rejection – FAST [211] and improved by adding the orientation estimation – ORB [212]. Binary detectors for other than corner structures also exist, e.g. for the detection of saddle points [8, 9].

There are other detectors that do not belong to the abovementioned groups. Most popular of them are the following. MSER [148] uses a segmentation algorithm for detecting regions with boundaries sharing similar pixel intensity. FOCI [300] looks for normalized intensity edge focal points and WASH [272] is based on the Delaunay triangulation of the edge map.

The popular deep learning-based detectors are the following. Yi *et al.* [285] proposed to learn



Figure 4.1: Selected matching SIFT (left) and SIFT-AffNet(right) keypoints and corresponding patches. One can see that not only patch centers correspond to each other, but also other pixels, although less precise.

feature orientation by minimizing descriptor distance between positive patches which correspond to the same point on the 3D surface. This allows to avoid hand-picking a "canonical" orientation, thus learning the one which is most suited for descriptor matching. Yi *et al.* [286] proposed a multistage framework for learning the descriptor, orientation and translation-covariant detector. The detector was learned by maximizing the intersection-over-union and the reprojection error between the corresponding regions. Lenc and Vedaldi [130] introduced the “covariant constraint” for learning various types of local feature detectors. Zhang *et al.* [293] proposed to “anchor” the detected features to some predefined features with known good discriminability like TILDE [274]. Savinov *et al.* [221] proposed a ranking approach for unsupervised learning of a feature detector. Choy *et al.* [47] trained a “Universal correspondence network” (UCN) for a direct correspondence estimation with contrastive loss on a patch descriptor distance, avoiding the detection stage. Superpoint detector and descriptor [62] share a similar idea for training the descriptor, while the detector is trained in a supervised way to find corners and line junctions. R2D2 [207] learn a detector to optimize both the repeatability and “reliability”. Reliability is proposed and called matchability in the AffNet paper [161], which is described in Section 4.4.

4.2 Keypoints are not just points

Local feature detectors are often referred as keypoint detectors and wide baseline stereo matching often is perceived as establishing (key-)point correspondences between images. While this might be true for some local features like SuperPoint [62], typically it is more than that.

Specifically, detectors like DoG [140], Harris [88], Hessian [28], KeyNet [23], ORB[212], and many others run on scale-space provide at least 3 parameters: x , y , and scale.

Most of the local descriptors – SIFT [140], HardNet [158] and so on – are not rotation invariant, so the patch orientation often has to be estimated anyway to match reliably. While this stage can be avoided by designing the descriptor in the rotation-invariant way [5, 30], this often requires a complex matching function [124, 30] and results in less discrimination ability [31].

The rotation estimation is done by various methods: corner center of mass (ORB [212], dominant gradient orientation (SIFT) [140] or by some learned estimator (OriNets [285, 161]). Sometimes it is possible to rely on the smartphone’s IMU or photographer, and assume that the images are upright[185].

Thus, we can assume that if the local descriptors match, this means the local feature scale and orientation also match at least approximately, as illustrated in Figure 4.1. Possible exceptions are cases when the patch is symmetrical and the orientation is ambiguous up to some symmetry group.

In addition, one could assume that we observe the patch not from the fronto-parallel position and try to estimate the local normal, or, more precisely, the affine shape of the feature point, modeling it as an ellipse instead of a circle, as illustrated in Figure 4.1. One could also think of affine shape estimation as finding the camera position, from where the patch is seen in some "canonical" view. This gives us 3 point correspondences from a single local feature match, see an example in Figure 4.2.

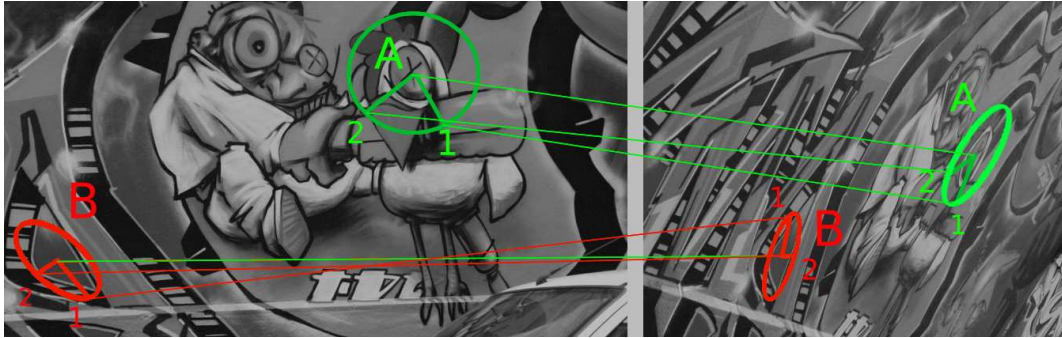


Figure 4.2: Local affine correspondences. While centers of both regions A and B are correct point matches, only A is a correct affine correspondence.

4.3 Benefits of local affine features

4.3.1 Making descriptor job easier

The most straightforward benefit of using local affine features is that they increase the repeatability of the detector and potentially reduce the appearance changes of a local patch caused by viewpoint difference [155]. This makes possible matching more challenging image pairs.

The practice is a little bit more complicated. While the affine-covariance helps the matching under the presence of the large viewpoint change, see Chapter 5, it reduces the local patch discriminativity for the simpler cases. In other words, one needs to be as much invariant or covariant, as needed, but not more. For example, affine-covariant detector helps to match the circle under different angles, but fails when one needs to distinguish between a circle and an ellipse.

Our benchmark [109], see in Chapter 6, which measures the accuracy of the output fundamental matrix, shows that the benefit of using affine instead of similarity-covariant features for the phototourism data might be quite unstable. On the one hand, using AffNet consistently improves multiview camera pose accuracy estimation by 3-5% relatively. On the other hand, in the large number of features setup, DoG-HardNet performs better in stereo than DoG-AffNet-HardNet.

Therefore, if the benefit of local features would be to only improve the descriptor extraction stage for mild viewpoint changes, it might be arguably not worth it. Luckily, there are more benefits, which are more pronounced.

4.3.2 Making RANSAC job easier

Let us recall how RANSAC[75] works.

1. Randomly sample a minimally required number of tentative correspondences to fit the geometrical model of the scene: 4 for homography, 7 for epipolar geometry, etc. and estimate the model.
2. Calculate "support": other correspondences, which are consistent with the model.
3. Repeat steps (1), (2) and output the model which is supported with the most of correspondences. If you were lucky and have sampled all-inlier sample, meaning that all correspondences used to estimate the model were correct, you would have a correct model.

Modern RANSACs[20, 60] are more complicated than we have just described, but the principle is the same. The most important part is the sampling and it is sensitive to the inlier ratio ν - the percentage of the correct correspondences in the set. Let us denote the minimal number of correspondences required to estimate the model as \mathbf{m} . To recover the correct model with the confidence \mathbf{p} one needs to sample the number of correspondences, which is described by formula:

$$N = \frac{\log(1-p)}{\log(1-\nu^{\mathbf{m}})} \quad (4.1)$$

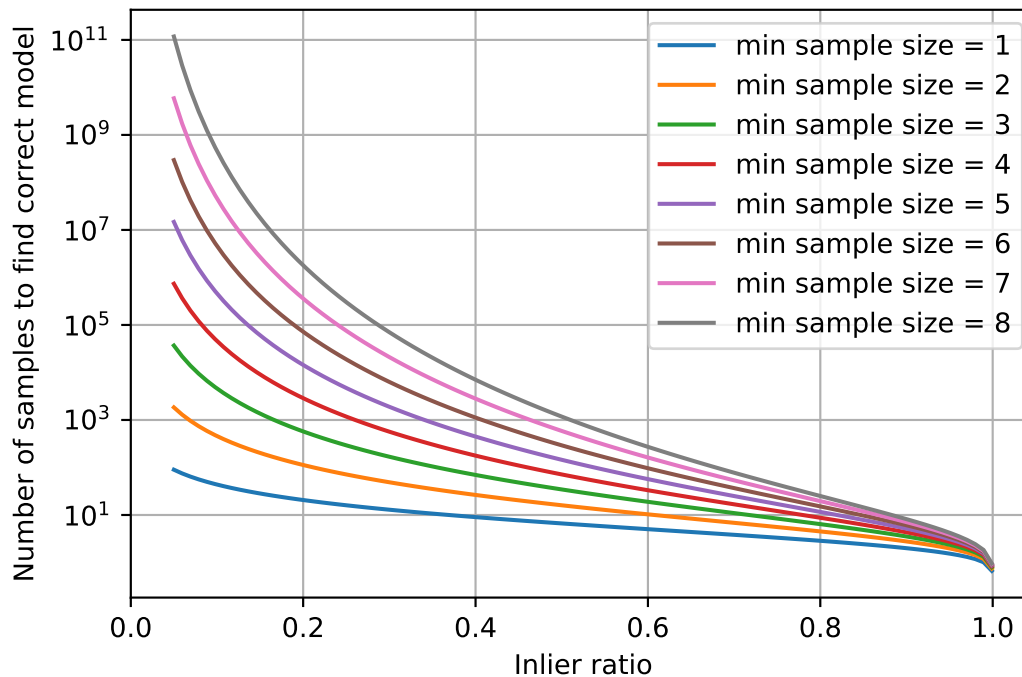


Figure 4.3: Number of samples RANSAC needs to draw to find the correct model with the given probability as function of inlier rate, when confidence rate is 99%.

Lets plot how the number of required samples changes with the inlier ratio for confidence equal 99%, see Figure 4.3. Note the log scale on Y axis. Reducing the minimal sample size required for the model estimation even by 1 saves an order of magnitude of computation. In reality, the benefit is smaller, as modern RANSACs like GC-RANSAC[20] and MAGSAC[60] could estimate the correct model from the sample containing outliers, but it is still substantial, especially for low inlier rate cases.

Image retrieval

The ideal case would be to estimate a model from just a single sample and that is exactly what is done in the spatial reranking paper by Philbin *et al.* [188].

Specifically, they are solving a particular object retrieval problem: given an image containing some object, return all images from the database, which also contain the same object.

The initial list of images is formed by the descriptor distance and then is reranked. The authors propose to approximate the perspective change between two images as an affine image transformation, and count the number of feature points, which are reprojected inside the second image. This number produces a better ranking than the original short-list.

Wide baseline stereo

While working for spatial re-ranking, 3-degrees of freedom camera model is too rough for the wide baseline stereo. Yet, going from 4 point correspondences (PC) to 2 affine correspondences (AC) for homography and from 7 PC to 3 AC for the fundamental matrix would be substantial benefit anyway for the robust model estimation.

Various variants of RANSAC working with local affine features were proposed in the last 15 years: Perdoch *et al.* [187], Pritts *et al.* [193], Barath and Kukulova [19], Rodríguez *et al.* [210].

Finally, the systematic study of using is presented by Barath *et al.* [22]. Authors show that if used naively, the affine correspondence may lead to worse results, because they are more noisy than point correspondences. However, there is a bag of tricks presented in the paper, which allow to solve the noise issue and make the affine RANSAC working in practice, resulting in orders of magnitude faster computation.

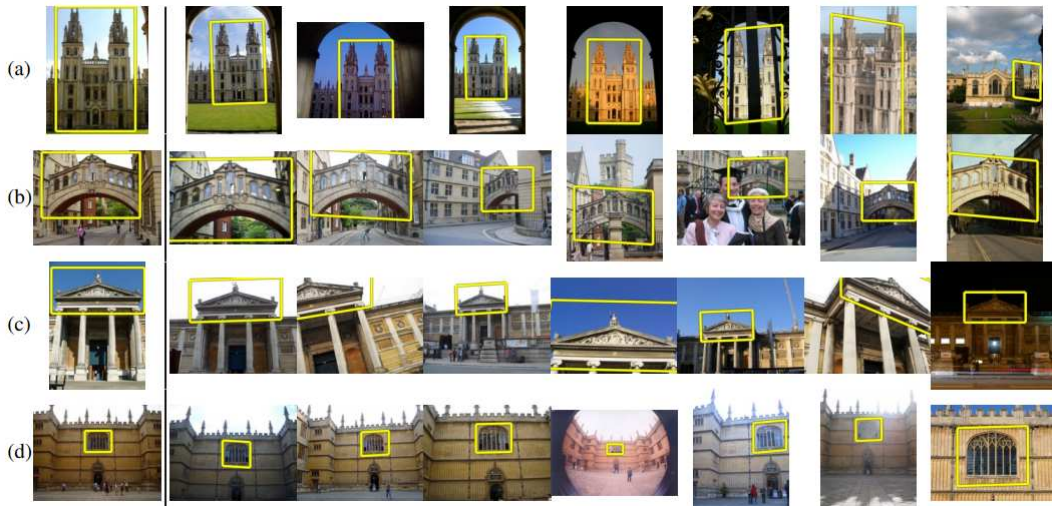


Figure 4.4: From [188]. Affine local features are used for the fast spatial verification.

Moreover, for special cases like autonomous driving, where the motion is mostly horizontal, one could even use 2 affine correspondences for both motion estimation and consistency check, significantly improving the efficiency of the outliers removal compared to the standard RANSAC loop [84].

Besides the special case considerations, additional constraints can also come from running other algorithms, like monocular depth estimation. Such a constraint could reduce the required number of matches from two affine correspondences to a single one for the calibrated camera case [102].

4.3.3 Application-specific benefits

Besides the wide baseline stereo, local affine features and correspondences have other applications. Here are some examples.

Image rectification

Instead of matching local features between two images, one might match them within a single image. Why would someone do it? This allows finding a repeated pattern: think about windows, doors, and so on. Typically they have the same physical size, therefore the difference in local features around them could tell us about the geometry of the scene and lens distortion. This is the idea of the series of works by Prittset *al.* [195, 194, 197, 196].

Surface normals estimation

Ivan Eichhardt and Levente Hajder have a series of works exploiting the local affine correspondences for surface normals estimation [24]

4.3.4 Related work

The area of learning local features has been active recently, but the attention has focused dominantly on learning descriptors [289, 85, 18, 288, 158, 294, 66] and translation-covariant detectors [274, 293, 130, 221]. The authors are not aware of any recent work on learning or improvement of local feature affine shape estimation. The most closely related work is thus the following.

Hartmann *et al.* [92] train random forest classifier for predicting feature matchability based on a local descriptor. "Bad" points are discarded, thus speeding up the matching process in a 3D reconstruction pipeline. Yi *et al.* [285] proposed to learn feature orientation by minimizing descriptor distance between positive patches, i.e. those corresponding to the same point on the 3D surface. This allows to avoid hand-picking a "canonical" orientation, thus learning the one which is the most suitable for descriptor matching. We have observed that direct application of the

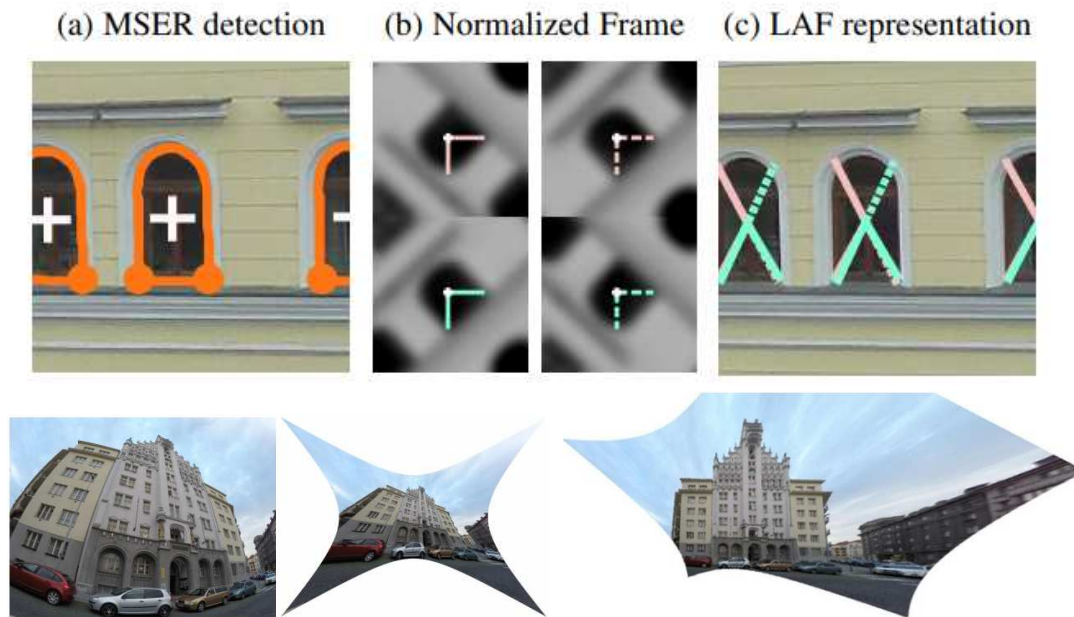


Figure 4.5: Local affine features can be used for image rectification. First, repeated patterns are detected in the image, then their geometry is used for distortion estimation. From [197]

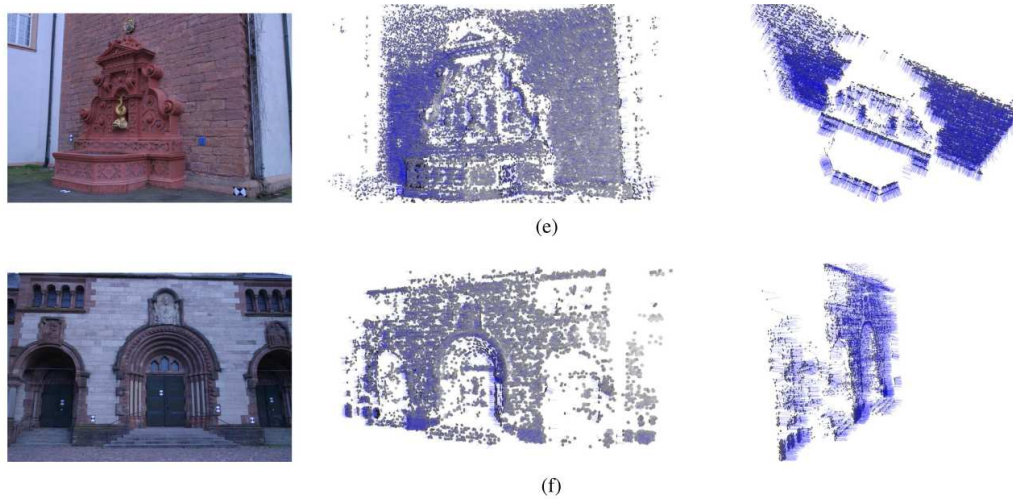


Figure 4.6: From [24]. Local affine features are used for estimation of surface normals (blue).

method [285] for affine shape estimation leads to learning degenerate shapes collapsed to single line. Yi *et al.* [286] proposed a multi-stage framework for learning the descriptor, orientation and translation-covariant detector. The detector was trained by maximizing the intersection-over-union and the reprojection error between corresponding regions.

Lenc and Vedaldi [130] introduced the “covariant constraint” for learning various types of local feature detectors. The proposed covariant loss is the Frobenius norm of the difference between the local affine frames. The disadvantage of such approach is that it could lead to features that are, while being repeatable, not necessarily suited for the matching task (see Section 4.4.2). On top of that, the common drawback of the Yi *et al.* [286] and Lenc and Vedaldi [130] methods is that they require to know the exact geometric relationship between patches which increases the amount of work needed to prepare the training dataset. Zhang *et al.* [293] proposed to “anchor” the detected features to some pre-defined features with known good discriminability like TILDE [274]. We remark that despite showing images of affine-covariant features, the results presented in the paper are for translation-covariant features only. Savinov *et al.* [221] proposed a ranking approach for unsupervised learning of a feature detector. While this is natural and efficient for learning the coordinates of the center of the feature, it is problematic to apply it for the affine shape estimation. The reason is that it requires sampling and scoring of many possible shapes.

Finally, Choy *et al.* [47] trained a “Universal correspondence network” (UCN) for a direct correspondence estimation with contrastive loss on a patch descriptor distance. This approach is related to the current work, yet the two methods differ in several important aspects. First, UCN used an ImageNet-pretrained network which is subsequently fine-tuned. We learn the affine shape estimation from scratch. Second, UCN uses dense feature extraction and negative examples extracted from the same image. While this could be a good setup for short baseline stereo, it does not work well for wide baseline, where affine features are usually sought. Finally, we propose the hard negative-constant loss instead of the contrastive one.

4.4 Learning affine shape and orientation

4.4.1 Affine shape parametrization

A local affine frame is defined by 6 parameters of the affine matrix. Two form a translation vector (x, y) which is given by the keypoint detector and in the rest of the Chapter we omit it and focus on the *affine transformation* matrix A ,

$$A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}. \quad (4.2)$$

Among many possible decompositions of matrix A , we use the following

$$A = \lambda R(\alpha) A' = \det A \begin{pmatrix} \cos \alpha & \sin \alpha \\ -\sin \alpha & \cos \alpha \end{pmatrix} \begin{pmatrix} a'_{11} & 0 \\ a'_{21} & a'_{22} \end{pmatrix}, \quad (4.3)$$

where $\lambda = \det A$ is the scale, $R(\alpha)$ the *orientation* matrix and A' ¹ is the *affine shape* matrix with $\det A' = 1$. A' is decomposed into identity matrix I and *residual shape* A'' :

$$A' = I + A'' = \begin{pmatrix} a'_{11} & 0 \\ a'_{21} & a'_{22} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \begin{pmatrix} a''_{11} & 0 \\ a''_{21} & a''_{22} \end{pmatrix} \quad (4.4)$$

We show that the different parameterizations of the affine transformation significantly influence the performance of CNN-based estimators of local geometry, see Table 4.2.

4.4.2 Hard negative-constant loss

We propose a loss function called hard negative-constant loss (HardNegC). It is based on the hard negative triplet margin loss [158] (HardNeg), but the distance to the hardest (i.e. closest) negative example is treated as constant and the respective derivative of L is set to zero:

$$L = \frac{1}{n} \sum_{i=1, n} \max(0, 1 + d(s_i, \hat{s}_i) - d(s_i, N)), \quad \frac{\partial L}{\partial N} := 0, \quad (4.5)$$

¹ A' has a (0,1) eigenvector, preserving the vertical direction.

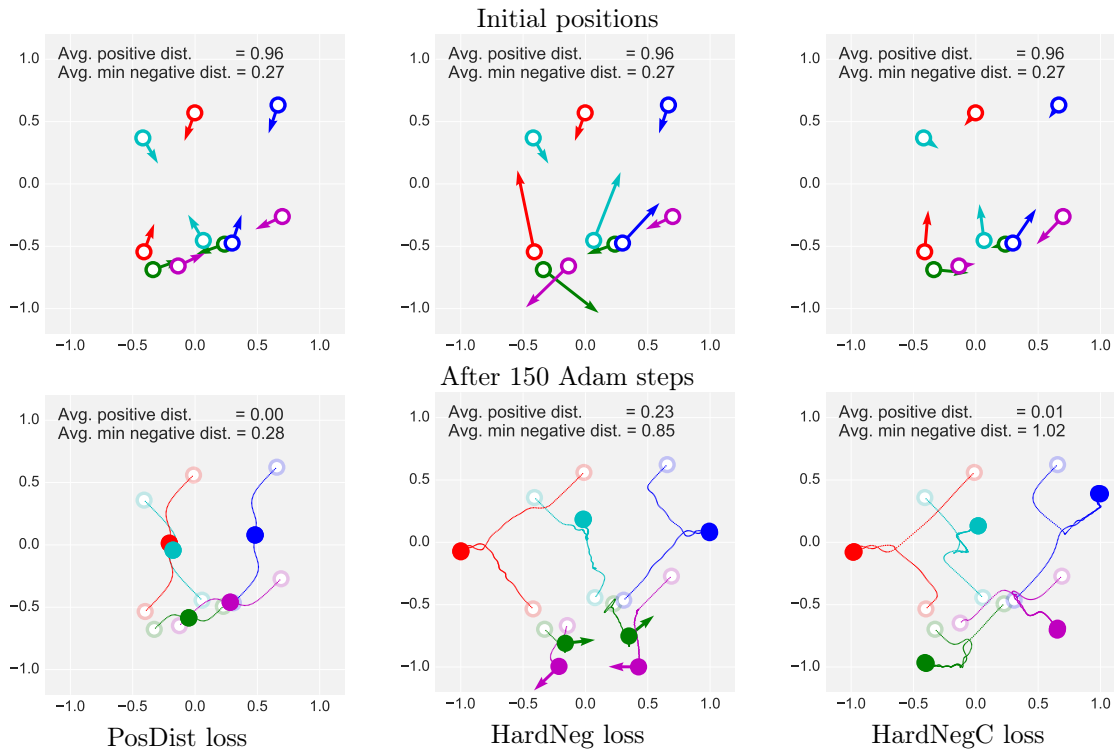


Figure 4.7: A toy example optimization problem illustrating the proposed hard negative-constant (HardNegC) loss. Five pairs of points, representing 2D descriptors, are generated and the losses are minimized by Adam [116]: the positive descriptor distance (PosDist) [285] – left, the hard negative (HardNeg) margin loss [158] – center, HardNegC – right. Top row: identical initial positions of five pairs of matching points. Arrows show the gradient direction and relative magnitude. Bottom row: points after 150 steps of Adam optimization, trajectories are shown by dots. HardNeg loss has a difficulty with the green and magenta point pairs, because the negative example lies between two positives. Minimization of the positive distance only leads to a small distance to the negative examples. The proposed HardNegC loss first pushes same class points close to each other and then distributes them to increase distance to the negative pairs.

where $d(s_i, \hat{s}_i)$ is the distance between the matching descriptors, $d(s_i, N)$ is a distance to the hardest negative example N in the mini-batch for i^{th} pair.

$$d(s_i, N) = \min \left(\min_{j \neq i} d(s_i, \hat{s}_j), \min_{j \neq i} d(s_j, \hat{s}_i) \right)$$

The difference between the Positive descriptor distance loss (PosDist) used for learning local feature orientation in [285] and the HardNegC and HardNeg losses is shown on a toy example in Figure 4.7. Five pairs of points in the 2D space are generated and their positions are updated by the Adam optimizer [116] for the three loss functions. PosDist converges first, but the different class points end up near each other, because the distance to the negative classes is not incorporated in the loss. The HardNeg margin loss has trouble when the points from different classes lie between each other. The HardNegC loss behavior first resembles the PosDist loss, bringing positive points together and then distributes them in the space, satisfying the triplet margin criterion.

4.4.3 Descriptor losses for shape registration

Exploring how local feature repeatability is connected with descriptor similarity, we conducted an shape registration experiment (Figure 4.8). Hessian features are detected in reference HSequences [16] illumination images and reprojected by (identity) homography to another image in the sequence. Thus, the repeatability is 1 and the reprojection error is 0. Then, the local descriptors (HardNet [158], SIFT [140], TFeat [18] and raw pixels) are extracted and the features are matched by first-to-second-nearest neighbor ratio [140] with threshold 0.8. This threshold was suggested by Lowe [140] as a good trade-off between false positives and false negatives. For SIFT,

22% of the geometrically correct correspondences are not the nearest SIFTs and they cannot be matched, regardless of the threshold. In our experiments, the 0.8 threshold worked well for all descriptors and we used it, in line with previous papers, in all experiments.

Notice that for all descriptors, the percentage of correct matches even for the *perfect* geometrical registration is only about 50%.

Adam optimizer is used to update the affine region A to minimize the descriptor-based losses: PosDist, HardNeg and HardNegC. The top two rows show the results for A matrices coupled for both images, bottom – the descriptor difference optimization is allowed to deform A and \hat{A} in both images independently, which leads to a pair of affine regions that are not in perfect geometric correspondence, yet they are more matchable. Note that no training of any kind is involved.

Such descriptor-driven optimization, not maintaining perfect registration, produces a descriptor that is matched successfully up to 90% of the detections under illumination changes.

For most of the unmatched regions, the affine shapes become degenerate lines – shown in the top graphs, and the number of degenerate ellipses is high for PosDist loss; HardNeg and HardNegC perform better.

The bottom row of Figure 4.8 shows the results of experiments where affine shape pairs are independent in each image. Optimization of descriptor losses leads to an increase of the geometric error on the affine shape. Error E is defined as the mean square error on A matrix difference:

$$E = \sum_{i=1}^n \frac{2(A_i - \hat{A}_i)^2}{\det A + \det \hat{A}} \quad (4.6)$$

Again, PosDist loss leads to a larger error. CNN-based descriptors – HardNet and TFeat – lead to a relative small geometric error when reaching the matchability plateau, while for SIFT and raw pixels the shapes diverge. Figure 4.9 shows the case when the initialized shapes include a small amount of reprojection error.

4.4.4 AffNet training procedure

The main blocks of the proposed training procedure are shown in Figure 4.11. First, a batch of matching patch pairs $(P_i, \hat{P}_i)_{i=1..n}$ is generated, where P_i and \hat{P}_i correspond to the same point on a 3D surface. Rotation and skew transformation matrices (T_i, T'_i) are randomly and independently generated. The patches P_i and \hat{P}_i are warped by (T_i, T'_i) respectively into A -transformed patches. Then, a 32×32 center patch is cropped and a pair of transformed patches is fed into the convolutional neural network AffNet, which predicts a pair of affine transformations A_i, \hat{P}_i , that are applied to the T_i -transformed patches via spatial transformers ST [104].

Thus, geometrically normalized patches are cropped to 32×32 pixels and fed into the descriptor network, e.g. HardNet, SIFT or raw patch pixels, obtaining descriptors (s_i, \hat{s}_i) . Descriptors (s_i, \hat{s}_i) are then used to form triplets by the procedure proposed in [158], followed by our newly proposed hard negative-constant loss (Eq. 4.5).

More formally, we are finding affine transformation model parameters θ such that estimated affine transformation A minimizes the descriptor HardNegC loss:

$$A(\theta|(P, \hat{P})) = \arg \min_{\theta} L(s, \hat{s}) \quad (4.7)$$

4.4.5 Training dataset and data preprocessing

UBC Phototour [39] dataset is used for training. It consists of three subsets: *Liberty*, *Notre Dame* and *Yosemite* with about $2 \times 400k$ normalized 64×64 patches in each, detected by DoG and Harris detectors. Patches are verified by the 3D reconstruction model. We randomly sample 10M pairs for training.

Although positive points correspond to roughly the same point on the 3D surface, they are not perfectly aligned, having position, scale, rotation, and affine noise. We have randomly generated affine transformations, which consist in a random rotation – tied for the pair of corresponding patches, and anisotropic scaling t in random direction by magnitude t_m , which is gradually increased during the training from the initial value of 3 to 5.8 at the middle of the training. The tilt is uniformly sampled from range $[0, t_m]$.

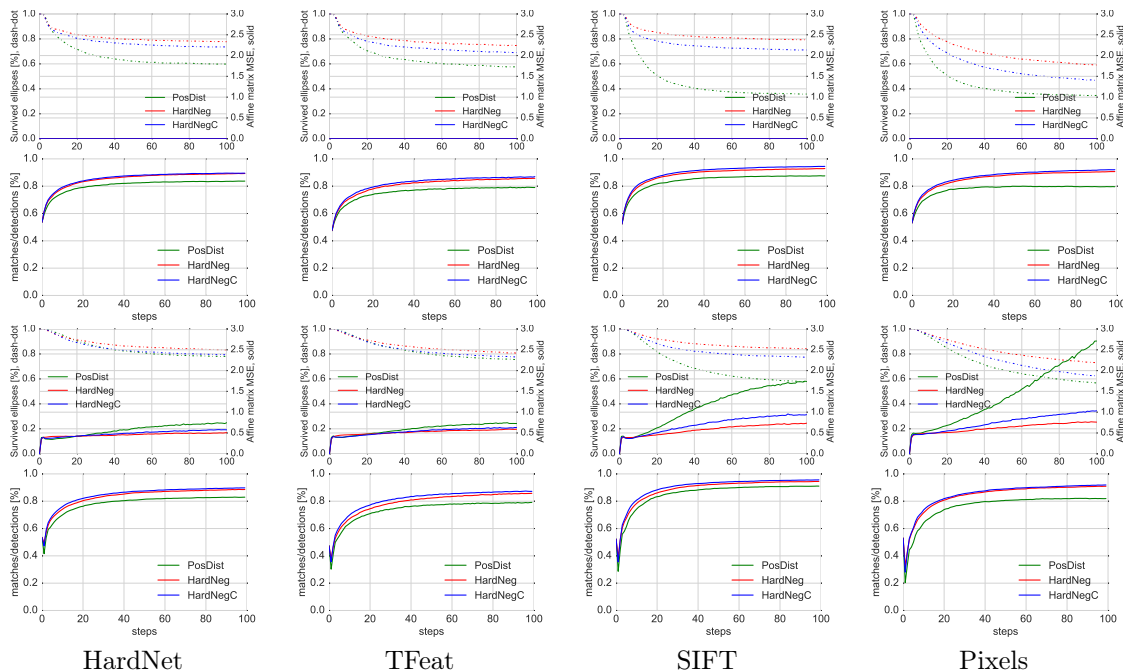


Figure 4.8: Matching score versus geometric repeatability experiment. Affine shape registration by a minimization of descriptor losses of corresponding features. Descriptor losses: green – L2-descriptor distance (PosDist) [285], red – hard triplet margin HardNeg [158], blue – proposed HardNegC. Average over HSequences, illumination subset. *All features are initially perfectly registered.* First two rows: single feature geometry for both images, second two rows: feature geometries are independent in each image. Top row: geometrical error of corresponding features (solid) and percentage of non-collapsed, i.e. elongation ≤ 6 , features (dashed). Bottom row: the percentage of correct matches. This experiment shows that even perfectly initially registered feature might not be matched with any of descriptors – initial matching score is roughly $\approx 30..50\%$. But it is possible to find measurement region, which offers both discriminativity and repeatability. PosDist loss squashes most of the features, leading to the largest geometrical error. HardNeg loss produces the best results in the number of survived feature and geometrical error. HardNegC performs slightly worse than HardNeg, slightly outperforming it on matching score. However, HardNegC is easier to optimize for AffNet learning – see Table 4.1.

4.4.6 Implementation details

The CNN architecture is adopted from HardNet[158], see Fig. 4.10, with the number of channels in all layers reduced 2x and the last 128D output replaced by a 3D output predicting the ellipse shape. The network formula is $16C3-16C3-32C3/2-32C3-64C3/2-64C3-3C8$, where $32C3/2$ stands for 3×3 kernel with 32 filters and stride 2. BatchNorm [99] layer followed by ReLU [177] is added after each convolutional layer, except the last one, which is followed by hyperbolic tangent activation. Dropout [241] with 0.25 rate is applied before the last convolution layer. Grayscale input patches 32×32 pixels are normalized by subtracting the per-patch mean and dividing by the per-patch standard deviation.

Optimization is done by SGD with learning rate 0.005, momentum 0.9, weight decay 0.0001. The learning rate decayed linearly [165] to zero within 20 epochs. The training was done with PyTorch [183] and took 24 hours on Titan X GPU; the bottleneck is the data augmentation procedure. The inference time is 0.1 ms per patch on Titan X, including patch sampling done on CPU and Baumberg iteration – 0.05 ms per patch on CPU.

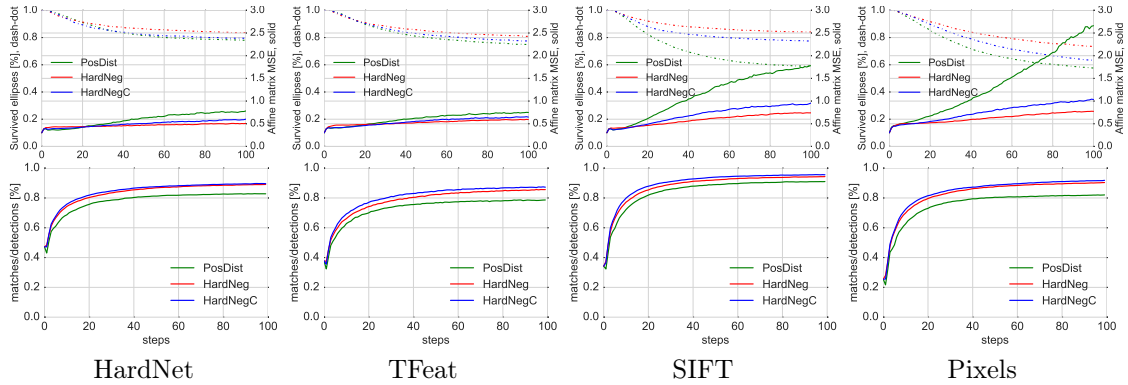


Figure 4.9: Minimization of descriptor loss by optimization of affine parameters of corresponding features. Average over HPatchesSeq, illumination subset. Top row: geometric error of corresponding features (full line) and percentage of non-collapsed, *i.e.* elongation ≤ 6 , features (dashed line). Bottom row: the fraction correct matches. All features initially have the same medium amount of reprojection noise. Left to right: HardNet, SIFT, TFeat, mean-normalized pixels descriptors.

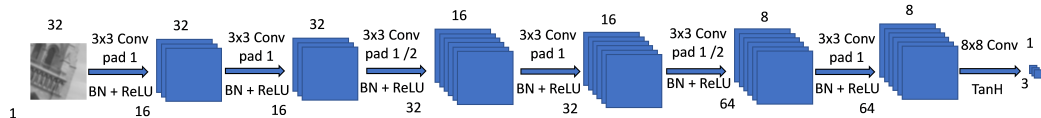


Figure 4.10: AffNet. Feature map spatial size – top, # channels – bottom. /2 stands for stride 2.

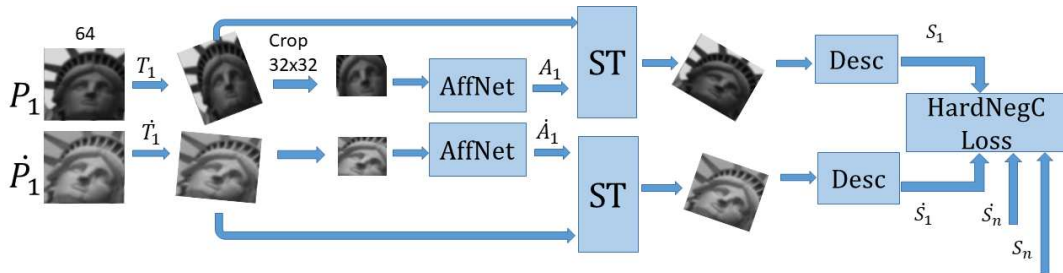


Figure 4.11: AffNet training. Corresponding patches P_i, \hat{P}_i undergo random affine transformation T_i, \hat{T}_i , are cropped and fed into AffNet, which outputs affine transformation A_i, \hat{A}_i to an unknown canonical shape. ST – the spatial transformer warps the patch into an estimated canonical shape. The patch is described by a differentiable CNN descriptor. $n \times n$ descriptor distance matrix is calculated and used to form triplets, according to the HardNegC loss.

4.5 Empirical evaluation

4.5.1 Loss functions and descriptors for learning measurement region

We trained different versions of the AffNet and orientation networks with different combinations, affine transformation parameterizations, and descriptors with the procedure described above. The results of the comparison based on the number of correct matches (reprojection error ≤ 3 pixels) on the hardest pair for each of the 116 sequences from the HSequences [16] dataset are shown in Tables 4.1, 4.2.

The proposed HardNetC loss is the only loss function with no "not converged" results. In the case of convergence, all tested descriptors and loss functions lead to comparable performance, unlike the registration experiments in the previous section. We believe it is because now the CNN always outputs the same affine transformation for a patch, unlike in the previous experiment, where repeated features may end up with different shapes.

Affine transformation parameterizations are compared in Table 4.2. All attempts to learn affine shape and orientation jointly in one network fail completely, or perform significantly worse than

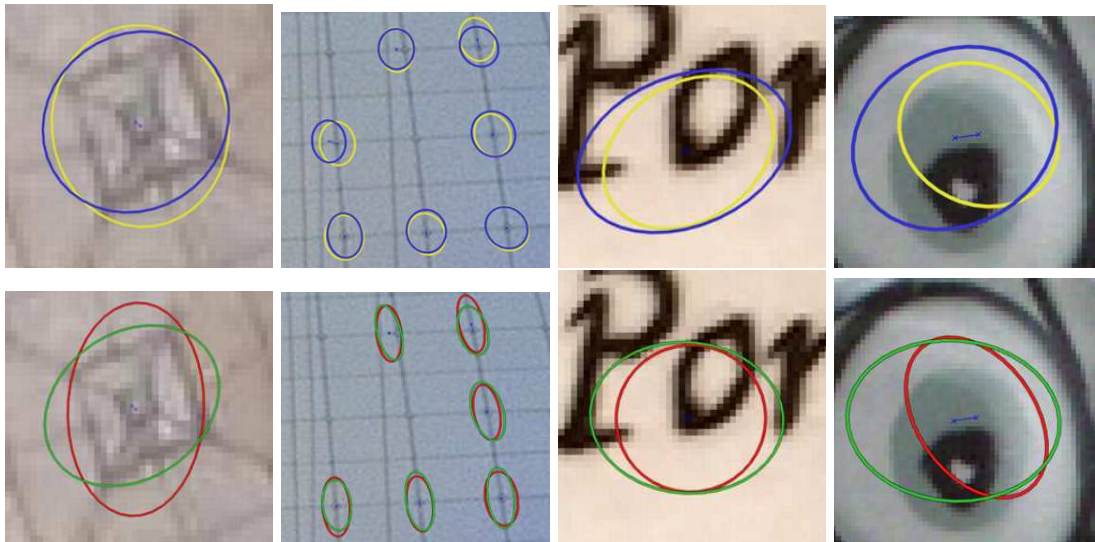


Figure 4.12: AffNet (top) and Baumberg (bottom) estimated affine shape. One ellipse is detected in the reference image, the other is a reprojected closest match from the second image. Baumberg ellipses tend to be more elongated, average axis ratio is 1.99 vs. 1.63 for AffNet, median: Baumberg 1.72 vs 1.39 AffNet. The statistics are calculated over 16M features on Oxford5k.

Table 4.1: Learning the affine transform: loss functions and descriptor comparison. The median of average number of correct matches on the HSequences [16] hardest image pairs 1-6 for the Hessian detector and the HardNet descriptor. The match considered correct for reprojection error ≤ 3 pixels. Affine shape is parametrized as in Eq. 4.4. n/c – did not converge.

| Training descriptor/loss | PosDist | HardNeg | HardNegC |
|----------------------------|------------|---------|------------|
| Affine shape | | | |
| SIFT | n/c | 385 | 386 |
| HardNet | n/c | n/c | 388 |
| Baumberg [25] | | 298 | |
| Orientation | | | |
| SIFT | 387 | 379 | 382 |
| HardNet | 386 | 383 | 380 |
| Dominant orientation [140] | | 339 | |

the two-stage procedure, when the affine shape is learned first and orientation is estimated on an affine-shape-normalized patch. Learning the residual shape A'' (Eq. 4.4) leads to the best results overall. Note that such parameterization does not contain enough parameters to include feature orientation, thus "joint" learning is not possible. Slightly worse performance is obtained by using an identity matrix prior for learnable biases in the output layer.

4.5.2 Repeatability

Repeatability of affine detectors: Hessian detector + affine shape estimator was benchmarked, following the classical work by Mikolajczyk *et al.* [155], but with the recently introduced larger HSequences [16] dataset by VLBenchmarks toolbox [128].

HSequences consists of two subsets. *Illumination* part contains 57 image sixplets with illumination changes, both natural and artificial. There is no difference in viewpoint in this subset, the geometrical relation between images in sixplets is identity. Second part is *Viewpoint*, where 59 image sixplets vary in scale, rotation, but mostly in horizontal tilt. The average viewpoint change is a bit smaller than in well-known *graffiti* sequence from Oxford-Affine dataset [155].

Local features are detected in pairs of images, reprojected by ground truth homography to

Table 4.2: Learning the affine transform: parameterization comparison. The average number of correct matches on the HPatchesSeq [16] hardest image pairs 1-6 for the Hessian detector and the HardNet descriptor. Cases compared, affine shape combined with the de-facto handcrafted standard dominant orientation, affine shape and orientation learnt separately or jointly. The match considered correct for reprojection error ≤ 3 pixels. The HardNegC loss and HardNet descriptor used for learning. n/c – did not converge.

| Eq. | Matrix | Estimated parameters | biases init | Orientation | | |
|-------|-----------------------|---|----------------|-------------|------------|----------------------------|
| | | | | Learned | | Dominant gradient [140] |
| | | | | jointly | separately | |
| (4.2) | A | $(a_{11}, a_{12}, a_{21}, a_{22})$ | 0 | n/c | n/c | n/c |
| (4.2) | A | $(a_{11}, a_{12}, a_{21}, a_{22})$ | 1 | n/c | 360 | 320 |
| (4.3) | A' , $R(\alpha)$ | $(a'_{11}, 0, a'_{21}, a'_{22}),$ $(\sin \alpha, \cos \alpha)$ | 1 | 250 | 327 | 286 |
| (4.4) | A'' | $(a''_{11}, a''_{21}, a''_{22})$ | 1 | - | 370 | 340 |
| (4.4) | A'' | $(1 + a''_{11}, a''_{21}, 1 + a''_{22})$ | 0 | - | 388 | 349 |

the reference image and the closest reprojected region is found for each region in the reference image. The correspondence is considered correct when the overlap error of the pair is less than 40%. The repeatability score for a given pair of images is a ratio between the number of correct correspondences and the smaller number of detected regions in the common part of a scene among two images.

Results are shown in Figure 4.13. The original affine shape estimation procedure, implemented in [185] is denoted Baum SS 19, as 19×19 patches are sampled from the scale space. AffNet takes 32×32 patches, which are sampled from the original image. Therefore, for a fair comparison, we also tested Baum versions, where patches are sampled from the original image, with 19 and 33 pixels patch size.

AffNet slightly outperforms all variants of Baumberg procedure for images with viewpoint changes in terms of repeatability and more significantly – in the number of correspondences. The difference is even bigger for the image with illumination change only, where AffNet performs almost the same as the plain Hessian, which is the upper bound here, as this part of the dataset has no viewpoint changes.

We have also tested AffNet with other detectors on the Viewpoint subset of the HPatches. The repeatabilities are the following (no affine adaptation/Baumberg/AffNet): DoG: 0.46/0.51/0.52, Harris: 0.41/0.44/0.47, Hessian: 0.47/0.52/0.56 The proposed method outperforms the standard (Baumberg) for all detectors.

One reason for such difference is the feature rejection strategy. Baumberg iterative procedure rejects the feature in one of three cases. First, elongated ellipses with long-to-short axis ratio more than six are rejected. Second, features touching boundary of the image are rejected. This is true for the AffNet postprocessing procedure as well, but AffNet produces less elongated shapes: the average axis ratio for Oxford5k 16M features is 1.63 vs. 1.99 for Baumberg. Both cases happen less often for AffNet, increasing the number of surviving features by 25%. We compared the performance of the Baumberg vs. AffNet with the same number of features in Section 4.5.4. Finally, features whose shape did not converge within sixteen iterations are removed. This is quite rare, it happens in approximately 1% of cases. Examples of shapes estimated by AffNet and the Baumberg procedure are shown in Fig. 4.12.

4.5.3 Wide baseline stereo

We conducted an experiment on wide baseline stereo, following the local feature detector benchmark protocol, defined in [163] on a set of two-view matching datasets [93, 282, 300, 74]. The local features are detected by a benchmarked detector, described by HardNet++ [158] and Half-RootSIFT [114] and geometrically verified by RANSAC [126]. Two following metrics are reported: the number of successfully matched image pairs and the average number of correct inliers per matched pair. We have replaced the original affine shape estimator in Hessian-Affine with AffNet in Hessian and Adaptive threshold Hessian (AdHess)

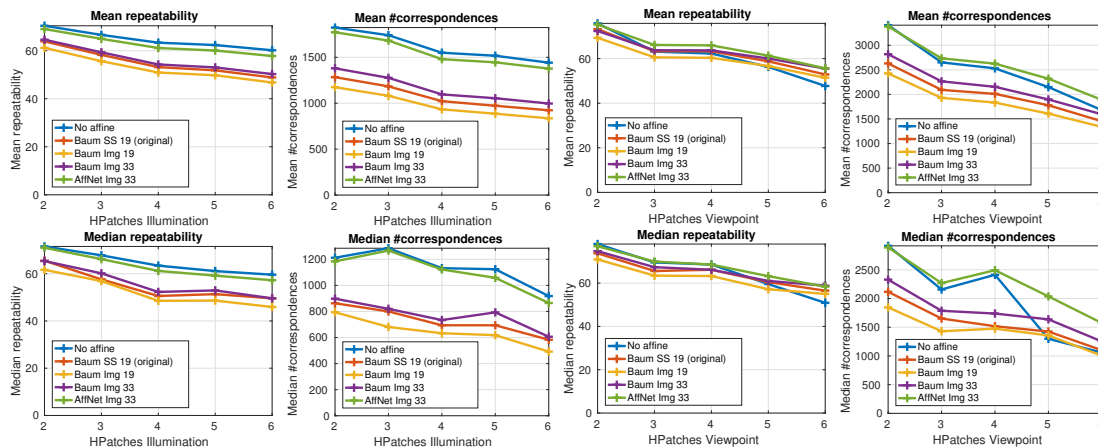


Figure 4.13: Repeatability and the number of correspondences (mean top, median bottom row) on the HSequences [16]. AffNet is compared with the de facto standard Baumberg iteration [25] according to the Mikolajczyk protocol [155]. Left – images with illumination differences, right – with viewpoint and scale changes. SS – patch is sampled from the scale-space pyramid at the level of the detection, image – from the original image; 19 and 33 – patch sizes. Hessian-Affine is from [185]. For illumination subset, performance of Hessian with no adaptation is an upper bound, and AffNet performs close to it.

Table 4.3: AffNet vs. Baumberg affine shape estimators on wide baseline stereo datasets, with Hessian and adaptive Hessian detectors, following the protocol [163]. The number of matched image pairs and the average number of inliers. The numbers of image pairs in a dataset are boxed. Best results are in **bold**.

| | EF [300] | | EVD [162] | | OxAff [155] | | SymB [93] | | GDB [282] | | LTLl [74] | |
|----------------|---|------------|--|-----------|--|-------------|--|------------|--|------------|--|-----------|
| Detector | 33 | inl. | 15 | inl. | 40 | inl. | 46 | inl. | 22 | inl. | 172 | inl. |
| HesAff [154] | 33 | 78 | 2 | 38 | 40 | 1008 | 34 | 153 | 17 | 199 | 26 | 34 |
| HesAffNet | 33 | 112 | 2 | 48 | 40 | 1181 | 37 | 203 | 19 | 222 | 46 | 36 |
| AdHesAff [163] | 33 | 111 | 3 | 33 | 40 | 1330 | 35 | 190 | 19 | 286 | 28 | 35 |
| AdHesAffNet | 33 | 165 | 4 | 42 | 40 | 1567 | 37 | 275 | 21 | 336 | 48 | 39 |

The results are shown in Table 4.3. AffNet outperforms Baumberg in both the number of registered image pairs and/or the number of correct inliers in all datasets, including painting-to-photo pairs in SymB [93] and multimodal pairs in GDB [282], despite it was not trained for that domains.

The total runtimes per image are the following (average for 800x600 images). Baseline HesAff + dominant gradient orientation + SIFT: no CNN components – 0.4 sec. HesAffNet (CNN) + dominant gradient orientation + SIFT – 0.8s, 3 CNN components: HesAffNet + OriNet + HardNet – 1.2 s. Now the data is naively transferred from CPU to GPU and back during each of the stages, which generates the major bottleneck.

4.5.4 Image retrieval

We evaluate the proposed approach on standard image retrieval datasets Oxford5k [188] and Paris6k [189]. Each dataset contains images (5062 for Oxford5k and 6391 for Paris6k) depicting 11 different landmarks and distractors. The performance is reported as mean average precision (mAP) [188]. Recently, these benchmarks have been revisited, annotation errors fixed, and new, more challenging sets of queries added [201]. The revisited datasets define new test protocols: *Easy*, *Medium*, and *Hard*.

We use the multi-scale Hessian-affine detector [155] with the Baumberg method for affine shape estimation. The proposed AffNet replaces Baumberg, which we denote HessAffNet. The use of HessAffNet increased the number of used features from 12.5M to 17.5M for Oxford5k and from

Table 4.4: Performance (mAP) evaluation of bag-of-words (BoW) image retrieval on the Oxford5k and Paris6k benchmarks. Vocabulary consisting of 1M visual words is learned on the independent dataset: Oxford5k vocabulary for Paris6k evaluation and *vice versa*. SV: spatial verification. QE(t): query expansion with t inliers threshold. The best results are in **bold**.

| Detector-Descriptor | Oxford5k | | | | Paris6k | | | |
|-------------------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| | BoW | +SV | +SV+QE(15) | +SV+QE(8) | BoW | +SV | +SV+QE(15) | +SV+QE(8) |
| HesAff-RootSIFT [14] | 55.1 | 63.0 | 78.4 | 80.1 | 59.3 | 63.7 | 76.4 | 77.4 |
| HesAffNet-RootSIFT | 61.6 | 72.8 | 86.5 | 88.0 | 63.5 | 71.2 | 81.7 | 83.5 |
| HesAff-TFeat-M* [18] | 46.7 | 55.6 | 72.2 | 73.8 | 43.8 | 51.8 | 65.3 | 69.7 |
| HesAffNet-TFeat-M* | 45.5 | 57.3 | 75.2 | 77.5 | 50.6 | 58.1 | 72.0 | 74.8 |
| HesAff-HardNet++ [158] | 60.8 | 69.6 | 84.5 | 85.1 | 65.0 | 70.3 | 79.1 | 79.9 |
| HesAffNetLess-HardNet++ | 64.3 | 73.3 | 86.1 | 87.3 | 62.0 | 68.7 | 79.1 | 79.2 |
| HesAffNet-HardNet++ | 68.3 | 77.8 | 89.0 | 91.1 | 65.7 | 73.4 | 83.3 | 83.3 |

Table 4.5: Performance (mAP) comparison with the state-of-the-art in local feature-based image retrieval. Vocabulary is learned on the independent dataset: Oxford5k vocabulary for Paris6k evaluation and *vice versa*. All results are with spatial verification and query expansion. VS: vocabulary size. SA: single assignment. MA: multiple assignments. The best results are in **bold**.

| Method | VS | Oxford5k | | Paris6k | |
|------------------------------|-----|-------------|-------------|-------------|-------------|
| | | SA | MA | SA | MA |
| HesAff-SIFT-BoW-fVocab [156] | 16M | 74.0 | 84.9 | 73.6 | 82.4 |
| HesAff-RootSIFT-HQE [261] | 65k | 85.3 | 88.0 | 81.3 | 82.8 |
| HesAff-HardNet++-HQE [158] | 65k | 86.8 | 88.3 | 82.8 | 84.9 |
| HesAffNet-HardNet++-HQE | 65k | 87.9 | 89.5 | 84.2 | 85.9 |

15.6M to 21.2M for Paris6k, because more features survive the affine shape adaptations, as explained in Section 4.5.2. We also performed an additional experiment by restricting the number of AffNet features to the same as in Baumberg – HesAffNetLess in Table 4.4. We evaluated HesAffNet with both hand-crafted descriptor RootSIFT [14] and state-of-the-art learned descriptors [18, 158].

First, HesAffNet is tested within the traditional bag-of-words (BoW) [238] image retrieval pipeline. A flat vocabulary with 1M centroids is created with the k-means algorithm and approximate nearest neighbor search [171]. All descriptors of an image are assigned to a respective centroid of the vocabulary, and then they are aggregated with a histogram of occurrences into a BoW image representation.

We also apply spatial verification (SV) [188] and the standard query expansion (QE) [189]. QE is performed on images that have either 15 (typically used) or 8 inliers after the spatial verification. The results of the comparison are presented in Table 4.4.

AffNet achieves the best results on both Oxford5k and Paris6k datasets, in most of the cases it outperforms the second best approach by a large margin. This experiment clearly shows the benefit of using AffNet in the local feature detection pipeline.

Additionally, we compare with state-of-the-art local-feature-based image retrieval methods. A visual vocabulary of 65k words is learned, with Hamming embedding (HE) [107] technique added that further refines the descriptor assignments with a 128 bits binary signature. We follow the same procedure as HesAff-RootSIFT-HQE [261] method. All parameters are set as in [261]. The performance of AffNet methods is the best, reported on both Oxford5k and Paris6k for local features.

Finally, on the revisited R-Oxford and R-Paris, we compare with state-of-the-art methods in image retrieval, both local and global feature based: the best-performing fine-tuned networks [95], ResNet101 with generalized mean pooling (ResNet101-GeM) [204] and ResNet101 with regional maximum activations pooling (ResNet101-R-MAC) [83]. Deep methods use re-ranking methods: α query expansion (α QE) [204], and global diffusion (DFS) [100]. Results are in Table 4.6.

HesAffNet performs best on the R-Oxford. It is consistently the best performing local feature method, yet is worse than deep methods on R-Paris. A possible explanation is that deep networks

Table 4.6: Performance (mAP, mP@10) comparison with the state-of-the-art in image retrieval on the R-Oxford and R-Paris benchmarks [201]. SV: spatial verification. HQE: hamming query expansion. α QE: α query expansion. DFS: global diffusion. The best results are in **bold**.

| Method | Medium | | | | Hard | | | |
|----------------------------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| | R-Oxford | | R-Paris | | R-Oxford | | R-Paris | |
| | mAP | mP@10 | mAP | mP@10 | mAP | mP@10 | mAP | mP@10 |
| ResNet101-GeM+ α QE [204] | 67.2 | 86.0 | 80.7 | 98.9 | 40.7 | 54.9 | 61.8 | 90.6 |
| ResNet101-GeM[204]+DFS [100] | 69.8 | 84.0 | 88.9 | 96.9 | 40.5 | 54.4 | 78.5 | 94.6 |
| ResNet101-R-MAC[83]+DFS [100] | 69.0 | 82.3 | 89.5 | 96.7 | 44.7 | 60.5 | 80.0 | 94.1 |
| ResNet50-DELF[179]-HQE+SV | 73.4 | 88.2 | 84.0 | 98.3 | 50.3 | 67.2 | 69.3 | 93.7 |
| HesAff-RootSIFT-HQE [261] | 66.3 | 85.6 | 68.9 | 97.3 | 41.3 | 60.0 | 44.7 | 79.9 |
| HesAff-RootSIFT-HQE+SV [261] | 71.3 | 88.1 | 70.2 | 98.6 | 49.7 | 69.6 | 45.1 | 83.9 |
| HesAffNet-HardNet++-HQE | 71.7 | 89.4 | 72.6 | 98.1 | 47.5 | 66.3 | 48.9 | 85.9 |
| HesAffNet-HardNet++-HQE+SV | 75.2 | 90.9 | 73.1 | 98.1 | 53.3 | 72.6 | 48.9 | 89.1 |

(ResNet and DELF) were finetuned from ImageNet, which contains Paris-related images, e.g., Sacre Coeur and Notre Dame Basilica in the “church” category. Therefore global deep nets are partially evaluated on the training set.

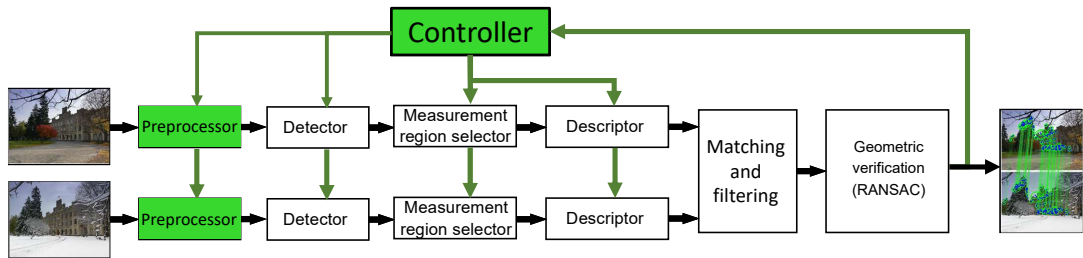
4.6 Summary

We presented a method for learning the affine shape of local features in a weakly supervised manner. The proposed HardNegC loss function might find other application domains as well. Our intuition is that the distance to the hard-negative estimates the local density of all points and provides a scale for the positive distance. The resulting AffNet regressor bridges the gap between the performance of the similarity-covariant and affine-covariant detectors on images with short baseline and big illumination differences and it improves the performance of affine-covariant detectors in the wide baseline setup. AffNet applied to the output of the Hessian detector improves the state-of-the-art in wide baseline matching, affine detector repeatability and image retrieval.

We experimentally show that descriptor matchability, not only repeatability, should be taken into account when learning a feature detector.

CHAPTER 5

MODS: Matching with On-Demand View Synthesis



We have discussed in the previous Chapters how difficult can be the WxBS problem. In this Chapter we study a problem of matching image pairs, taken from extreme viewpoints. We introduce a benchmark and the Extreme View Dataset (EVD) and show that the standard WxBS pipeline performs poorly on this dataset. We present a solution – by transforming a standard feedforward pipeline into a controlled system by adding a feedback loop and on-demand view synthesis. The resulting algorithm called MODS – matching with on-demand view synthesis – is able to match not only images acquired under oblique camera angles, but improves on other WxBS problems as well, like matching historical photographs.

5.1 Introduction

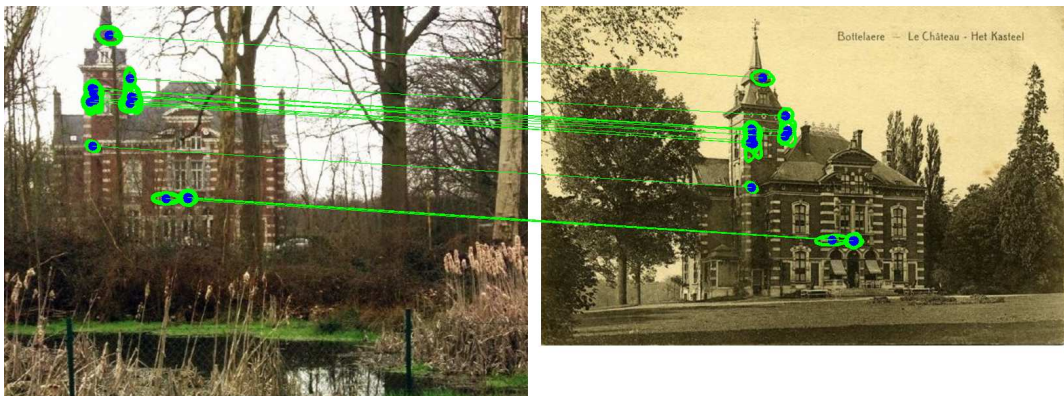


Figure 5.1: A pair of historical photographs, that is not matchable by standard methods. Matched only with help of affine view synthesis in the MODS[162] framework. Recent historical virtual reality application uses MODS features [175].

Standard wide-baseline stereo or 3D reconstruction pipelines work well in many situations. Even if an image pair is not matched, it is usually not a problem. For example, one could still

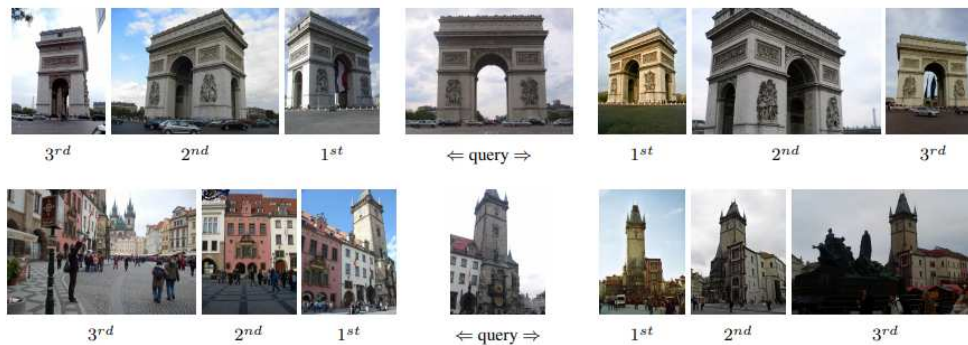


Figure 5.2: Frontal and side views of the Arc de Triomphe (top) and Astronomical Clock (bottom). One cannot match leftmost and rightmost views, but can register together those images if there are enough proxy images in between them. Figure from [229].

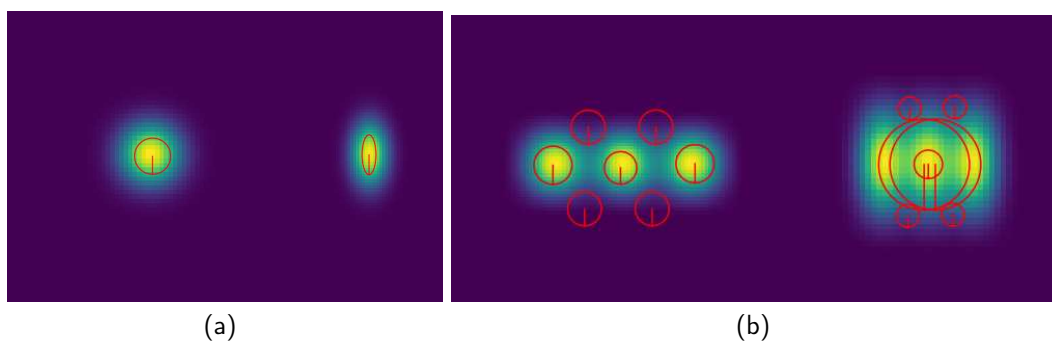


Figure 5.3: Hessian-Affine detections of the synthetic blobs. (a) Detection of the single isolated blob, face-on (left) and from the oblique angle (right). The same support region is detected. (b) Detection of the several blobs near to each other, face-on (left) and from the oblique angle (right). The detections are merging, when the blobs are pictured from a side viewpoint. This results in a failure of the blob detector invariance.

register images from very different viewpoints, if there is a sequence of images in between, as shown in Figure 5.2, from "From Single Image Query to Detailed 3D Reconstruction" paper [229]. However, that might not always be possible. For example, the number of pictures is limited because they are historical [147, 175] and there is no way how one could go and take more without inventing a time machine. What to do? One way would be to use affine features like Hessian-AffNet[161] or MSER[148]. However, they help only up to some extent and what if the views we need to match are more extreme? See an example in Figure 5.1. Let us consider an image which consists of three blobs, which are situated close to each other. Under the tilt transform they merge into a single blob, as shown in Figure 5.3. It is not the shape of the region, which is detected incorrectly, but the centers of the features themselves. One way to address the problem is to simulate the real camera viewpoint change by affine or perspective warps of the current image. This idea was first proposed by Lepetit and Fua in 2006[131]. They synthesized views to find distinctive keypoints repeatedly detectable under affine deformations. Synthetic views provided a training set for learning a random forest classifier that labeled individual feature points. Feature points in different images with the same label were assumed to be in correspondence. The simple keypoint detector of Lepetit and Fua is very fast, but invariant only to translation and rotation and thus the number of views necessary to achieve acceptable repeatability was high. The method was tested on pairs undergoing significant affine transformations, but the final representation did not scale and can not be easily used for indexing.

Later, Morel *et al.* [170] proposed a new matching pipeline – see Alg. 1. The authors showed that view synthesis extends the handled range of viewpoint differences. The ASIFT algorithm starts by generating synthetic views (described in Section 5.2.1) for both images. Next, feature detection and description are performed using standard SIFT [140] in each synthesized view. Tentative correspondences are formed for all pairs of views synthesized from the first and second

Algorithm 1 ASIFT**Input:** I_1, I_2 – two images.**Output:** List of corresponding points; fundamental matrix F .

```

for  $I_1$  and  $I_2$  independently do
  1.1 Generate synthetic views according to the tilt-rotation-detector setup.
  1.2 Detect and describe local features.
end for
2 Generate tentative correspondences for each pair of the views synthesized from  $I_1$  and  $I_2$ 
 $I_1$  and  $I_2$  independently using the 2nd closest ratio.
3 Add correspondences to the global list.
  Reproject the corresponding features to the original images.
4 Filter duplicate, “one-to-many” and “many-to-one” matches.
5 Geometrically verify tentative correspondences using ORSA [167]
  while estimating  $F$ .

```

image. The matching stage thus entails n^2 independent matching problems, where n is the number of synthesized views per image. The set of correspondences between the images is the union of results for all synthesized pairs. The *duplicate filtering* stage of ASIFT prunes correspondences with small spatial distance (2 pixels) of local features in both images – all such correspondences except one (random) are eliminated from the final correspondence set. “*One-to-many*” correspondences – correspondences of features which are close to each other (are situated in the radius of $\sqrt{2}$ pixels) in one image while spread in other synthetic views are also eliminated, despite the fact that some of the pairs can be correct. Finally, geometric verification is performed by ORSA [167]. ORSA is a RANSAC-based method, which exploits an a-contrario approach to detect incorrect epipolar geometries. Instead of having a constant error threshold, ORSA looks for matches that have the highest “diameter”, i.e., matches which cover a large image area. ASIFT was shown to match images of a scene with a viewpoint difference up to 80° for planar objects [170]. Computational costs are in the order of tens of seconds to a few minutes.

The latest extensions of wide-baseline matching pipeline are limited to modifications of the ASIFT algorithm. Liu *et al.* [139] synthesized perspective warps rather than affine. Pang *et al.* [182] replaced SIFT by SURF [26] in the ASIFT algorithm to reduce the computation time.

5.2 The MODS algorithm

The main idea of the proposed iterative MODS algorithm (see Alg. 2) is to repeat a sequence of two-view matching procedures, until a required number of geometrically verified correspondences is found. In each iteration, a different and potentially complementary detector is used and a different set of views is synthesized. The algorithm starts with fast detectors with limited invariance, proceeding progressively to more complex, robust, but computationally costly ones. MODS is thus capable of solving simple matching problems fast without losing the ability to deal with very difficult cases where a combination of detectors is employed to extend the state-of-the-art.

The adopted sequence of detectors and view synthesis parameters is an outcome of an extensive experimental search. The objective was to solve the most challenging problems in the development set, i.e., to correctly recover their two-view geometry, while keeping the speed comparable to standard single-detector wide-baseline matchers for simple problems. Details about the selected configuration and the optimization process are given in Section 5.3. The rest of the section describes the steps involved in the iterations of the MODS algorithm, which is compared to the standard two-view matching and ASIFT pipelines.

5.2.1 Synthetic views generation

MODS (Alg. 2) starts by synthetic view generation. It is well known that a homography H can be approximated by an affine transformation A at a point using the first order Taylor expansion. The affine transformation can be uniquely decomposed by SVD into a rotation, skew, scale and rotation around the optical axis [90]. In [170], the authors proposed to decompose the affine

Algorithm 2 MODS**Input:** I_1, I_2 – two images; θ_m – minimum required number of matches; S_{\max} – maximum number of iterations.**Output:** Fundamental or homography matrix F or H; list of corresponding points.**Variables:** N_{matches} – detected correspondences, Iter – current iteration.

```

while ( $N_{\text{matches}} < \theta_m$ ) and (Iter <  $S_{\max}$ ) do
  for  $I_1$  and  $I_2$  independently do
    1.1 Generate synthetic views according to the
        scale-tilt-rotation-detector-descriptor setup for the Iter (Tables 5.1, 5.4).
    1.2 Detect and describe local features.
    1.3 Reproject local features to the original image.
        Add the described features to the global list.
    end for
    2 Generate tentative correspondences using the first geom. inconsistent rule.
    3 Filter duplicate matches.
    4 Geometrically verify tentative correspondences with DEGENSAC [51]
        while estimating F or H.
  end while

```

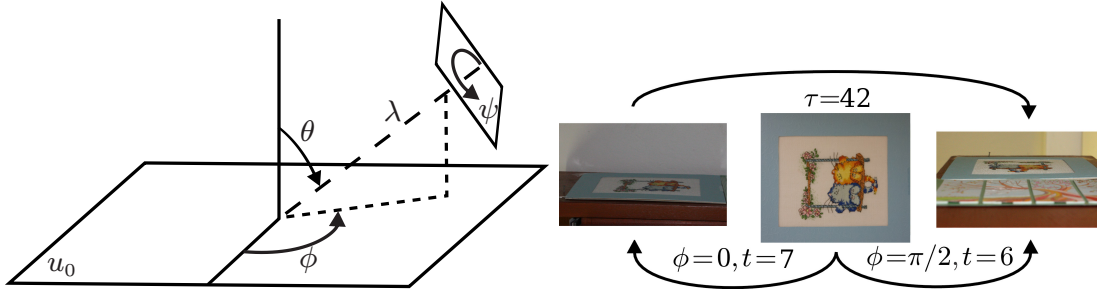


Figure 5.4: (left) the affine camera model (5.1). Latitude $\theta = \arccos 1/t$ – latitude, longitude ϕ , scale λ scale. (right) Transitional tilt τ for absolute tilt t and rotation ϕ .

transformation A as

$$\begin{aligned}
 A &= H_\lambda R_1(\psi) T_t R_2(\phi) = \\
 &= \lambda \begin{pmatrix} \cos \psi & -\sin \psi \\ \sin \psi & \cos \psi \end{pmatrix} \begin{pmatrix} t & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{pmatrix} \quad (5.1)
 \end{aligned}$$

where $\lambda > 0$, R_1 and R_2 are rotations, and T_t is a diagonal matrix with $t > 1$. Parameter t is called the absolute tilt, $\phi \in \langle 0, \pi \rangle$ is the optical axis longitude and $\psi \in \langle 0, 2\pi \rangle$ is the rotation of the camera around the optical axis (see Figure 5.4). Each synthesized view is thus parametrized by the tilt, longitude and optionally the scale and represents a sample of the view-sphere resp. view-volume around the original image.

The view synthesis proceeds in the following steps: at first, a scale synthesis is performed by building a Gaussian scale-space with Gaussian $\sigma = \sigma_{\text{base}} \cdot S$ and downsampling factor S ($S < 1$). Then, each image in the scale-space is in-plane rotated by longitude ϕ with step $\Delta\phi = \Delta\phi_{\text{base}}/t$. In the third step, all rotated images are convolved with a Gaussian filter with $\sigma = \sigma_{\text{base}}$ along the vertical and $\sigma = t \cdot \sigma_{\text{base}}$ along the horizontal direction to eliminate aliasing in a final tilting step. The final tilt is applied by shrinking the image along the horizontal direction by factor t . The synthesis parameters are: the set of scales $\{S\}$, $\Delta\phi_{\text{base}}$ – the longitude sampling step at tilt $t = 1$, the set of simulated tilts $\{t\}$.

5.2.2 Local feature detection and description

The second step of MODS is the detection and description of local features. It is known that different local feature detectors are suitable for different types of images [152] and that some detectors are complementary in the image structures they respond to [1]. Our experiments show that combining detectors improves the overall robustness and speed of the matching procedure.

Table 5.1: MODS step configurations are defined by a detector, descriptor and the set of the synthesized views. RootSIFT is used for all detectors but ORB which is described by BRIEF.

| Iter. | Setup |
|-------|--|
| 1 | ORB, $\{S\} = \{1\}$, $\{t\} = \{1\}$, $\Delta\phi = 360^\circ/t$ |
| 2 | ORB, $\{S\} = \{1\}$, $\{t\} = \{1; 5; 9\}$, $\Delta\phi = 360^\circ/t$ |
| 3 | MSER, $\{S\} = \{1; 0.25; 0.125\}$, $\{t\} = \{1\}$, $\Delta\phi = 360^\circ/t$ |
| 4 | MSER, $\{S\} = \{1; 0.25; 0.125\}$, $\{t\} = \{1; 3; 6; 9\}$, $\Delta\phi = 360^\circ/t$ |
| 5 | HessAff, $\{S\} = \{1\}$, $\{t\} = \{1; 2; 4; 6; 8\}$, $\Delta\phi = 360^\circ/t$ |
| 6 | HessAff, $\{S\} = \{1\}$, $\{t\} = \{1; 2; 4; 6; 8\}$, $\Delta\phi = 120^\circ/t$ |
| 7 | HessAff, $\{S\} = \{1\}$, $\{t\} = \{1; 2; 4; 6; 8; 10\}$, $\Delta\phi = 60^\circ/t$ |

Table 5.2: MODS variants tested. The corresponding steps are the same as in Table 5.1. The starred MODS:3*-6* configuration slightly differs from Table 5.1 and corresponds to [164].

| Name | Steps | | | | | | |
|------------------|-------|---|----|----|---|---|---|
| MODS == MODS:1-7 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| MODS:2-7 | - | 2 | 3 | 4 | 5 | 6 | 7 |
| MODS:3*-6 | - | - | 3* | 4* | 5 | 6 | - |
| MODS:3-7 | - | - | 3 | 4 | 5 | 6 | 7 |
| MODS:3,2-7 | 3 | - | 2 | 4 | 5 | 6 | 7 |
| MODS:2,4,7 | - | 2 | - | 4 | - | - | 7 |
| MODS:5,1-7 | 5 | 1 | 2 | 3 | 4 | 6 | 7 |

MODS combines a fast similarity covariant FAST (in ORB implementation) detector and affine covariant detectors MSER and Hessian-Affine. The normalized patches are described by the binary descriptor BRIEF [212] (in ORB implementation) and a recent modification of SIFT [140] – the RootSIFT [14]. The local feature frames computed on the synthesized views are backprojected to the coordinate system of the original image by the known affine matrix A and associated with the descriptor and the originating synthetic view. MODS step configurations are specified in Table 5.1.

For the MSER and Hessian-Affine detectors, the fast affine feature extraction process from [129] was applied.

5.2.3 Tentative correspondence generation

The next step of the MODS algorithm is the generation of tentative correspondences. Different strategies for the computation of tentative correspondences in wide-baseline matching were proposed. The standard method for matching SIFT(-like) descriptors is based on the ratio of the distances to the closest and the second closest descriptors in the other image [140]. While the performance of this test is in general very good, it degrades when multiple observations of the same feature are present. In this case, the presence of similar descriptors will lead to the first to second SIFT ratio to be close to 1 and the correspondences will "annihilate" each other, despite the fact that they represent the same geometric constraints and are therefore not mutually contradictory (see Figure 5.5). The problem of multiple detections is amplified in matching by view synthesis since covariantly detected local features are often repeatedly discovered in multiple synthetic views.

To address this problem, we propose a modified matching strategy denoted *first to first geometrically inconsistent – FGINN*. Instead of comparing the first to the second closest descriptor distance, the distance of the first descriptor and the closest descriptor that is geometrically inconsistent with the first one is used. We call descriptors in one image geometrically inconsistent if the Euclidean distance between the centers of the regions is $\geq n$ pixels (default: $n = 10$). The difference of the first-to-second closest ratio strategy and the FGINN strategy is illustrated in Figure 5.5.



Figure 5.5: Green regions – correct correspondences rejected by the standard first to second closest ratio test (second closest region is in red), but recovered by the first to first geometrically inconsistent ratio (first geometrically inconsistent region is in yellow) matching strategy. DoG regions (top), MSERs (bottom).

5.2.4 Geometric verification

The last step of the MODS is the geometric verification. It consists of three substeps.

Duplicate filtering

The redetection of covariant features in synthetic views results in duplicates in tentative correspondences. The *duplicate filtering* prunes correspondences with close spatial distance (≈ 10 pixels) of local features in both images – all these correspondences except one – with the smallest descriptor distance ratio – are eliminated from the final correspondences list. The number of pruned correspondences can be used later for evaluating the quality (probability of being correct) in PROSAC-like [49] geometric verification.

RANSAC

The LO-RANSAC [50] algorithm searches for the maximal set of geometrically consistent tentative correspondences. The model of the transformation is set either to homography or epipolar geometry, or automatically determined by a DegenSAC [51] procedure.

Local affine frame check

Since the epipolar geometry constraint is much less restrictive than a homography, wrong correspondences consistent with some (random) fundamental matrix appear. The local affine frame consistency check (LAF-check) eliminates virtually all incorrect correspondences. The procedure uses coordinates of the closest and furthest ellipse points from the ellipse center of both matched local affine frames to check whether the whole local feature is consistent with the estimated geometry model (see Figure 5.6). The check is performed with the geometric model obtained by RANSAC. Regions which do not pass the check are discarded from the list of inliers. If the

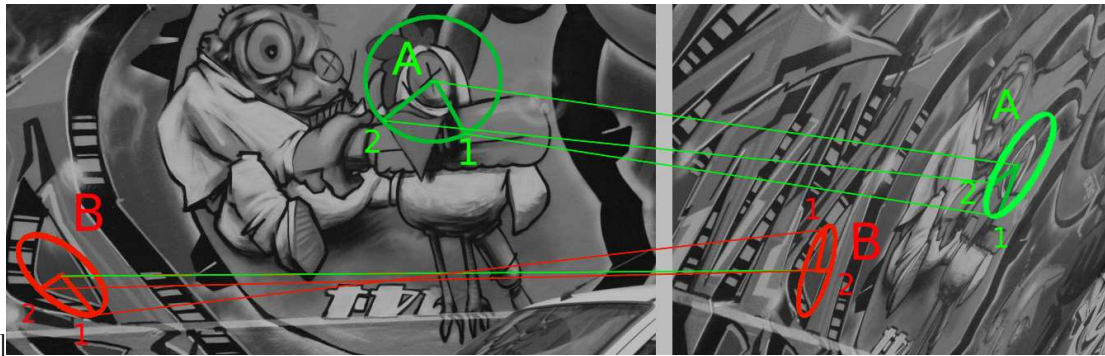


Figure 5.6: The LAF-check. While centers of both regions A and B are consistent with found homography, farthest (1) and closest (2) points of the ellipse pass the check only for region A.



Figure 5.7: Two examples of image sets from the synthetic dataset. Original unwarped images are from the Oxford buildings dataset [188].

number of correspondences after the LAF-check is fewer than the user defined minimum, the matcher continues with the next step of view synthesis.

5.3 Implementation and parameter setup

In this subsection, we discuss the tilt-rotation-detector setups of the MODS algorithm, and threshold selection for the *first to first geometrically inconsistent* – *FGINN* matching strategy validation.

5.3.1 View synthesis for different detectors and descriptors

The two main parameters of the view synthesis, tilt $\{t\}$ sampling and the latitude step $\Delta\phi_{\text{base}}$, were explored in the following synthetic experiment.

A set of simulated views with latitude angles $\theta = (0, 20, 40, 60, 65, 70, 75, 80, 85^\circ)$, corresponding to tilt series $t = (1.00, 1.06, 1.30, 2.00, 2.36, 2.92, 3.86, 5.75, 11.47)$ ¹ was generated for each of 150 random images from the Oxford Building Dataset² [188]. Example images are shown in Figure 5.7. The ground truth affine matrix A was computed for each simulated view using equation (5.1) and used in the final verification step. The original image was matched against its warped version, and the running time and number of inliers for each combination of the detector, tilt and rotation (see Table 5.3) were computed. In all, 84 setups for each of the 8 detectors on the 150 image pairs were evaluated. As an example, we show the relation between the density of the view-sphere sampling and the number of images matched with the DoG detector in Figure 5.8.

Since our goal is to find a variety of detector-tilt-rotation configurations operating with different matching ability – run-time trade-offs, we defined “easy”, “medium” and “hard” problems on the synthetic dataset. Successful two-view matching was defined as recovering $n \geq 50$ ground truth correspondences on a synthetically warped image. The threshold is set high – synthetic warping of an image is underestimating the reduction of the number of matchable features induced by the

¹assuming that the original image is the fronto-parallel view

²available at <http://www.robots.ox.ac.uk/~vgg/data/oxbuildings/>

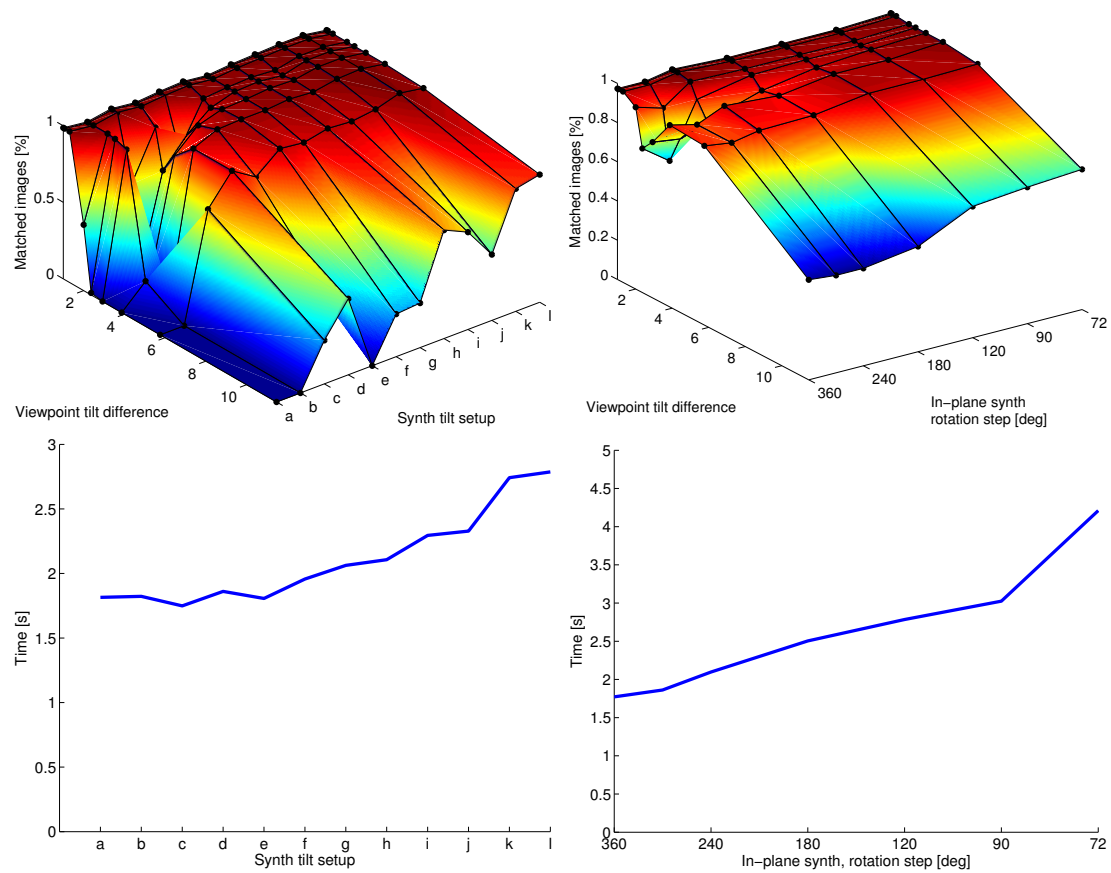


Figure 5.8: Top: the percentage of images matched depending the synthetic viewpoint difference for the DoG detector with tilt configurations given in Table 5.3. Left: different tilt synthesis configurations (for $\Delta\phi = 240^\circ$), right: different rotation synthesis configurations (for tilt set (d) = {1,5,9}). Bottom: running time per image pair for the respective configuration.

Table 5.3: View synthesis configuration evaluation. Configuration is a triplet - the detector, descriptor and the set of the synthesized views.

| | |
|---------------------------------|---|
| Detector-descriptor combination | DoG-SIFT, Hessian-Affine-SIFT, Harris-Affine-SIFT, MSER-SIFT, SURF-SURF, SURF-FREAK, AGAST-FREAK, ORB |
| Tilt set | $a = \{1\}$, $b = \{1,2\}$, $c = \{1,8\}$, $d = \{1,5,9\}$, $e = \{1,4\}$, $f = \{1,4,8\}$, $g = \{1,2,4,8\}$, $h = \{1,3,6,9\}$, $i = \{1,4,8,10\}$, $j = \{1,2,4,6,8\}$, $k = \{1, \sqrt{2}, 2, 2\sqrt{2}, 4, 4\sqrt{2}, 8\}$, $l = \{1,2,3,4,5,6,7,8,9\}$. |
| $\phi_{base} [^\circ]$ | 360, 240, 180, 120, 90, 72, 60 |

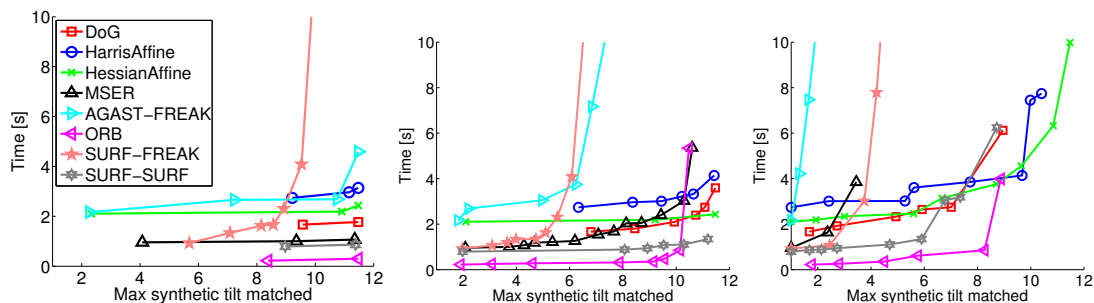


Figure 5.9: Performance of view synthesis configurations on the synthetic dataset. Average time needed to match (left) “easy”, (center) “medium” and (right) “hard” problems. The time for the fastest detector configuration that solved corresponding problems for a given tilt is shown.

effects of a corresponding viewpoint change, e.g., due to non-planarity of the scene or illumination changes. The matcher is considered to solve an “easy” problem if the percentage of the matched images $f \geq 50\%$ of total images, “medium” if $f \geq 90\%$ of images matched and solved “hard” if $f \geq 99\%$ of the images are matched.

The experiment with the synthetically warped dataset gives a hint about the limits of configurations. Three configurations that solved the maximum tilt difference for each case fastest for a given detector were selected for evaluation. The configurations are specified in Table 5.4.

The average time necessary to match a given synthetic tilt difference for different detectors with the optimal configuration is shown in Figure 5.9. The computations were performed on the Intel i7 3.9GHz (8 cores) desktop with 8Gb RAM with parallel processing.

Note that view synthesis significantly increases the matching performance of all detectors, but not uniformly. The left plot of Figure 5.9 shows that a very sparse viewsphere sampling greatly improves matching at almost no computational cost for all detectors. However, after reaching a certain density, additional views do not add correspondences in the hardest cases – see the right graph of Figure 5.9. The ORB detector-descriptor clearly outperforms other detectors in terms of speed, but fails to match all images with the maximum tilt difference. The Hessian-Affine shows the best performance and it matched all pairs.

5.3.2 First geometrically inconsistent nearest neighbor ratio correspondence selection strategy

The following protocol was used to find the thresholds and to evaluate the performance of the proposed First Geometrically Inconsistent Nearest Neighbor *FGINN* strategy. First, similarity covariant regions were detected using the DoG detector (we also tried Hessian-Affine, MSER and SURF, with very similar results) and described using four popular descriptors – RootSIFT, SURF, LIOP [296] and MROGH [72] which are typically matched with the second-nearest region SNN strategy. Then for each keypoint descriptor, the first, second, and first geometrically inconsistent descriptors in the other image were found. The matching keypoints were then labeled as correct if their Sampson error was within 1 pixel of the ground truth location given by homography for the image pair, and incorrect otherwise.

Table 5.4: View synthesis configurations with best synthetic dataset performance.

| Local feature | Configurations | | |
|---------------|---|---|---|
| | EASY | MEDIUM | HARD |
| DoG-SIFT | $\{t\} = \{1; 5; 9\}, \Delta\phi = 360^\circ/t$ | $\{t\} = \{1; 2; 3; 4; 5; 6; 7; 8; 9\}, \Delta\phi = 180^\circ/t$ | $\{t\} = \{1; 2; 4; 6; 8\}, \Delta\phi = 60^\circ/t$ |
| HarrAff-SIFT | $\{t\} = \{1; 3; 6; 9\}, \Delta\phi = 360^\circ/t$ | $\{t\} = \{1; 2; 3; 4; 5; 6; 7; 8; 9\}, \Delta\phi = 360^\circ/t$ | $\{t\} = \{1; 2; 3; 4; 5; 6; 7; 8; 9\}, \Delta\phi = 120^\circ/t$ |
| HessAff-SIFT | $\{t\} = \{1; 5; 9\}, \Delta\phi = 360^\circ/t$ | $\{t\} = \{1; 5; 9\}, \Delta\phi = 360^\circ/t$ | $\{t\} = \{1; 2; 4; 6; 8\}, \Delta\phi = 60^\circ/t$ |
| MSER-SIFT | $\{t\} = \{1; 8\}, \Delta\phi = 360^\circ/t$ | $\{t\} = \{1; \sqrt{2}; 2; 2\sqrt{2}; 4; 4\sqrt{2}; 8\}, \Delta\phi = 60^\circ/t$ | $\{t\} = \{1; 3; 6; 9\}, \Delta\phi = 60^\circ/t$ |
| AGAST-FREAK | $\{t\} = \{1; 4; 8; 10\}, \Delta\phi = 360^\circ/t$ | $\{t\} = \{1; 3; 6; 9\}, \Delta\phi = 60^\circ/t$ | $\{t\} = \{1; 3; 6; 9\}, \Delta\phi = 72^\circ/t$ |
| ORB | $\{t\} = \{1; 5; 9\}, \Delta\phi = 360^\circ/t$ | $\{t\} = \{1; 2; 3; 4; 5; 6; 7; 8; 9\}, \Delta\phi = 90^\circ/t$ | $\{t\} = \{1; \sqrt{2}; 2; 2\sqrt{2}; 4; 4\sqrt{2}; 8\}, \Delta\phi = 72^\circ/t$ |
| SURF-FREAK | $\{t\} = \{1; \sqrt{2}; 2; 2\sqrt{2}; 4; 4\sqrt{2}; 8\}, \Delta\phi = 60^\circ/t$ | $\{t\} = \{1; \sqrt{2}; 2; 2\sqrt{2}; 4; 4\sqrt{2}; 8\}, \Delta\phi = 90^\circ/t$ | $\{t\} = \{1; \sqrt{2}; 2; 2\sqrt{2}; 4; 4\sqrt{2}; 8\}, \Delta\phi = 72^\circ/t$ |
| SURF-SURF | $\{t\} = \{1; 5; 9\}, \Delta\phi = 360^\circ/t$ | $\{t\} = \{1; 2; 3; 4; 5; 6; 7; 8; 9\}, \Delta\phi = 360^\circ/t$ | $\{t\} = \{1; 2; 3; 4; 5; 6; 7; 8; 9\}, \Delta\phi = 72^\circ/t$ |

The experiment was performed on 26 image pairs of the publicly available datasets [155],[54] (image pairs 1-3, precise homography provided) and [126] (the homography was estimated using the provided precise ground truth correspondences). The recall-precision curves for correspondences from all images were plotted with a varying ratio threshold from 0 to 1 in Figure 5.10. The FGINN curves for SIFT and SURF slightly outperform standard SNNs, while for LIOP and MROGH the difference is much more significant. The significantly higher benefit of the FGINN rule for LIOP and MROGH can be explained by their lower sensitivity to keypoint shift, which in turn means that undesirable suppression of keypoints happens in a larger neighborhood. The lower sensitivity to shifts was experimentally verified.

5.4 Experiments

We have tested MODS and, as a baseline, ASIFT³ and single detector configurations specified in Table 5.4 on seven public. datasets [169],[164],[10], [93], [113], [4], [157].

Implementation details of the MODS algorithm and parameter setting The kd-tree algorithm from FLANN library [171] was used to efficiently find the N-closest descriptors. The distance ratio thresholds of the FGINN matching strategy were experimentally selected based on the CDFs of matching and non-matching descriptors.

The MODS algorithm allows to set the minimum desired number of inliers which have a very low probability to be a random result as a stopping criterion. The recommended value – 15 inliers for the homography – did not produce false positive results in experiments. Computations were performed on Intel i3 CPU @ 2.6GHz with 4Gb RAM with 4 cores.

5.4.1 MODS variants testing on Extreme Viewpoint and Oxford Dataset

To evaluate the performance of matching algorithms, we introduce a two-view matching evaluation dataset⁴ with extreme viewpoint changes, see Table 5.5. The dataset includes image pairs from publicly available datasets: ADAM and MAG [170], GRAF [152] and THERE [54]. The ground truth homography matrices were estimated by LO-RANSAC using correspondences from all detectors in view synthesis configuration $\{t\} = \{1; \sqrt{2}; 2; 2\sqrt{2}; 4; 4\sqrt{2}; 8\}, \Delta\phi = 72^\circ/t$. The number of inliers for each image pair was ≥ 50 and the homographies were manually inspected. For the image pairs GRAF and THERE precise homographies are provided by Cordes *et al.* [54]. Transition tilts τ were computed using equation (5.1) with SVD decomposition of the linearized homography at center of the first image of the pair (see Table 5.5). Oxford [152] dataset with 42 image pairs (1-2, ..., 1-6) was used for easier wide baseline problems.

³Reference code from http://demo.ipol.im/demo/my_affine_sift

⁴Available at <http://cmp.felk.cvut.cz/wbs/index.html>

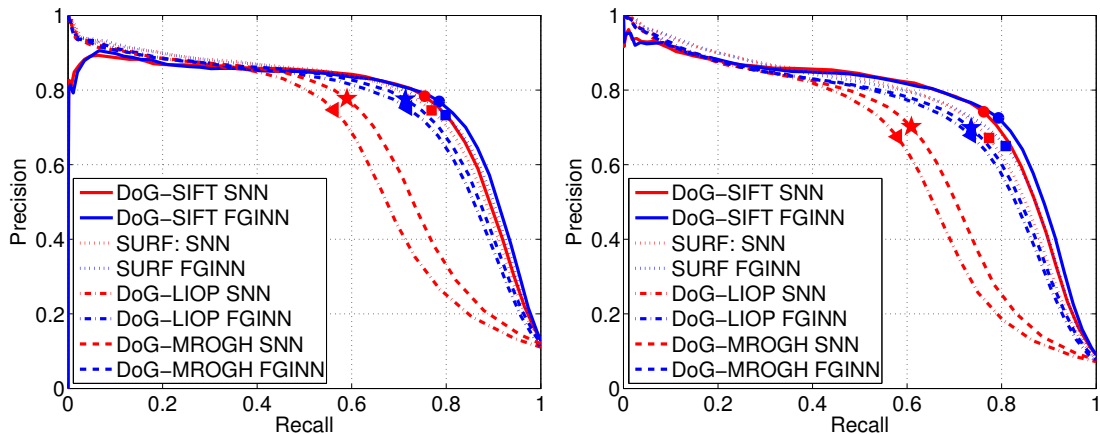


Figure 5.10: Comparison of the FGINN and SNN matching strategies for SIFT, SURF, LIOP and MROGH on images from the Oxford [155] and Cordes *et al.* [54] datasets. Markers show operating points for the common distance ratio threshold = 0.8. Matching without (left) view synthesis and using view synthesis (right) with parameters $\{t\} = \{1; 5; 9\}$, $\Delta\phi = 360^\circ/t$. The recall and precision of the correspondence filtering step, therefore the maximum recall is 1 when all correspondences are kept.

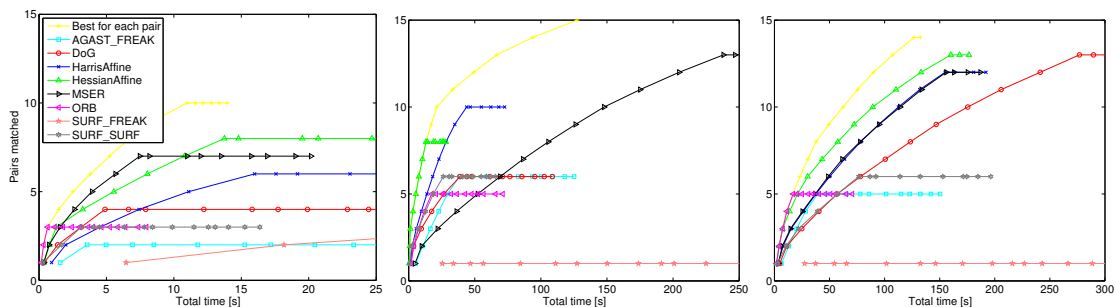


Figure 5.11: Performance of different configurations on the Extreme View Dataset. Cumulative percentage of image pairs successfully matched in a given time for configurations defined in Table 5.4. Each mark represents one image pair. The fastest detector configuration which was able to match each pair was selected. Left – ‘easy’, middle – ‘medium’, right – ‘hard’ configurations. An image is considered matched if 10 correct inliers were found.

Experimental protocol The evaluated algorithms matched image pairs and the output key-points correspondences were checked with ground truth homographies. The image pair is considered as solved, when at least 10 output correspondences are correct.

Figure 5.11 compares the different view synthesis configurations. Note that no single detector solved all image pairs. The Hessian-Affine, MSER, Harris-Affine and DoG successfully solved resp. 13, 13, 12, and 13 out of the 15 image pairs, however, at the expense of the high computational cost. We also noticed that if one would know the suitable detector and configuration for each image, it is possible to match all image pairs.

The MODS algorithm with more time-consuming configurations solves all image pairs and does it faster than a suitable configuration for each image pair – see Figure 5.12. We have tested several variants of the MODS configurations, stated in Table 5.2. Experiments show that the proposed MODS configuration is very fast on the easy WBS problems as in Oxford dataset (see Figure 5.12, right graph) and has very little overhead on the harder EVD dataset – it is the second best after the configuration without ORB steps. The results of the MODS medium configuration – without the first sparse synthesis step – shows the fruitfulness of the progressive view synthesis.

ASIFT is able to match only 6 image pairs from the dataset. The ASIFT algorithm generates a lower number of correct inliers and works slower than our identical DoG configuration (which has the same tilt-rotation set). The main causes are the elimination of “one-to-many”, including

Table 5.5: The Extreme View Dataset – EVD. Image sources: C – [54], Ox – [152], M – [170].

| # | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|--------------|-----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| Name | THERE | GRAF | ADAM | MAG | GRAND | PKK | FACE | GIRL | SHOP | DUM | INDEX | CAFE | FOX | CAT | VIN |
| Ref. | C | Ox | M | M | EVD | EVD | EVD | EVD | EVD | EVD | EVD | EVD | EVD | EVD | EVD |
| τ -tilt | 6.3 | 3.6 | 4.8 | 20 | 2.9 | 7.1 | 6.9 | 8.0 | 9.1 | 6.9 | 8.5 | 11.9 | 22.5 | 47 | 49.8 |
| Size | 1536 | 800 | 600 | 600 | 1000 | 1000 | 1000 | 1000 | 1000 | 1000 | 1000 | 800 | 1000 | 1000 | 1000 |
| [px] | x 1024 | x 640 | x 450 | x 450 | x 667 | x 750 | x 750 | x 750 | x 562 | x 729 | x 750 | x 533 | x 563 | x 598 | x 715 |



Table 5.6: ASIFT view synthesis configuration

| Detector | View synthesis setup |
|----------|--|
| DoG | $\{S\} = \{1\}$, $\{t\} = \{1; \sqrt{2}; 2; 2\sqrt{2}; 4; 4\sqrt{2}; 8\}$, $\Delta\phi = 72^\circ/t$ |

correct, correspondences, the inferiority of the standard second closest ratio matching strategy, and a simple brute-force algorithm of matching used in ASIFT.

Fig. 5.13 shows the breakdown of the computational time. The most time-consuming parts – detection and description (including the dominant orientation estimation) – take 40% and 35% resp. of all time. Without applying the fast SIFT computation from [129], the SIFT description takes more than 50% of the time. The ORB is an exception - the synthesis is not so profitable, since it takes more time than the detection and description itself. Note that the whole process is almost linear in the area of the synthesized views. The only super-linear part, matching, takes only 10% of the time.

5.4.2 MODS testing on other datasets

Extreme zoom dataset

We introduce the Extreme Zoom dataset (EZD), which is a small subset of the retrieval dataset used in [157]. It consists of six sets of images with an increasing level of zoom (see examples in Figure 5.14). The state-of-the-art matcher – ASIFT [170] and registration algorithm DualBootstrap [282] as well as results for MSER, ORB and Hessian-Affine matchers without view synthesis were compared to MODS. Image pairs are matched with the tested algorithms. An image pair is considered solved when at least 10 output correspondences are correct. Results are shown in Table 5.7.

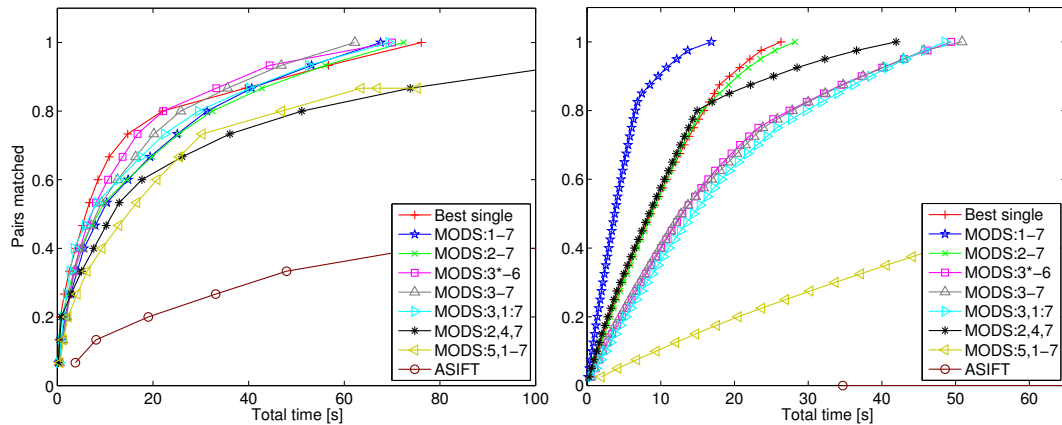


Figure 5.12: Performance of MODS configurations specified in Table 5.2. Cumulative percentage of image pairs successfully matched in time. Left - EVD dataset, right - Oxford dataset. The graphs are cropped. Each mark represents one image pair. The fastest single detector configuration which is able to match each pair was selected and plotted separately. Image considered matched if 10 correct inliers was found.

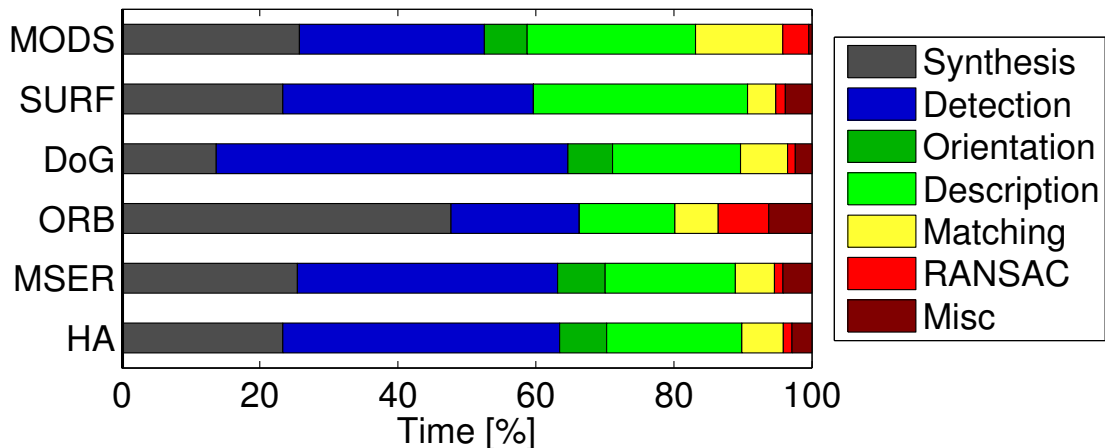


Figure 5.13: Percentages of time spent in the main stages of the matching with view synthesis process on a single core, EASY configuration. Detection and SIFT description, i.e. the dominant gradient estimation and the descriptor computation are the most time-consuming parts.

Ultra wide baseline dataset

The MODS performance was evaluated on the city-from-air dataset from [10]. The dataset comprises 30 pairs (examples shown in Figure 5.15) of photographs of buildings taken from the air. The viewpoint difference is quite large, the images contain repeated structures, illumination differs. The authors proposed a matcher based on HoG [58] descriptor with a view synthesis and compare it to ASIFT and D-Nets [276] for skyscraper frontal face matching. We follow the evaluation protocol which considers a pair matched correct only if the facade plane is matched ($\geq 75\%$ correct inliers). If the output homography was ground/roof, it is considered incorrect. The results are shown in Table 5.8.

Note that no special adjustment is done in MODS for homography selection, so the reported performance is a lower bound.

Datasets with other than geometric changes

Despite being designed for (extreme) wide baseline stereo problems, MODS performance was evaluated on other datasets: GDB-ICP [113] (modality, viewpoint and photometry changes), SymBench [93] (photometrical changes and photo-vs-painting pairs), and MMS [4] (infrared-

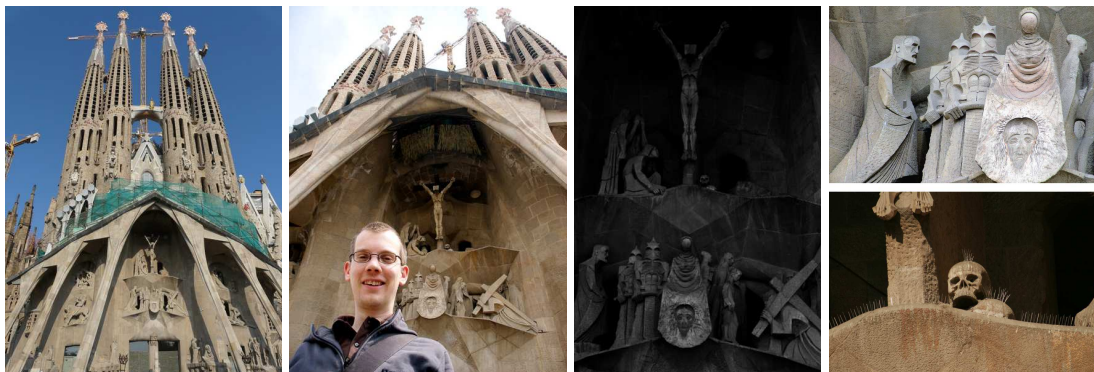


Figure 5.14: Examples from the Extreme Zoom Dataset

Table 5.7: Results on the Extreme Zoom Dataset.

| Matcher | Zoom level, matched | | | |
|---------|---------------------|------------|-------------|------------|
| | I (max 6) | II (max 6) | III (max 6) | IV (max 4) |
| ORB | 0 | 0 | 0 | 0 |
| MSER | 2 | 2 | 1 | 0 |
| DBstrap | 3 | 1 | 0 | 0 |
| HessAff | 5 | 3 | 2 | 0 |
| ASIFT | 3 | 3 | 2 | 0 |
| MODS | 6 | 3 | 3 | 0 |

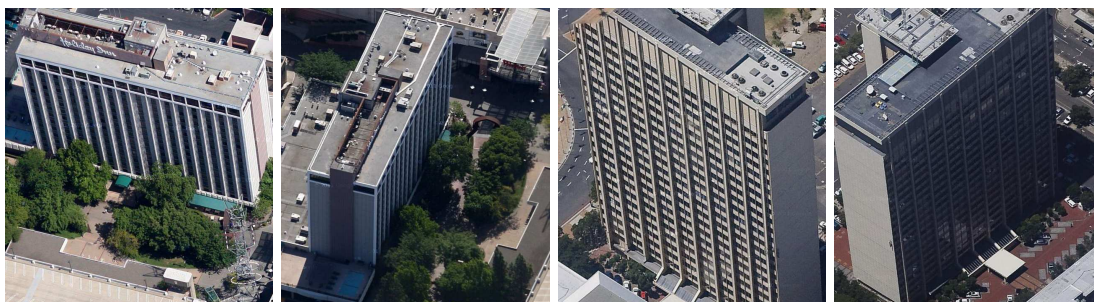


Figure 5.15: Examples of image pairs from the Ultra wide-baseline dataset [10]

vs-visible pairs) – see Table 5.9. The state-of-the-art matcher – ASIFT [170] and registration algorithm DualBootstrap [282] as well, as the results of MSER, ORB and Hessian-Affine matchers without view synthesis were compared to MODS.

Image pairs are matched with the tested algorithms. Output keypoint correspondences were checked against the ground truth homographies. An image pair is considered solved when at least 10 output correspondences are correct. Our primary evaluation criterion is the ability to find sufficiently correct geometric transformations in a reasonable time; accurate geometry can be found in a consecutive step.

The computations were performed on Intel i3 3.0GHz desktop (4 cores) with 4Gb RAM.

Results are shown in Table 5.10. MODS is the fastest method and it is able to match most image pairs in GDB-ICP and SymBench datasets without using symmetrical parts or other problem-specific features. Images from the MMS dataset (as well as other thermal images) produce a small number of features as they do not contain many textured surfaces and have a very short geometrical baseline. Those are the main reasons why area-based method – Dual-Bootstrap – works significantly better than the feature-based methods.

After lowering the threshold for detectors allowing to detect more feature points and using the orientation restricted SIFT [114] in addition to the RootSIFT, MODS-IR solved 83 out of 100 image pairs from MMS dataset.

Table 5.8: Results on Ultra wide-baseline dataset (all results except MODS taken from [10])

| Method | Correct (or shifted) | Different plane | Failure/ ground plane |
|-----------------------------|-------------------------|--------------------|--------------------------|
| Altwaijry and Belongie [10] | 9 | 1 | 20 |
| ASIFT-homography | 1 | 5 | 24 |
| D-Nets | 7 | 2 | 21 |
| MODS | 8 | 1 | 21 |

Table 5.9: Evaluation Datasets

| Short name | #images | Nuisance |
|---------------------|-----------|------------------------|
| GDB-ICP [113], 2007 | 22 pairs | Illumination, modality |
| SymBench [93], 2012 | 46 pairs | Illumination, modality |
| MMS [4], 2012 | 100 pairs | Modality |
| EVD [164], 2013 | 15 pairs | Viewpoint change |

Table 5.10: Performance on non-WBS datasets. For comparison, results of MSER, ORB and Hessian-Affine matchers without view synthesis are added. Best results are in **bold**, average time per image pair is shown.

| Matcher | GDB-ICP | | SymBench | | MMS | | EVD | |
|-------------|-----------------|-------------|-----------------|-------------|-----------------|-------------|-----------------|-------------|
| | pairs solved | time [s] | pairs solved | time [s] | pairs solved | time [s] | pairs solved | time [s] |
| ASIFT | 15/22 | 41.5 | 27/46 | 14.7 | 8/100 | 3.2 | 5/15 | 12.4 |
| MODS:1-7 | 17/22 | 2.8 | 38/46 | 3.7 | 12/100 | 2.0 | 15/15 | 2.4 |
| DBstrap | 16/22 | 17.6 | 38/46 | 21.7 | 79/100 | 9.3 | 0/15 | 1.9 |
| ORB | 0/22 | 0.2 | 0/46 | 0.4 | 0/100 | 0.1 | 0/15 | 0.3 |
| MSER | 8/22 | 1.7 | 21/46 | 0.8 | 0/100 | 0.2 | 4/15 | 0.6 |
| HessAff | 11/22 | 1.9 | 29/46 | 1.5 | 2/100 | 0.4 | 2/15 | 1.1 |
| MODS-IR:1-7 | 21/22 | 7.6 | 39/46 | 15.1 | 83/100 | 9.1 | 14/15 | 8.6 |

5.4.3 MODS testing on a non-planar dataset

The dataset and evaluation protocol

The evaluation dataset consists of 35 image sequences taken from the Turntable dataset [169] (“Bottom” camera) shown in Table 5.11. Eight image sets contain objects with relatively large planar surfaces and the remaining ones are low-textured, “general 3D” objects.

The view marked as “0°” in the Turntable dataset was used as a reference view and 0 – 90° and 270-355 ° views with a 5° step were matched against it using the procedure described in Sec. 5.4, forming a [−90°, 90°] sequence. Note that the reference view is not usually the “frontal” or “side” view, but rather some intermediate view which caused asymmetry in the results (see Fig. 5.17, Table 5.12).

The output of the matchers is a set of correspondences and the estimated geometrical transformation. The accuracy of the matched correspondences was chosen as the performance criterion, similarly to the protocol in [57]. For all output correspondences, the symmetrical epipolar error [90] e_{SymEG} was computed according to the following expression:

$$e_{\text{SymEG}}(\mathbf{F}, \mathbf{u}, \mathbf{v}) = (\mathbf{v}^\top \mathbf{F} \mathbf{u})^2 \times \left(\frac{1}{(\mathbf{F} \mathbf{u})_1^2 + (\mathbf{F} \mathbf{u})_2^2} + \frac{1}{(\mathbf{F}^\top \mathbf{v})_1^2 + (\mathbf{F}^\top \mathbf{v})_2^2} \right), \quad (5.2)$$

where \mathbf{F} – fundamental matrix, \mathbf{u}, \mathbf{v} – corresponding points, $(\mathbf{F} \mathbf{u})_j^2$ – the square of the j -th entry of the vector $\mathbf{F} \mathbf{u}$.

The ground truth fundamental matrix was obtained from the difference in camera positions [90], assuming that the turntable is fixed and the camera moves around the object, according to the

Table 5.11: Reference views of the image sequences used in the evaluation (from [169]).















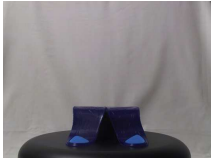











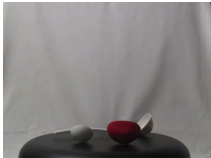





| | | | |
|---|--|---|--|
| Abroller  | Bannanas  | Camera2  | Car  |
| Car2  | CementBase  | Cloth  | Conch  |
| Desk  | Dinosaur  | Dog  | DVD  |
| FloppyBox  | FlowerLamp  | Gelsole  | GrandfatherClock  |
| Horse  | Keyboard  | Motorcycle  | MouthGuard  |
| PaperBin  | PS2  | Razor  | RiceCooker  |
| Rock  | RollerBlade  | Spoons  | TeddyBear  |
| Toothpaste  | Tricycle  | Tripod  | VolleyBall  |

Table 5.12: Experiment on the 3D dataset. The configurations are defined in Table 5.4. The number of correctly matched image pairs and the run-time per pair. Best results are boxed. Results within 90% of the best are in **bold**. The configurations are sorted by average time for image pair.

| Matcher | Image sets solved (out of 35) | | | | | | | | | | Time [s] |
|----------------------|-------------------------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|----------|
| | Viewpoint angular difference | | | | | | | | | | |
| | 0° | 5° | 10° | 15° | 20° | 25° | 30° | 35° | 40° | ≥ 45° | |
| HessianAffine easy | 35 | 32 | 29 | 26 | 22 | 17 | 13 | 14 | 9 | 6 | 0.9 |
| SURF-SURF easy | 35 | 31 | 24 | 18 | 15 | 12 | 7 | 5 | 4 | 2 | 0.9 |
| HessianAffine medium | 35 | 30 | 26 | 24 | 18 | 17 | 12 | 10 | 7 | 10 | 1.0 |
| ORB easy | 35 | 31 | 27 | 20 | 16 | 11 | 10 | 5 | 5 | 4 | 1.0 |
| AGAST-FREAK easy | 35 | 28 | 29 | 26 | 19 | 17 | 12 | 9 | 7 | 6 | 1.3 |
| DoG easy | 35 | 33 | 30 | 22 | 18 | 12 | 10 | 7 | 5 | 3 | 1.4 |
| MSER easy | 35 | 28 | 22 | 21 | 15 | 13 | 10 | 8 | 7 | 5 | 1.5 |
| SURF-SURF hard | 35 | 26 | 16 | 11 | 9 | 7 | 6 | 4 | 3 | 3 | 2.1 |
| HarrisAffine easy | 35 | 29 | 22 | 12 | 8 | 6 | 5 | 3 | 2 | 1 | 2.7 |
| AGAST-FREAK hard | 35 | 31 | 28 | 25 | 20 | 17 | 15 | 10 | 8 | 5 | 4.7 |
| HarrisAffine medium | 35 | 27 | 20 | 11 | 7 | 6 | 5 | 4 | 3 | 2 | 4.7 |
| HessianAffine hard | 35 | 29 | 24 | 20 | 20 | 17 | 15 | 11 | 9 | 6 | 5.6 |
| DoG medium | 35 | 32 | 26 | 21 | 16 | 11 | 10 | 9 | 6 | 5 | 6.8 |
| AGAST-FREAK medium | 35 | 29 | 23 | 22 | 18 | 14 | 13 | 6 | 6 | 6 | 7.2 |
| ORB hard | 35 | 31 | 24 | 19 | 16 | 7 | 5 | 4 | 1 | 4 | 7.3 |
| SURF-FREAK easy | 35 | 25 | 21 | 13 | 10 | 8 | 6 | 3 | 2 | 2 | 7.3 |
| ORB medium | 35 | 30 | 26 | 18 | 17 | 10 | 5 | 4 | 2 | 4 | 8.3 |
| SURF-FREAK medium | 35 | 22 | 15 | 10 | 9 | 7 | 6 | 4 | 1 | 2 | 9.1 |
| SURF-SURF medium | 35 | 25 | 20 | 13 | 11 | 7 | 3 | 3 | 2 | 2 | 10.7 |
| SURF-FREAK hard | 35 | 25 | 17 | 14 | 10 | 9 | 4 | 3 | 1 | 2 | 10.9 |
| MSER hard | 35 | 29 | 24 | 24 | 19 | 16 | 12 | 13 | 8 | 8 | 14.5 |
| HarrisAffine hard | 35 | 27 | 16 | 8 | 5 | 5 | 5 | 3 | 2 | 2 | 15.2 |
| DoG hard | 35 | 32 | 24 | 17 | 16 | 13 | 12 | 8 | 7 | 8 | 16.7 |
| MSER medium | 35 | 29 | 26 | 23 | 21 | 14 | 10 | 9 | 7 | 9 | 17.5 |
| ASIFT | 35 | 32 | 24 | 18 | 13 | 8 | 7 | 5 | 4 | 3 | 27.6 |
| MODS:1-7 | 35 | 31 | 27 | 27 | 22 | 22 | 14 | 9 | 10 | 11 | 6.7 |

following equation:

$$F = K^{-\top} R K^{\top} [K R^{\top} t]_{\times},$$

$$R = \begin{pmatrix} \cos \phi & 0 & -\sin \phi \\ 0 & 1 & 0 \\ \sin \phi & 0 & \cos \phi \end{pmatrix}, K = \begin{pmatrix} \frac{mf}{\text{FR}_x} & 0 & \frac{m}{2} \\ 0 & \frac{nf}{\text{FR}_y} & \frac{n}{2} \\ 0 & 0 & 1 \end{pmatrix}, t = r \begin{pmatrix} \sin \phi \\ 0 \\ 1 - \cos \phi \end{pmatrix}, \quad (5.3)$$

where R is the orientation matrix of the second camera, K – the camera projection matrix, t – the virtual translation of the second camera, r – the distance from camera to the object, ϕ – the viewpoint angle difference, FR_x, FR_y – the focal plane resolution, f – the focal length, m, n – the sensor matrix width and height in pixels. The last five parameters were obtained from EXIF data.

One of the evaluation problems is that background regions, i.e. regions that are not on the object placed on the turntable, are often detected and matched, influencing the geometry transformation estimation. The matches are correct, but consistent with an identity transform of the (background of) the test images, not the fundamental matrix associated with the movement of the object on the turntable. To solve this problem, the median value of the correspondence errors was chosen as the measure of precision because of its tolerance to the low number of outliers (e.g. the above-mentioned background correspondences), and its sensitivity to the incorrect geometric model estimated by RANSAC.

An image pair is considered as *correctly* matched if the median symmetrical epipolar error on the correspondences using the ground truth fundamental matrix is ≤ 6 pixels.

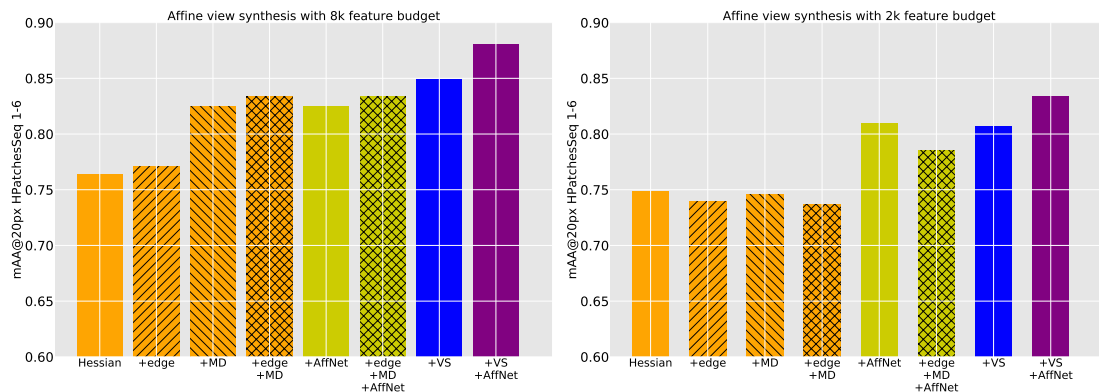


Figure 5.16: Ablation study of the different benefit from the affine view synthesis: edge – ability to detect elongated blobs by relaxed edge threshold, MD – multiple affine descriptors, AffNet – affine shape estimation by AffNet, VS – view synthesis. Feature budget: left – 8k, right – 2k

Results

Figure 5.17 and Table 5.12 show the percentage and the number of image sequences respectively for which the reference and tested views for the given viewing angle difference were matched correctly.

The difference between EASY, MEDIUM and HARD configurations is small for structured scenes – unlike planar ones. Difficulties in matching are caused not by the inability to detect distorted regions but by object self-occlusions. Therefore the synthesis of additional views does not bring more correspondences.

Experiments with view synthesis confirmed results [169] that the Hessian-Affine outperforms other detectors for matching of structured scenes and can be used alone in such scenes. MODS shows similar performance, but is slower than the Hessian-Affine configuration.

The computations were performed on Intel i5 3.0GHz (4 cores) desktop with 16Gb RAM. Examples of the matched images are shown in Fig. 5.18.

5.5 Why does affine synthesis help?

ASIFT and other view-synthesis based approaches are known more than decade. However, we are not aware of a study explaining why affine synthesis helps in practice. Could one get a similar performance without view synthesis? To answer this question, we performed an ablation study to test the following hypothesis:

0. May it be that the most of improvements come from the fact that we have much more features? That is why we fix the number of features for all approaches.
1. Some regions from ASIFT, when reprojected to the original image, are quite narrow. Could we get them just by removing edge-like feature filtering, which is done in SIFT, Hessian and other detectors. Denoted **+edge**
2. Instead of doing affine view synthesis, one could directly use the same affine parameters to get the affine regions to describe, so the each keypoint would have several associated regions+descriptors. Denoted **+MD**
3. Using AffNet to directly estimated local affine shape without multiple descriptors. Denoted **+AffNet**
4. Combine (1), (2) and (3).

The study was conducted on the HPatches Sequences dataset[16], the hardest image pairs (1-6) of the viewpoint subset. The metric is similar to the one used in [109] – the mean average accuracy. Instead of the camera pose we threshold the reprojection error of the estimated homography. First, the mean reprojection error for the part visible in both images is calculated. Second, a

sequence of thresholds in \log_2 space from 1.0 to 20.0 is applied. Finally, all per-threshold accuracies are averaged into the final score.

We run the Hessian detector with the RootSIFT descriptor, FLANN matching, and LO-RANSAC, as implemented in MODS. Features are sorted according to the detector response and their total number is clipped to 2048 and 8000 to ensure that the improvements do not come from just having more features.

Note that we do not study the view synthesis effect on regular image pairs. Instead, we were focusing on the case when view synthesis is required: matching oblique views of mostly planar scenes.

Results are shown in Figure 5.16. Indeed, all of the factors: detecting more edge-like features, having multiple descriptors, or better affine shape improve the results over the plain Hessian detector. However, even all combined are not enough to match performance of the affine view synthesis + plain Hessian detector. Yet, the best setup is to use both Hessian-AffNet and view synthesis. The picture is a bit different in a small feature budget: neither multiple-(affine)-descriptors per keypoint, nor allowing edge-like feature help. On the other hand, affine view synthesis still improves results of the Hessian. And, again, the best performance is achieved with a combination of view synthesis and AffNet shape estimation.

5.6 Summary

An algorithm for two-view matching called Matching On Demand with view Synthesis algorithm (MODS) was introduced. The most important contributions of the algorithm are its ability to adjust its complexity to the problem at hand, and its robustness, i.e., the ability to solve a broader range of wide-baseline problems than the state of the art. This is achieved while being fast on simple problems.

The apparent robustness vs. speed trade-off is finessed by the use of progressively more time-consuming feature detectors, and by on-demand generation of synthesized images that is performed until a reliable estimate of the geometry is obtained. The MODS method demonstrates that the answer to the question "which detector is the best?" depends on the problem at hand, and that it is fruitful to focus on the "how to combine detectors" problem.

We are the first to propose a view synthesis for two-view wide-baseline matching with affine-covariant detectors, which is superficially counter-intuitive, and we show that matching with the Hessian-Affine or MSER detectors outperforms the state-of-the-art ASIFT. View synthesis performs well when used with simple and very fast detectors like ORB, which obtains results similar to ASIFT but in orders of magnitude shorter time.

Minor contributions include an improved method for tentative correspondence selection, applicable both with and without view synthesis and a modification of the standard first to second nearest distance due that increases the number of correct matches by 5-20% at no additional computational cost.

The evaluation of the MODS algorithm was carried out both on standard publicly available datasets as well as a new set of geometrically challenging wide baseline problems that we collected and will make public. The experiments show that the MODS algorithm solves matching problems beyond the state-of-the-art and yet is comparable in speed to standard wide-baseline matchers on easy problems. Moreover, MODS performs well on other classes of difficult two-view problems like matching of images from different modalities, with large difference of acquisition times or with significant lighting changes.

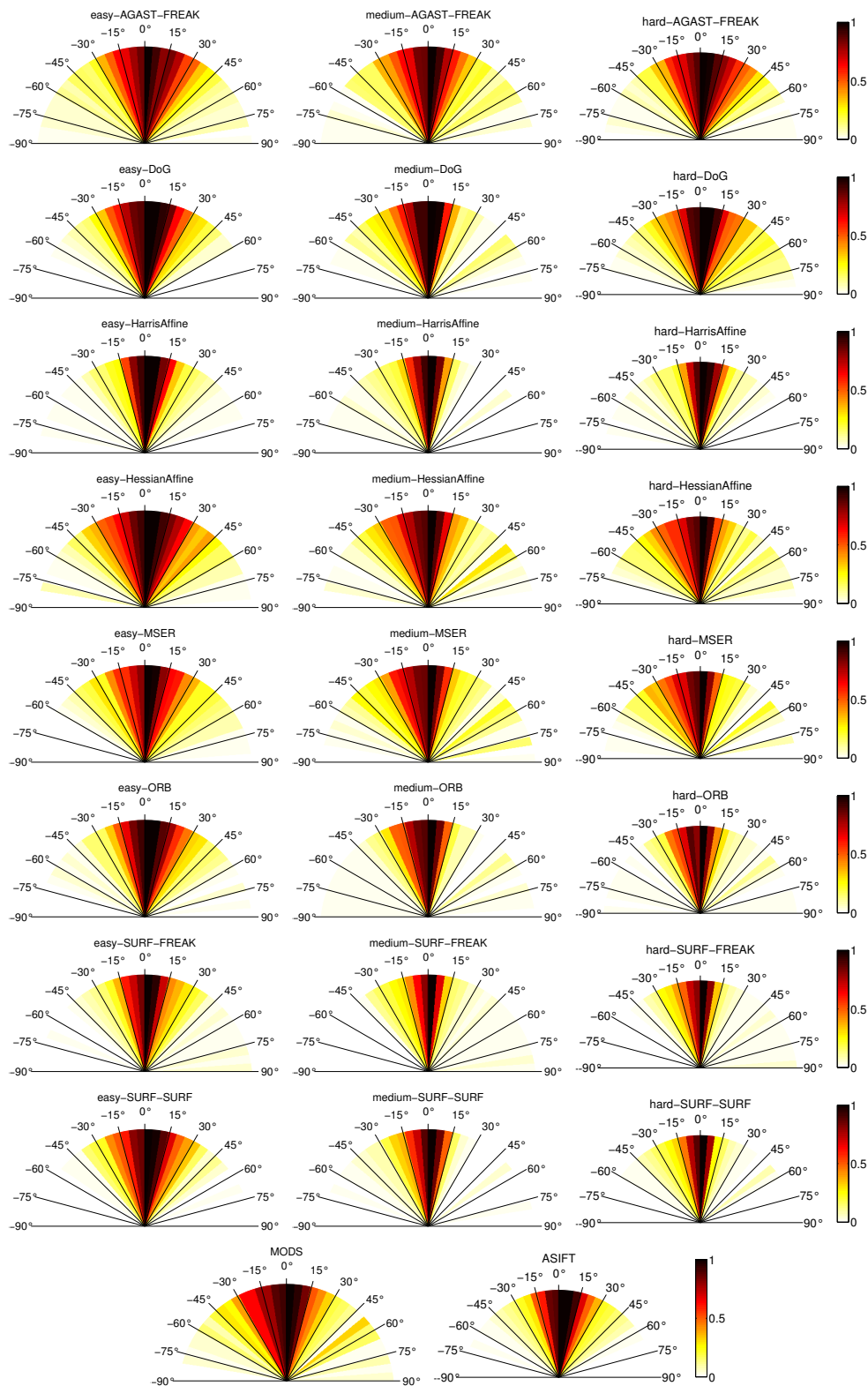


Figure 5.17: A comparison of view synthesis configurations on the Turntable dataset [169]. The fraction of correctly matched images for a given viewpoint difference.

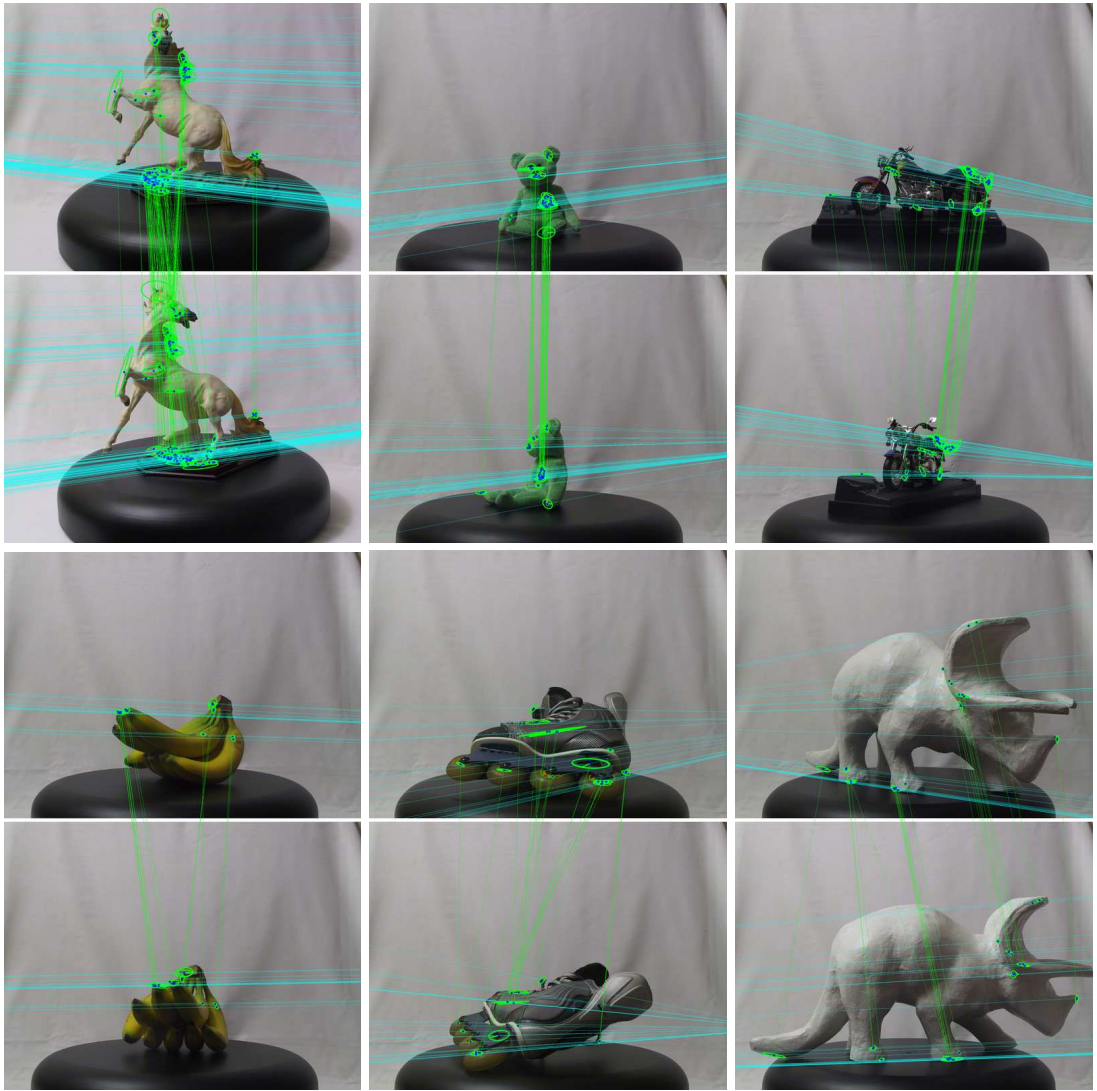
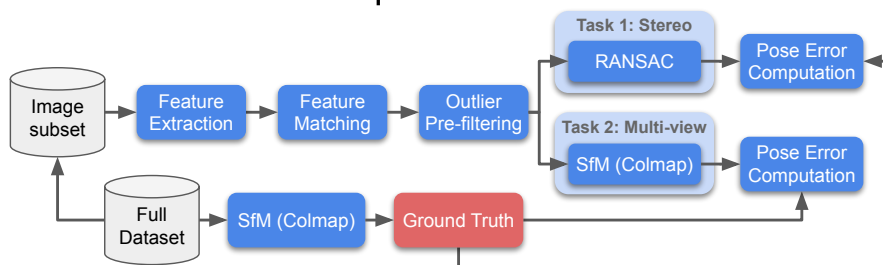


Figure 5.18: Correspondences found by MODS. Green – corresponding regions, cyan – epipolar lines. Objects with significant self-occlusion and mostly homogenous texture were selected.

CHAPTER 6

Image Matching Across Wide Baselines: From Paper to Practice



This Chapter presents a benchmark of the all components of the wide baseline stereo pipeline components. We carefully tune all the baselines and show importance of the match filtering strategies and modern RANSACs.

6.1 Introduction

Matching two or more views of a scene is at the core of fundamental computer vision problems, including image retrieval [140, 13, 203, 260, 179], 3D reconstruction [3, 96, 226, 299], re-localization [219, 217, 144], and SLAM [176, 61, 62]. Despite decades of research, image matching remains unsolved in the general wide-baseline scenario. Image matching is a challenging problem with many factors that need to be taken into account, e.g., viewpoint, illumination, occlusions, and camera properties. It has therefore been traditionally approached with sparse methods – that is, with local features.

Recent efforts have moved towards *holistic*, end-to-end solutions [115, 17, 41]. Despite their promise, they are yet to outperform the *separatists* [220, 298] that are based on the classical paradigm of step-by-step solutions. For example, in a classical wide baseline stereo pipeline [192], one (1) extracts local features, such as SIFT [140], (2) creates a list of tentative matches by nearest-neighbor search in descriptor space, and (3) retrieves the pose with a minimal solver inside a robust estimator, such as the 7-point algorithm [89] in a RANSAC loop [75]. To build a 3D reconstruction out of a set of images, same matches are fed to a bundle adjustment pipeline [90, 268] to jointly optimize the camera intrinsics, extrinsics, and 3D point locations. This modular structure simplifies the engineering a solution to the problem and allows for incremental improvements, of which there have been hundreds, if not thousands.

New methods for each of the sub-problems, such as feature extraction and pose estimation, are typically studied in isolation, using intermediate metrics, which simplifies their evaluation. However, there is no guarantee that gains in one part of the pipeline will translate to the final application, as these components interact in complex ways. For example, patch descriptors, including very recent work [94, 279, 256, 174], are often evaluated on Brown’s seminal patch retrieval database [36], introduced in 2007. They show dramatic improvements – up to 39x relative [279] – over handcrafted methods such as SIFT, but it is unclear whether this remains true on real-world applications. In fact, we later demonstrate that the gap narrows dramatically when decades-old baselines are properly tuned.

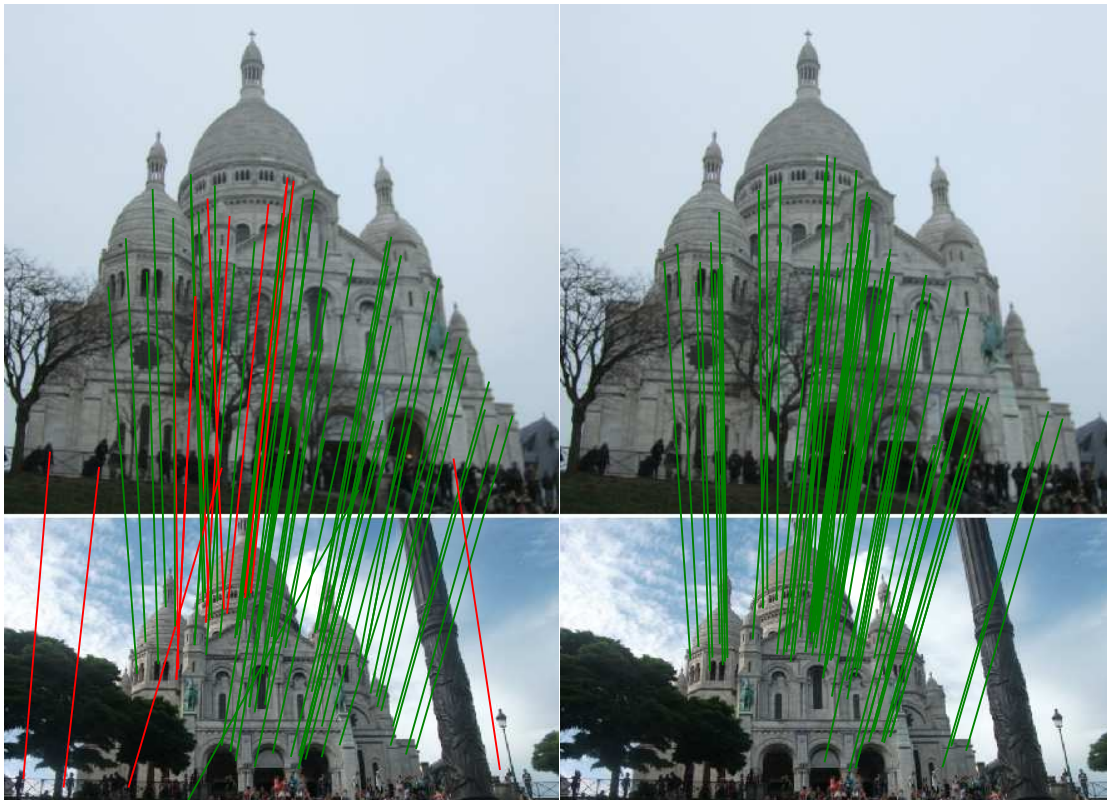


Figure 6.1: Every paper claims to outperform the state of the art. Is this possible, or an artifact of insufficient validation? On the left, we show stereo matches obtained with **D2-Net** (2019) [69], a state-of-the-art local feature, using OpenCV RANSAC with its default settings. We color the inliers in green if they are correct and in red otherwise. On the right, we show **SIFT** (1999) [140] with a carefully tuned MAGSAC [60] – notice how the latter performs much better. This illustrates our take-home message: to correctly evaluate a method’s performance, it needs to be embedded within the pipeline used to solve a given problem, and the different components in said pipeline need to be tuned carefully and jointly, which requires engineering and domain expertise. We fill this need with a new, modular benchmark for sparse image matching, incorporating dozens of built-in methods.

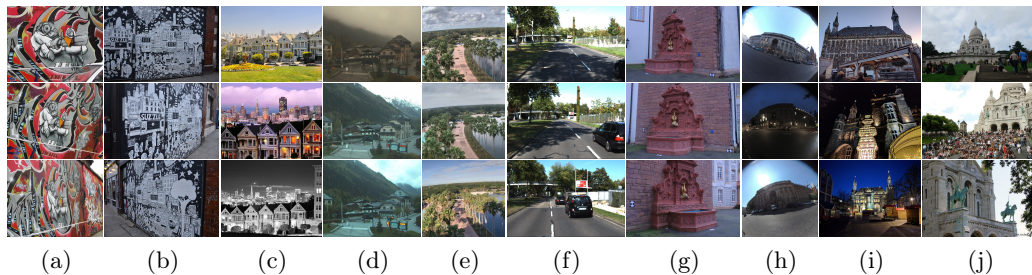


Figure 6.2: **On the limitations of previous datasets.** To highlight the need for a new benchmark, we show examples from datasets and benchmarks featuring posed images that have been previously used to evaluate local features and robust matchers (some images are cropped for the sake of presentation). From left to right: (a) VGG Oxford [151], (b) HPatches [16], (c) Edge Foci [300], (d) Webcam [274], (e) AMOS [103], (f) Kitti [81], (g) Strecha [243], (h) SILDa [71], (i) Aachen [219], (j) Ours. Notice that many (a-e) contain only planar structures or illumination changes, which makes it easy – or trivial – to obtain ground truth poses, encoded as homographies. Other datasets are small – (g) contains very accurate depth maps, but only two scenes and 19 images total – or do not contain a wide enough range of photometric and viewpoint transformations. Aachen (i) is closer to ours (j), but relatively small, limited to one scene, and focused on re-localization on the day vs. night use-case.

We posit that it is critical to look beyond intermediate metrics and focus on downstream performance. This is particularly important *now* as, while deep networks are reported to outperform algorithmic solutions on classical, sparse problems such as outlier filtering [284, 205, 290, 248, 34], bundle adjustment [251, 233], SfM [275, 11] and SLAM [253, 120], our findings in this Chapter suggest that this may not always be the case. To this end, we introduce a benchmark for wide-baseline image matching, including:

- (a) A dataset with thousands of phototourism images of 25 landmarks, taken from diverse viewpoints, with different cameras, in varying illumination and weather conditions – all of which are necessary for a comprehensive evaluation. We reconstruct the scenes with SfM, without the need for human intervention, providing depth maps and ground truth poses for 26k images, and reserve another 4k for a private test set.
- (b) A modular pipeline¹ incorporating dozens of methods for feature extraction, matching, and pose estimation, both classical and state-of-the-art, as well as multiple heuristics, all of which can be swapped out and tuned separately.
- (c) Two downstream tasks – stereo and multi-view reconstruction – evaluated with both downstream and intermediate metrics, for comparison.
- (d) A thorough study of dozens of methods and techniques, both hand-crafted and learned, and their combination, along with a recommended procedure for effective hyper-parameter selection.

The framework enables researchers to evaluate how a new approach performs in a standardized pipeline, both *against* its competitors², and *alongside* state-of-the-art solutions for other components, from which it simply cannot be detached. This is crucial, as the true performance can be easily hidden by sub-optimal hyperparameters.

6.2 Related Work

The literature on image matching is too vast for a thorough overview. We cover relevant methods for feature extraction and matching, pose estimation, 3D reconstruction, applicable datasets, and evaluation frameworks.

¹<https://github.com/vcg-uvic/image-matching-benchmark>

²<https://vision.uvic.ca/image-matching-challenge>

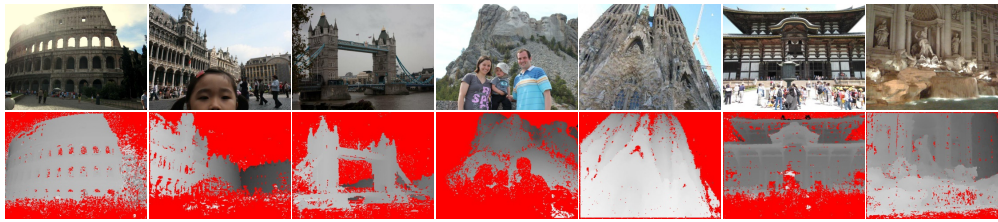


Figure 6.3: **The Image Matching Challenge PhotoTourism (IMC-PT) dataset.** We show a few selected images from our dataset and their corresponding depth maps, with occluded pixels marked in red.

6.2.1 Local features

Local features became a staple in computer vision with the introduction of SIFT [140]. They typically involve three distinct steps: keypoint detection, orientation estimation, and descriptor extraction. Other popular classical solutions are SURF [26], ORB [212], and AKAZE [7].

Modern descriptors often train deep networks on pre-cropped patches, typically from SIFT keypoints (*i.e.* Difference of Gaussians or DoG). They include Deepdesc [234], TFeat [18], L2-Net [288], HardNet [158], SOSNet [256], and LogPolarDesc [70] – most of them are trained on the same dataset [36]. Recent works leverage additional cues, such as geometry or global context, including GeoDesc [143] and ContextDesc [142]. There have been multiple attempts to learn keypoint detectors separately from the descriptor, including TILDE [274], TCDet [293], QuadNet [221], and Key.Net [23]. An alternative is to treat this as an end-to-end learning problem, a trend that started with the introduction of LIFT [286] and includes DELF [179], SuperPoint [62], LF-Net [181], D2-Net [69] and R2D2 [207].

6.2.2 Robust matching

Inlier ratios in wide-baseline stereo can be below 10% – and sometimes even lower. This is typically approached with iterative sampling schemes based on RANSAC [75], relying on closed-form solutions for pose solving such as the 5- [178], 7- [89] or 8-point algorithm [91]. Improvements to this classical framework include local optimization [50], using likelihood instead of reprojection (MLE-SAC) [265], speed-ups using probabilistic sampling of hypotheses (PROSAC) [49], degeneracy check using homographies (DEGENSAC) [51], Graph Cut as a local optimization (GC-RANSAC) [20], and auto-tuning of thresholds using confidence margins (MAGSAC) [60].

As an alternative direction, recent works, starting with CNe (Context Networks) in [284], train deep networks for outlier rejection taking correspondences as input, often followed by a RANSAC loop. Follow-ups include [205, 295, 248, 290]. Differently from RANSAC solutions, they typically process all correspondences in a single forward pass, without the need to iteratively sample hypotheses. Despite their promise, it remains unclear how well they perform in real-world settings compared to a well-tuned RANSAC.

6.2.3 Structure from Motion (SfM)

In SfM, one jointly optimizes the location of the 3D points and the camera intrinsics and extrinsics. Many improvements have been proposed over the years [3, 96, 56, 80, 299]. The most popular frameworks are VisualSfM [281] and COLMAP [226] – we rely on the latter to both generate the ground truth and as the backbone of our multi-view task.

6.2.4 Datasets and benchmarks

Early works on local features and robust matchers typically relied on the Oxford dataset [151], which contains 48 images and ground truth homographies. It helped establish two common metrics for evaluating local feature performance: repeatability and matching score. Repeatability evaluates the keypoint detector: given two sets of keypoints over two images projected into each other, it is defined as the ratio of keypoints whose support regions’ overlap is above a threshold. The matching score (MS) is similarly defined, but also requires their descriptors to be the nearest

neighbours. Both require pixel-to-pixel correspondences – *i.e.*, features outside valid areas are ignored.

A modern alternative to Oxford is HPatches [16], which contains 696 images with differences in illumination *or* viewpoint – but not both. However, the scenes are planar, without occlusions limiting its applicability.

Other datasets that have been used to evaluate local features include DTU [1], Edge Foci [300], Webcam [274], AMOS [198], and Strecha’s [243]. They all have limitations – be it narrow baselines, noisy ground truth, or a small number of images. In fact, most learned descriptors have been trained and evaluated on [38], which provides a database of pre-cropped patches with correspondence labels, and measures the performance in terms of patch retrieval. While this seminal dataset and evaluation methodology helped develop many new methods, it is not clear how the results translate to different scenarios – particularly since new methods outperform classical ones such as SIFT by orders of magnitude, which suggests overfitting.

Datasets used for navigation, re-localization, or SLAM, in outdoor environments are also relevant to our problem. These include KITTI [81], Aachen [219], Robotcar [145], and CMU seasons [217, 15]. However, they do not feature the wide range of transformations present in the phototourism data. Megadepth [134] is a more representative, phototourism-based dataset, which using COLMAP, as in our case – it could, in fact, easily be folded into ours.

Benchmarks, by contrast, are few and far between – they include VLBenchmark [128], HPatches [16], and SILDa [71] – all limited in scope. A large-scale benchmark for SfM was proposed in [227], which built 3D reconstructions with different local features. However, only a few scenes contain ground truth, so most of their metrics are qualitative – *e.g.* number of observations or average reprojection error. Yi *et al.* [284] and Bian *et al.* [33] evaluate different methods for pose estimation on several datasets – however, few methods are considered and they are not carefully tuned.

We highlight some of these datasets/benchmarks, and their limitations in Fig. 6.2. We are, to the best of our knowledge, the first to introduce a public modular benchmark for 3D reconstruction with sparse methods using downstream metrics, and featuring a comprehensive dataset with a large range of image transformations.

6.3 The Image Matching Challenge PhotoTourism Dataset

While it is possible to obtain very accurate poses and depth maps under controlled scenarios with devices like LIDAR, this is costly and requires a specific set-up that does not scale well. For example, Strecha’s dataset [243] follows that approach but contains only 19 images. We argue that a truly representative dataset must contain a wider range of transformations – including different imaging devices, time of day, weather, partial occlusions, etc. Phototourism images satisfy this condition and are readily available.

We thus build on 25 collections of popular landmarks originally selected in [96, 254], each with hundreds to thousands of images. Images are downsampled with bilinear interpolation to a maximum size of 1024 pixels along the long-side and their poses were obtained with COLMAP [226], which provides the (pseudo) ground truth. We do exhaustive image matching before Bundle Adjustment – unlike [228], which uses only 100 pairs for each image – and thus provide enough matching images for any conventional SfM to return near-perfect results in standard conditions.

Our approach is to obtain a ground truth signal using reliable, off-the-shelf technologies, while making the problem as easy as possible – and then evaluate new technologies on a much harder problem, using only a subset of that data. For example, we reconstruct a scene with hundreds or thousands of images with vanilla COLMAP and then evaluate “modern” features and matchers against its poses using only two images (“stereo”) or up to 25 at a time (“multiview” with SfM). For a discussion regarding the accuracy of our ground truth data, please refer to Section 6.3.3.

In addition to point clouds, COLMAP provides dense depth estimates. These are noisy, and have no notion of occlusions – a depth value is provided for every pixel. We remove occluded pixels from depth maps using the reconstructed model from COLMAP; see Fig. 6.3 for examples. We rely on these “cleaned” depth maps to compute classical, pixel-wise metrics – repeatability and matching score. We find that some images are flipped 90°, and use the reconstructed pose to rotate them – along with their poses – so they are roughly ‘upright’, which is a reasonable assumption for this type of data.

| Name | Group | Images | 3D points |
|---------------------------------|-------|--------|-----------|
| “Brandenburg Gate” | T | 1363 | 100040 |
| “Buckingham Palace” | T | 1676 | 234052 |
| “Colosseum Exterior” | T | 2063 | 259807 |
| “Grand Place Brussels” | T | 1083 | 229788 |
| “Hagia Sophia Interior” | T | 888 | 235541 |
| “Notre Dame Front Facade” | T | 3765 | 488895 |
| “Palace of Westminster” | T | 983 | 115868 |
| “Pantheon Exterior” | T | 1401 | 166923 |
| “Prague Old Town Square” | T | 2316 | 558600 |
| “Reichstag” | T | 75 | 17823 |
| “Taj Mahal” | T | 1312 | 94121 |
| “Temple Nara Japan” | T | 904 | 92131 |
| “Trevi Fountain” | T | 3191 | 580673 |
| “Westminster Abbey” | T | 1061 | 198222 |
| “Sacre Coeur” (SC) | V | 1179 | 140659 |
| “St. Peter’s Square” (SPS) | V | 2504 | 232329 |
| Total T + V | | 25.7k | 3.6M |
| “British Museum” (BM) | E | 660 | 73569 |
| “Florence Cathedral Side” (FCS) | E | 108 | 44143 |
| “Lincoln Memorial Statue” (LMS) | E | 850 | 58661 |
| “London (Tower) Bridge” (LB) | E | 629 | 72235 |
| “Milan Cathedral” (MC) | E | 124 | 33905 |
| “Mount Rushmore” (MR) | E | 138 | 45350 |
| “Piazza San Marco” (PSM) | E | 249 | 95895 |
| “Sagrada Familia” (SF) | E | 401 | 120723 |
| “St. Paul’s Cathedral” (SPC) | E | 615 | 98872 |
| Total E | | 3774 | 643k |

Table 6.1: **Scenes in the IMC-PT dataset.** We provide some statistics for the training (T), validation (V), and test (E) scenes.

6.3.1 Dataset details

Out of the 25 scenes, containing almost 30k registered images in total, we select 2 for validation and 9 for testing. The remaining scenes can be used for training, if so desired, and are not used in this Chapter. We provide images, 3D reconstructions, camera poses, and depth maps, for every training and validation scene. For the test scenes we release only a subset of 100 images and keep the ground truth private, which is an integral part of the Image Matching Challenge. Results on the private test set can be obtained by sending submissions to the organizers, who process them.

The scenes used for training, validation and test are listed in Table 6.1, along with the acronyms used in several figures. For the validation experiments – Sections 6.5 and 6.7 – we choose two of the larger scenes, “Sacre Coeur” and “St. Peter’s Square”, which in our experience provide results that are quite representative of what should be expected on the Image Matching Challenge Dataset. These two subsets have been released, so that the validation results are reproducible and comparable. They can all be downloaded from the challenge website².

6.3.2 Estimating the co-visibility between two images

When evaluating image pairs, one has to be sure that two given images share a common part of the scene – they may be registered by the SfM reconstruction without having any pixels in common as long as other images act as a ‘bridge’ between them. Co-visible pairs of images are determined with a simple heuristic. 3D model points, detected in both the considered images, are localised in 2D in each image. The bounding box around the keypoints is estimated. The ratio of the bounding box area and the whole image is the “visibility” of image i in j and j in i respectively; see Fig. 6.5 for examples. The minimum of the two “visibilities” is the “co-visibility” ratio $v_{i,j} \in [0, 1]$, which characterizes how challenging matching of the particular image pair is. The co-visibility varies significantly from scene to scene. The histogram of co-visibilitys is

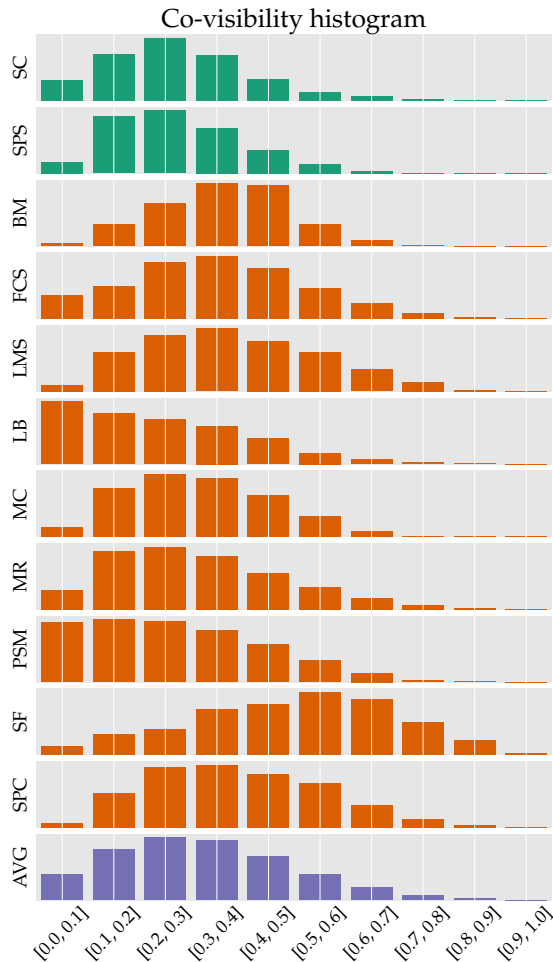


Figure 6.4: **Co-visibility histogram**. We break down the co-visibility measure for each scene in the validation (green) and test (red) sets, as well as the average (purple). Notice how the statistics may vary significantly from scene to scene.

shown in Fig. 6.4, providing insights into how “hard” each scene is – without accounting for some occlusions.

For stereo, the minimum co-visibility threshold is set to 0.1. For the multi-view task, subsets where at least 100 3D points are visible in each image, are selected, as in [284, 290]. We find that both criteria work well in practice.

6.3.3 On the quality of the “ground-truth”

Our core assumption is that accurate poses can be obtained from large sets of images without human intervention. Such poses are used as the “ground truth” for evaluation of image matching performance on pairs or small subsets of images – a harder, proxy task. Should this assumption hold, the (relative) poses retrieved with a large enough number of images would not change as more images are added, and these poses would be the same regardless of which local feature is used. To validate this, we pick the scene “Sacre Coeur” and compute SfM reconstructions with a varying number of images: 100, 200, 400, 800, and 1179 images (the entire “Sacre Coeur” dataset), where each set contains the previous one; new images are being added and no images are removed. We run each reconstruction three times, and report the average result of the three runs, to account for the variance inside COLMAP. The standard deviation among different runs is reported in Table 6.2. Note that these reconstructions are only defined up to a scale factor – we do not know the absolute scale that could be used to compare the reconstructions against each other. That is why we use a simple, pairwise metric instead. We pick all the pairs out of the 100 images present in the smallest subset, and compare how much their relative pose change with

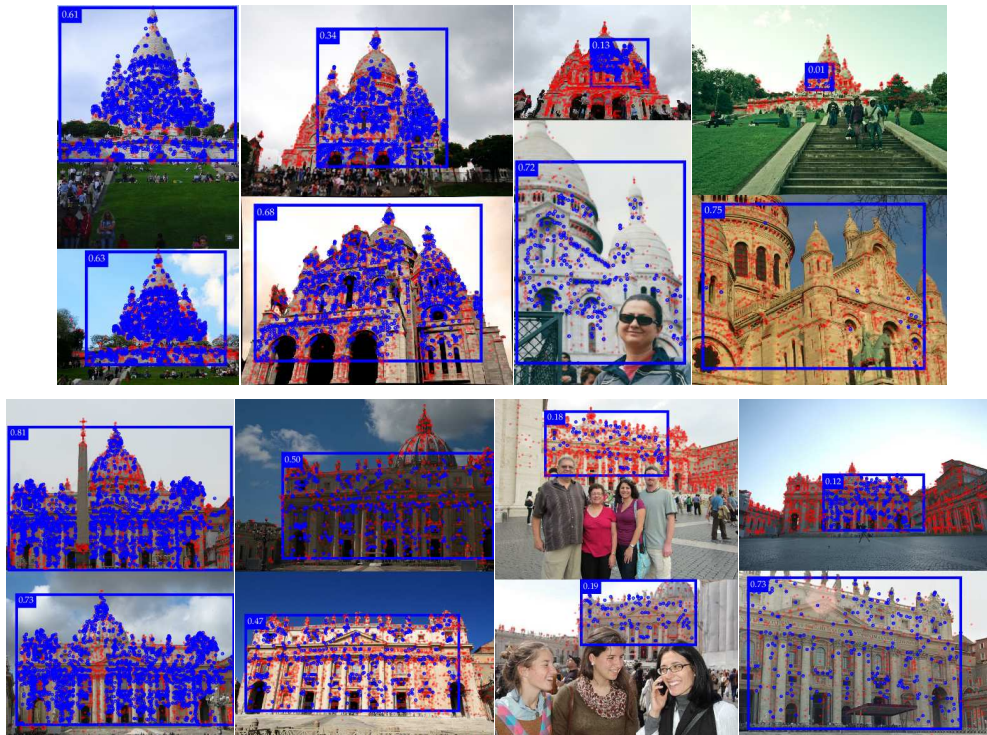


Figure 6.5: **Co-visibility examples** – image pairs from validation scenes “Sacre Coeur” (top) and “St. Peter’s Square” (bottom). Image keypoints that are part of the 3D reconstruction are blue if they are co-visible in both images, are red otherwise. The bounding box of the co-visible points, used to compute a per-image co-visibility ratio, is shown in blue. The co-visibility value for the image pair is the lower of these two values. Examples include different ‘difficulty’ levels. All of the pairs are used in the evaluation except the top-right one, as we set a cut-off at 0.1.

| Local featured type | Number of Images | | | | |
|-------------------------|------------------|-------|-------|-------|-------|
| | 100 | 200 | 400 | 800 | all |
| SIFT [140] | 0.06° | 0.09° | 0.06° | 0.07° | 0.09° |
| SIFT (Upright) [140] | 0.07° | 0.07° | 0.04° | 0.06° | 0.09° |
| HardNet (Upright) [158] | 0.06° | 0.06° | 0.06° | 0.04° | 0.05° |
| SuperPoint [62] | 0.31° | 0.25° | 0.33° | 0.19° | 0.32° |
| R2D2 [207] | 0.12° | 0.08° | 0.07° | 0.08° | 0.05° |

Table 6.2: **Standard deviation of the pose difference of three COLMAP runs with different number of images.** Most of them are below 0.1°, except for SuperPoint.

| Local feature type | Number of images | | | |
|-------------------------|------------------|---------------|---------------|---------------|
| | 100 vs. all | 200 vs. all | 400 vs. all | 800 vs. all |
| SIFT [140] | 0.58° / 0.22° | 0.31° / 0.08° | 0.23° / 0.05° | 0.18° / 0.04° |
| SIFT (Upright) [140] | 0.52° / 0.16° | 0.29° / 0.08° | 0.22° / 0.05° | 0.16° / 0.03° |
| HardNet (Upright) [158] | 0.35° / 0.10° | 0.33° / 0.08° | 0.23° / 0.06° | 0.14° / 0.04° |
| SuperPoint [62] | 1.22° / 0.71° | 1.11° / 0.67° | 1.08° / 0.48° | 0.74° / 0.38° |
| R2D2 [207] | 0.49° / 0.14° | 0.32° / 0.10° | 0.25° / 0.08° | 0.18° / 0.05° |

Table 6.3: **Pose convergence in SfM.** We report the mean/median of the difference (in degrees) between the poses extracted with the full set of 1179 images for “Sacre Coeur”, and different subsets of it, for four local feature methods – to keep the results comparable we only look at the 100 images in common across all subsets. We report the maximum among the angular difference between rotation matrices and translation vectors. The estimated poses are stable, with as little as 100 images.

| Reference | Compared to | | | |
|------------|----------------|-------------------|---------------|---------------|
| | SIFT (Upright) | HardNet (Upright) | SuperPoint | R2D2 |
| SIFT [140] | 0.20° / 0.05° | 0.26° / 0.05° | 1.01° / 0.62° | 0.26° / 0.09° |

Table 6.4: **Difference between poses obtained with different local features.** We report the mean/median of the difference (in degrees) between the poses extracted with SIFT (Upright), HardNet (Upright), SuperPoint, or R2D2, and those extracted with SIFT. We use the maximum of the angular difference between rotation matrices and translation vectors. SIFT (Upright), HardNet (Upright), and R2D2 give near-identical results to SIFT.

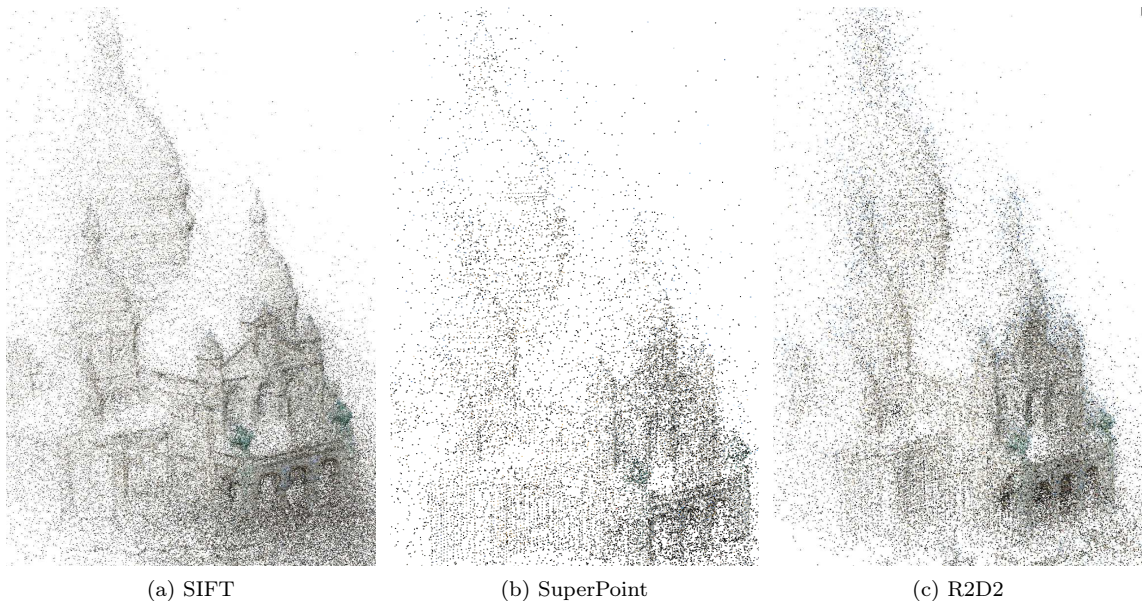


Figure 6.6: **COLMAP with different local features.** We show the reconstructed point cloud for the scene “Sacre Coeur” using three different local features: SIFT, SuperPoint, and R2D2, using all images available (1179). The reconstructions with SIFT and R2D2 are both dense, albeit somewhat different. The reconstruction with SuperPoint is quite dense, considering it can only extract a much smaller number of features effectively, but its poses appear less accurate.

respect to their counterparts reconstructed using the entire set – we do this for every subset, *i.e.*, 100, 200, etc. Ideally, we would like the differences between the relative poses to approach zero as more images are added. We list the results in Table 6.3, for different local feature methods. Notice how the poses converge, especially in terms of the median, as more images are used, for all methods – and that the reconstructions using only 100 images are already very stable. For SuperPoint we use a smaller number of features (2k per image), which is not enough to achieve pose convergence, but the error is still reduced as more images are used.

We conduct a second experiment to verify that there is no bias towards using SIFT features for obtaining the ground truth. We compare the poses obtained with SIFT to those obtained with other local features – note that our primary metric uses nothing but the *estimated poses* for evaluation. We report results in Table 6.4. We also show the histogram of pose differences in Fig. 6.7. The differences in pose due to the use of different local features have a median value below 0.1° . In fact, the pose variation of individual reconstructions with SuperPoint is of the same magnitude as the difference between reconstructions from SuperPoint and other local features: see Table 6.3. We conjecture that the reconstructions with SuperPoint, which does not extract a large number of keypoints, are less accurate and stable. This is further supported by the fact that the point cloud obtained with the entire scene generated with SuperPoint is less dense (125K 3D points) than the ones generated with SIFT (438K) or R2D2 (317k); see Fig. 6.6. In addition, we note that SuperPoint keypoints have been shown to be less accurate when it comes to precise alignment [62, Table 4, $\epsilon = 1$]. Note also that the poses from R2D2 are nearly identical to those from SIFT.

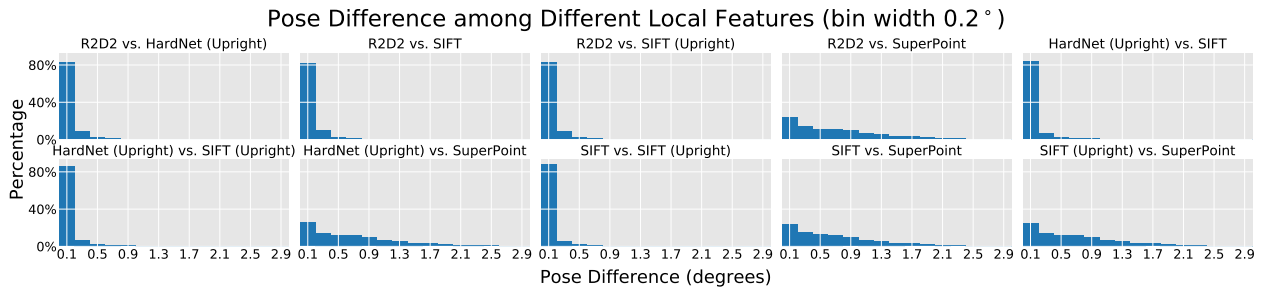


Figure 6.7: **Histograms of pose differences between reconstructions with different local feature methods.** We consider five different local features – including rotation-sensitive and upright SIFT – resulting in 10 combinations. The plots show that about 80% percent of image pairs are within a 0.2° pose difference, with the exception of those involving SuperPoint.

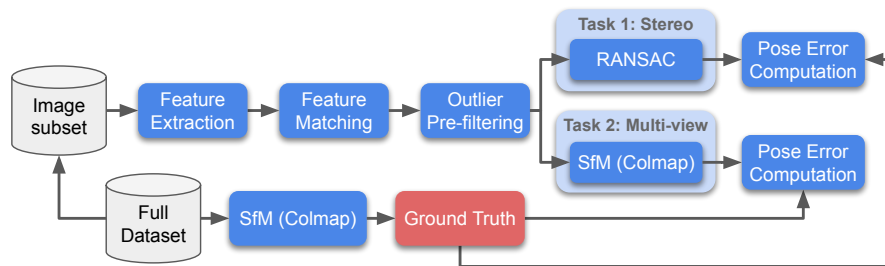


Figure 6.8: **The benchmark pipeline** takes a subset of N images of a scene as input, extracts features for each, and computes matches for all $M = \frac{1}{2}N(N-1)$ image pairs. After an optional filtering step, the matches are fed to two different tasks. Performance is measured *downstream*, by a pose-based metric, common across tasks. The ground truth is extracted once, on the full set of images.

These observations reinforce our trust in the accuracy of our ground truth – given a sufficient number of images, the choice of local feature is irrelevant, at least for the purpose of retrieving accurate poses. Our evaluation considers pose errors of up to 10° , at a resolution of 1° – significantly smaller than the fluctuations observed here, which we consider negligible. Note that these conclusions may not hold on large-scale SfM requiring loop closures, but our dataset contains landmarks, which do not suffer from this problem.

In addition, we note that the use of dense, ground truth depth from SfM, which is arguably less accurate than camera poses, has been verified by multiple parties, for training and evaluation, including: CNe [284], DFE [205], LF-Net [181], D2-Net [69], LogPolarDesc [70], OANet [291], and SuperGlue [216] among others, suggesting it is sufficiently accurate – several of these rely on the data used in our work.

As a final observation, while the poses are *stable*, they could still be *incorrect*. This can happen on highly symmetric structures: for instance, a tower with a square or circular cross section. In order to prevent such errors from creeping into our evaluation, we visually inspected all the images in our test set. Out of 900 of them, we found 4 misregistered samples, all of them from the same scene, “London Bridge”, which were removed from our data.

6.4 Pipeline

We outline our pipeline in Fig. 6.8. It takes as input $N=100$ images per scene. The feature extraction module computes up to K features from each image. The feature matching module generates a list of putative matches for each image pair, *i.e.* $\frac{1}{2}N(N-1) = 4950$ combinations. These matches can be optionally processed by an outlier pre-filtering module. They are then fed to two tasks: stereo, and multiview reconstruction with SfM. We now describe each of these components in detail.

6.4.1 Feature extraction

We consider three broad families of local features. The first includes full, “classical” pipelines, most of them handcrafted: SIFT [140] (and RootSIFT [14]), SURF [26], ORB [212], and AKAZE [7]. We also consider FREAK [6] descriptors with BRISK [132] keypoints. We take these from OpenCV. For all of them, except ORB, we lower the detection threshold to extract more features, which increases performance when operating with a large feature budget. We also consider DoG alternatives from VLFeat [273]: (VL-)DoG, Hessian [28], Hessian-Laplace [153], Harris-Laplace [153], MSER [148]; and their affine-covariant versions: DoG-Affine, Hessian-Affine [153, 25], DoG-AffNet [161], and Hessian-AffNet [161].

The second group includes descriptors learned on DoG keypoints: L2-Net [288], Hardnet [158], Geodesc [143], SOSNet [256], ContextDesc [142], and LogPolarDesc [70].

The last group consists of pipelines learned end-to-end (e2e): Superpoint [62], LF-Net [181], D2-Net [69] (with both single- (SS) and multi-scale (MS) variants), and R2D2 [207]. Additionally, we consider Key.Net [23], a learned detector paired with HardNet and SOSNet descriptors – we pair it with original implementation of HardNet instead than the one provided by the authors, as it performs better³.

Post-IJCV update. We have added 3 more local features to the benchmark after the official publication of the paper – DoG-AffNet-HardNet [158, 161], DoG-TFeat [18] and DoG-MKD-Concat [172]. We take them from the kornia [208] library. Results of this features are included into the Tables and Figures with a special mark *.

6.4.2 Feature matching

We break this step into four stages. Given images \mathbf{I}^i and \mathbf{I}^j , $i \neq j$, we create an initial set of matches by nearest neighbor (NN) matching from \mathbf{I}^i to \mathbf{I}^j , obtaining a set of matches $\mathbf{m}_{i \rightarrow j}$. We optionally do the same in the opposite direction, $\mathbf{m}_{j \rightarrow i}$. Lowe’s ratio test [140] is applied to each list to filter out non-discriminative matches, with a threshold $r \in [0, 1]$, creating “curated” lists $\tilde{\mathbf{m}}_{i \rightarrow j}$ and $\tilde{\mathbf{m}}_{j \rightarrow i}$. The final set of putative matches is the lists intersection, $\tilde{\mathbf{m}}_{i \rightarrow j} \cap \tilde{\mathbf{m}}_{j \rightarrow i} = \tilde{\mathbf{m}}_{i \leftrightarrow j}^\cap$ (known in the literature as one-to-one, mutual NN, bipartite, or cycle-consistent), or union $\tilde{\mathbf{m}}_{i \rightarrow j} \cup \tilde{\mathbf{m}}_{j \rightarrow i} = \tilde{\mathbf{m}}_{i \leftrightarrow j}^\cup$ (symmetric). We refer to them as “both” and “either”, respectively. We also implement a simple unidirectional matching, *i.e.*, $\tilde{\mathbf{m}}_{i \rightarrow j}$. Finally, the distance filter is optionally applied, removing matches whose distance is above a threshold.

The “both” strategy is similar to the “symmetrical nearest neighbor ratio” (sNNR) [31], proposed concurrently – SNNR combines the nearest neighbor ratio in both directions into a single number by taking the harmonic mean, while our test takes the maximum of the two values.

6.4.3 Outlier pre-filtering

Context Networks [284], or CNe for short, proposed a method to find sparse correspondences with a permutation-equivariant deep network based on PointNet [200], sparking a number of follow-up works [205, 59, 295, 290, 248]. We embed CNe into our framework. It often works best when paired with RANSAC [284, 248], so we consider it as an *optional* pre-filtering step before RANSAC – and apply it to both stereo and multiview. As the published model was trained on one of our validation scenes, we re-train it on “Notre Dame Front Facade” and “Buckingham Palace”, following CNe training protocol, *i.e.*, with 2000 SIFT features, unidirectional matching, and no ratio test. We evaluated the new model on the test set and observed that its performance is better than the one that was released by the authors. It could be further improved by using different matching schemes, such as bidirectional matching, but we have not explored this in this paper and leave as future work.

We perform one additional, but necessary, change: CNe (like most of its successors) was originally trained to estimate the Essential matrix instead of the Fundamental matrix [284], *i.e.*, it assumes known intrinsics. In order to use it within our setup, we normalize the coordinates by the size of the image instead of using ground truth calibration matrices. This strategy has also been used in [248], and has been shown to work well in practice.

³In [23] the models are converted to TensorFlow – we use the original PyTorch version.

6.4.4 Stereo task

The list of tentative matches is given to a robust estimator, which estimates $\mathbf{F}_{i,j}$, the Fundamental matrix between \mathbf{I}_i and \mathbf{I}_j . In addition to (locally-optimized) RANSAC [75, 50], as implemented in OpenCV [35], and sklearn [184], we consider more recent algorithms with publicly available implementations: DEGENSAC [51], GC-RANSAC [20] and MAGSAC [60]. We use the original GC-RANSAC implementation⁴, and not the most up-to-date version, which incorporates MAGSAC, DEGENSAC, and later changes.

For DEGENSAC we additionally consider disabling the degeneracy check, which theoretically should be equivalent to the OpenCV and sklearn implementations – we call this variant “PyRANSAC”. Given $\mathbf{F}_{i,j}$, the known intrinsics $\mathbf{K}_{\{i,j\}}$ were used to compute the Essential matrix $\mathbf{E}_{i,j}$, as $\mathbf{E}_{i,j} = \mathbf{K}_j^T \mathbf{F}_{i,j} \mathbf{K}_i$. Finally, the relative rotation and translation vectors were recovered with a chirality check with OpenCV’s `recoverPose`.

6.4.5 Multiview task

Large-scale SfM is notoriously hard to evaluate, as it requires accurate ground truth. Since our goal is to benchmark *local features* and *matching methods*, and not SfM algorithms, we opt for a different strategy. We reconstruct a scene from small image subsets, which we call “bags”. We consider bags of 5, 10, and 25 images, which are randomly sampled from the original set of 100 images per scene – with a co-visibility check. We create 100 bags for bag sizes 5, 50 for bag size 10, and 25 for bag size 25 – *i.e.*, 175 SfM runs in total.

We use COLMAP [226], feeding it the matches computed by the previous module – note that this comes before the robust estimation step, as COLMAP implements its own RANSAC. If multiple reconstructions are obtained, we consider the largest one. We also collect and report statistics such as the number of landmarks or the average track length. Both statistics and error metrics are averaged over the three bag sizes, each of which is in turn averaged over its individual bags.

6.4.6 Error metrics

Since the stereo problem is defined up to a scale factor [90], our main error metric is based on *angular errors*. We compute the difference, in degrees, between the *estimated* and *ground-truth* translation and rotation *vectors* between two cameras. We then threshold it over a given value for all possible – *i.e.*, co-visible – pairs of images. Doing so over different angular thresholds renders a curve. We compute the mean Average Accuracy (mAA) by integrating this curve up to a maximum threshold, which we set to 10° – this is necessary because large errors always indicate a bad pose: 30° is not necessarily better than 180° , both estimates are wrong. Note that by computing the area under the curve we are giving more weight to methods which are more accurate at lower error thresholds, compared to using a single value at a certain designated threshold.

This metric was originally introduced in [284], where it was called mean Average Precision (mAP). We argue that “accuracy” is the correct terminology, since we are simply evaluating how many of the predicted poses are “correct”, as determined by thresholding over a given value – *i.e.*, our problem does not have “false positives”.

The same metric is used for multiview. Because we do not know the scale of the scene *a priori*, it is not possible to measure the translation error in metric terms. While we intend to explore this in the future, such a metric, while more interpretable, is not without problems – for instance, the range of the distance between the camera and the scene can vary drastically from scene to scene and make it difficult to compare their results. To compute the mAA in pose estimation for the multiview task, we take the mean of the average accuracy for every pair of cameras – setting the pose error to ∞ for pairs containing unregistered views. If COLMAP returns multiple models which cannot be co-registered (which is rare) we consider only the largest of them for simplicity.

For the stereo task, we can report this value for different co-visibility thresholds: we use $v = 0.1$ by default, which preserves most of the “hard” pairs. Note that this is not applicable to the multiview task, as all images are registered at once via bundle adjustment in SfM.

Finally, we consider repeatability and matching score. Since many end-to-end methods do not report and often do not have a clear measure of scale – or support region – we simply threshold

⁴<https://github.com/danini/graph-cut-ransac/tree/benchmark-version>

by pixel distance, as in [211]. For the multiview task, we also compute the Absolute Trajectory Error (ATE) [244], a metric widely used in SLAM. Since, once again, the reconstructed model is scale-agnostic, we first scale the reconstructed model to that of the ground truth and then compute the ATE. Note that ATE needs a minimum of three points to align the two models.

6.4.7 Implementation

The benchmark code has been open-sourced¹ along with every method used in the paper⁵. The implementation relies on SLURM [287] for scalable job scheduling, which is compatible with our supercomputer clusters – we also provide on-the-cloud, ready-to-go images⁶. The benchmark can also run on a standard computer, sequentially. It is computationally expensive, as it requires matching about 45k image pairs. The most costly step – leaving aside feature extraction, which is very method-dependent – is typically feature matching: 2–6 seconds per image pair⁷, depending on the descriptor size. Outlier pre-filtering takes about 0.5–0.8 seconds per pair, excluding some overhead to reformat the data into its expected format. RANSAC methods vary between 0.5–1 second – as explained in Section 6.5 we limit their number of iterations based on a compute budget, but the actual cost depends on the number of matches. Note that these values are computed on the validation set – for the test set experiments we increase the RANSAC budget, in order to remain compatible with the rules of the Image Matching Challenge². We find COLMAP to vary drastically between set-ups. New methods will be continuously added, and we welcome contributions to the code base.

6.5 Details are Important

Our experiments indicate that each method needs to be carefully tuned. In this section we outline the methodology we used to find the right hyperparameters on the validation set, and demonstrate why it is crucial to do so.

6.5.1 RANSAC: Leveling the field

Robust estimators are, in our experience, the most sensitive part of the stereo pipeline, and thus the one we first turn to. All methods considered in this Section have three parameters in common: the confidence level in their estimates, τ ; the outlier (epipolar) threshold, η ; and the maximum number of iterations, Γ . We find the confidence value to be the least sensitive, so we set it to $\tau = 0.999999$.

We evaluate each method with different values for Γ and η , using reasonable defaults: 8k SIFT features with bidirectional matching with the “both” strategy and a ratio test threshold of 0.8. We plot the results in Fig. 6.9, against their computational cost – for the sake of clarity we only show the curve corresponding to the best reprojection threshold η for each method.

Our aim with this experiment is to place all methods on an “even ground” by setting a common budget, as we need to find a way to compare them. We pick 0.5 seconds, where all methods have mostly converged. Note that these are different implementations and are obviously not directly comparable to each other, but this is a simple and reasonable approach. We set this budget by choosing Γ as per Fig. 6.9, instead of actually *enforcing* a time limit, which would not be comparable across different set-ups. Optimal values for Γ can vary drastically, from 10k for MAGSAC to 250k for PyRANSAC. MAGSAC gives the best results for this experiment, closely followed by DEGENSAC. We patch OpenCV to increase the limit of iterations, which was hardcoded to $\Gamma = 1000$; this patch is now integrated into OpenCV. This increases performance by 10-15% relative, within our budget. However, PyRANSAC is significantly better than OpenCV version even with this patch, so we use it as our “vanilla” RANSAC instead. The sklearn implementation is too slow for practical use.

We find that, in general, default settings can be woefully inadequate. For example, OpenCV recommends $\tau = 0.99$ and $\eta = 3$ pixels, which results in a mAA at 10° of 0.3642 on the validation set – a performance drop of 29.3% relative.

⁵<https://github.com/vcg-uvic/image-matching-benchmark-baselines>

⁶<https://github.com/etrulls/slurm-gcp>

⁷Time measured on ‘n1-standard-2’ VMs on Google Cloud Compute: 2 vCPUs with 7.5 GB of RAM and no GPU.

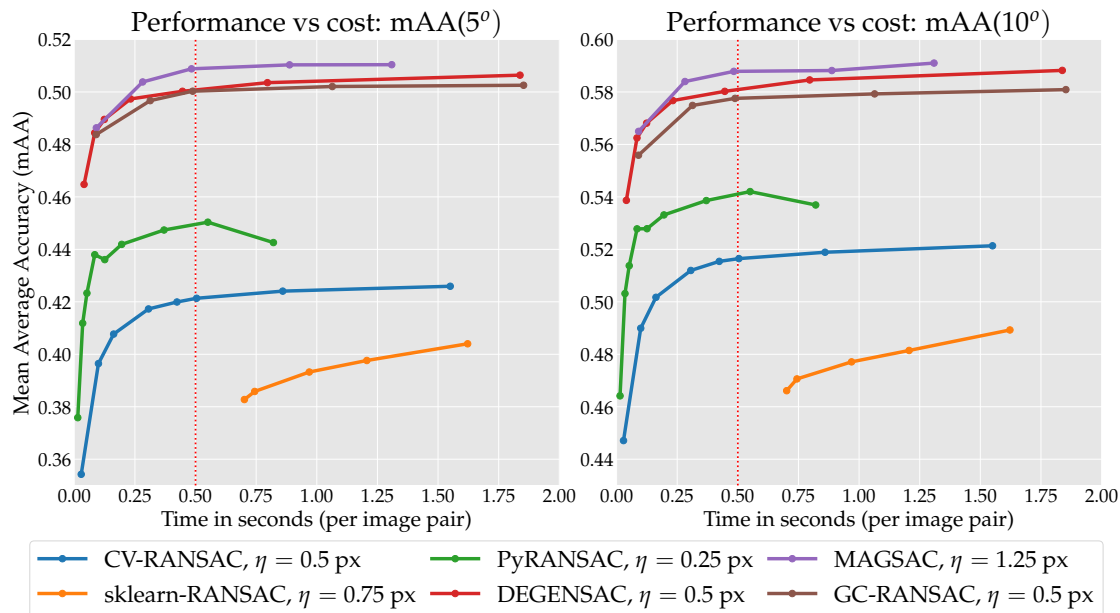


Figure 6.9: **Validation – Performance vs. cost for RANSAC.** We evaluate six RANSAC variants, using 8k SIFT features with “both” matching and a ratio test threshold of $r=0.8$. The inlier threshold η and iterations limit Γ are variables – we plot only the best η for each method, for clarity, and set a budget of 0.5 seconds per image pair (dotted red line). For each RANSAC variant, we pick the largest Γ under this time “limit” and use it for all validation experiments. Computed on ‘n1-standard-2’ VMs on Google Compute (2 vCPUs, 7.5 GB).

6.5.2 RANSAC: One method at a time

The last free parameter is the inlier threshold η . We expect the optimal value for this parameter to be different for each local feature, with looser thresholds required for methods operating on higher recall/lower precision, and end-to-end methods trained on lower resolutions.

We report a wide array of experiments in Fig. 6.10 that confirm our intuition: descriptors learned on DoG keypoints are clustered, while others vary significantly. Optimal values are also different for each RANSAC variant. We use the ratio test with the threshold recommended by the authors of each feature, or a reasonable value if no recommendation exists, and the “both” matching strategy – this cuts down on the number of outliers.

6.5.3 Ratio test: One feature at a time

Having “frozen” RANSAC, we turn to the feature matcher – note that it comes *before* RANSAC, but it cannot be evaluated in isolation. We select PyRANSAC as a “baseline” RANSAC and evaluate different ratio test thresholds, separately for the stereo and multiview tasks. For this experiment, we use 8k features with all methods, except for those which cannot work on this regime – SuperPoint and LF-Net. This choice will be substantiated in Section 6.5.4. We report the results for bidirectional matching with the “both” strategy in Fig. 6.11, and with the “either” strategy in Fig. 6.12. We find that “both” – the method we have used so far – performs best overall. Bidirectional matching with the “either” strategy produces many (false) matches, increasing the computational cost in the estimator, and requires very small ratio test thresholds – as low as $r=0.65$. Our experiments with unidirectional matching indicate that is slightly worse, and it depends on the order of the images, so we did not explore it further.

As expected, each feature requires different settings, as the distribution of their descriptors is different. We also observe that optimal values vary significantly between stereo and multiview, even though one might expect that bundle adjustment should be able to better deal with outliers. We suspect that this indicates that there is room for improvement in COLMAP’s implementation of RANSAC.

Note how the ratio test is critical for performance, and one could arbitrarily select a threshold that favours one method over another, which shows the importance of proper benchmarking.

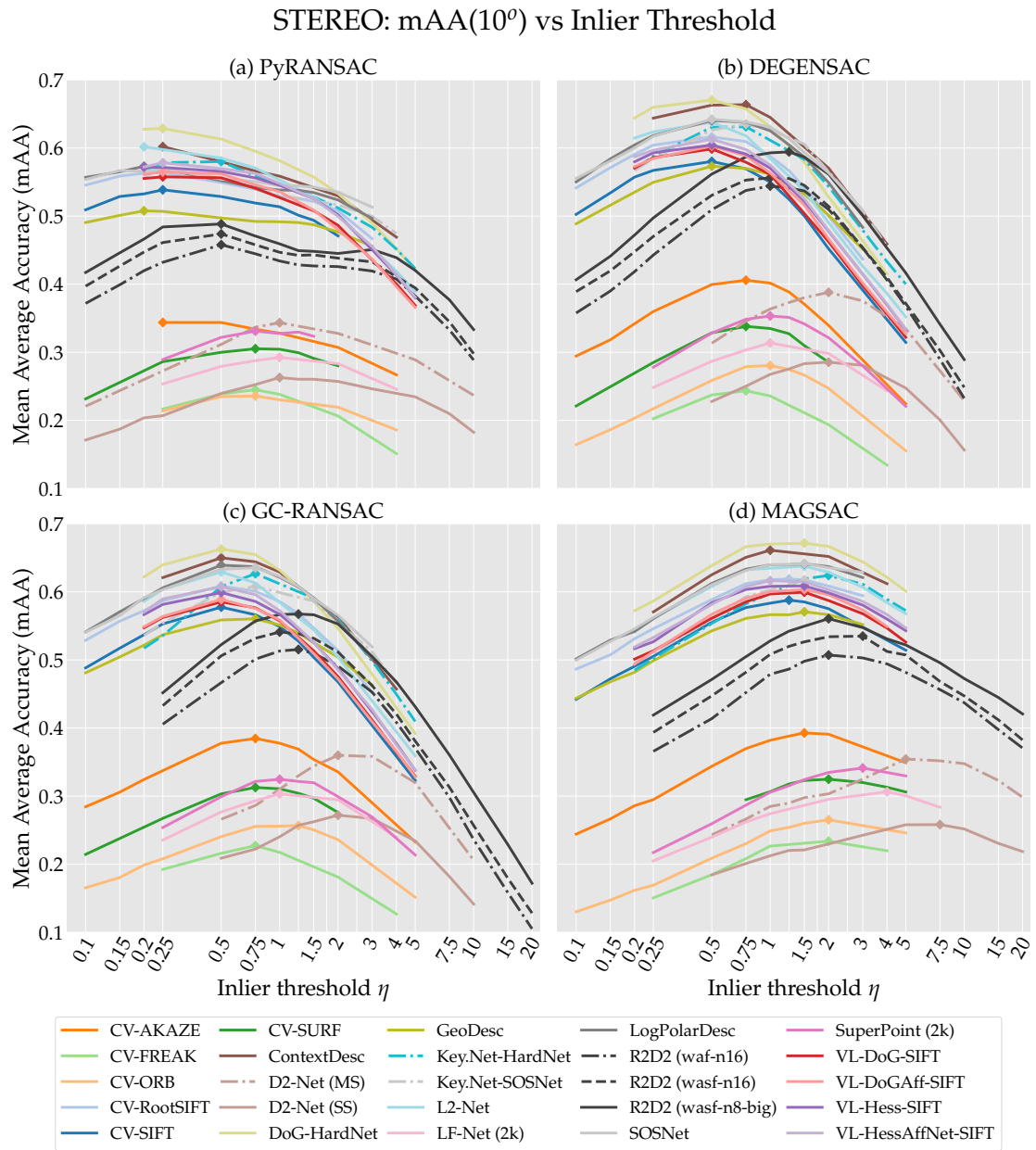


Figure 6.10: **Validation – Inlier threshold for RANSAC, η .** We determine η for each combination, using 8k features (2k for LF-Net and SuperPoint) with the “both” matching strategy and a reasonable value for the ratio test. Optimal parameters (diamonds) are listed in the Section 6.7.

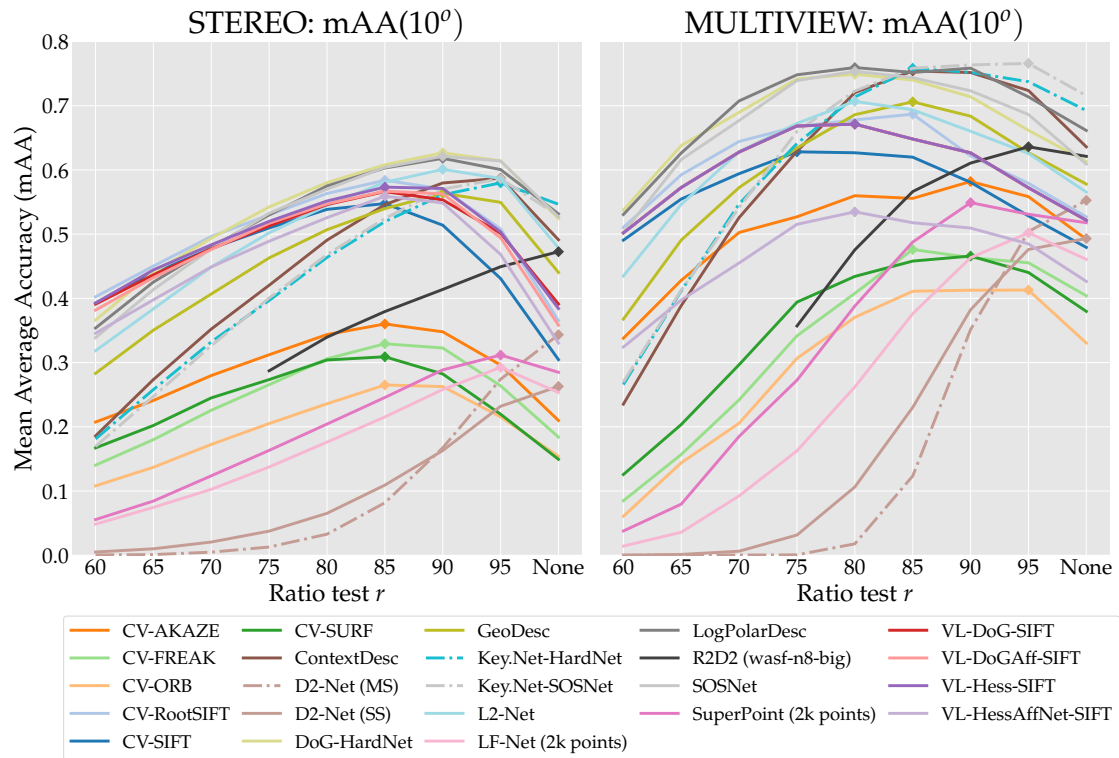


Figure 6.11: **Validation – Optimal ratio test r for matching with “both”.** We evaluate bidirectional matching with the “both” strategy (the best one), and different ratio test thresholds r , for each feature type. We use 8k features (2k for SuperPoint and LF-Net). For stereo, we use PyRANSAC.

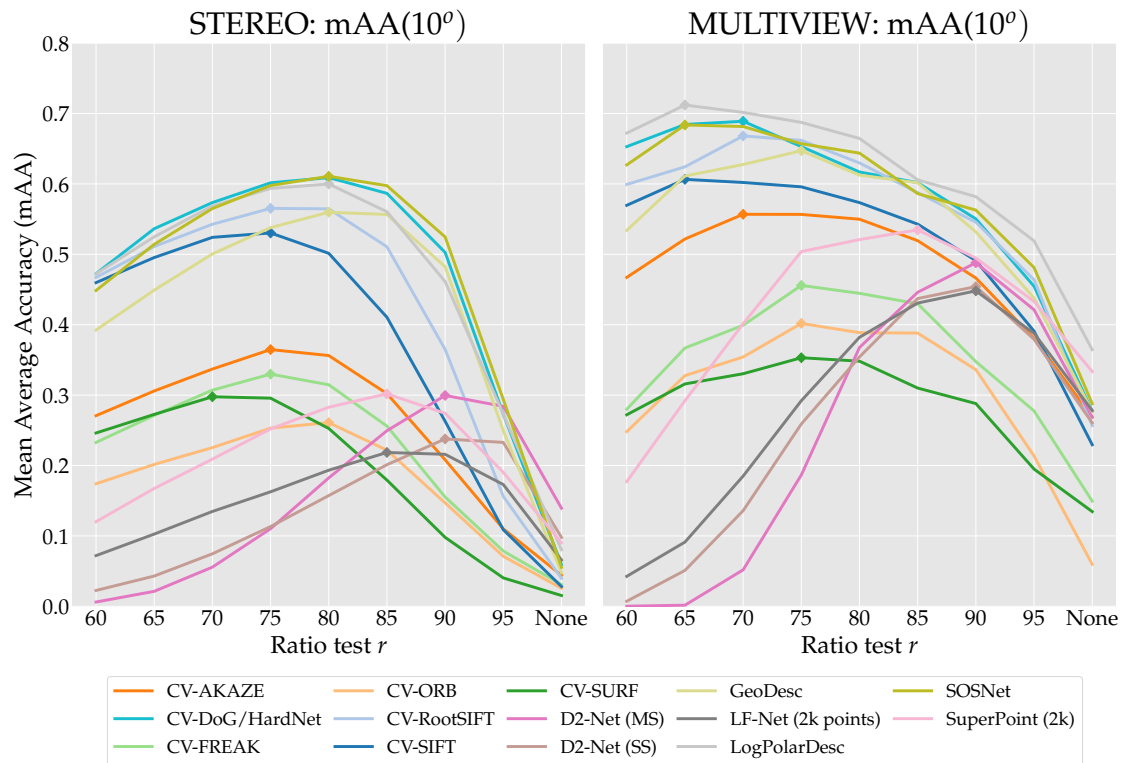


Figure 6.12: **Validation – Optimal ratio test r for matching with “either”.** Equivalent to Fig. 6.11 but with the “either” matching strategy. This strategy requires aggressive filtering and does not reach the performance of “both”, we thus explore only a subset of the methods.

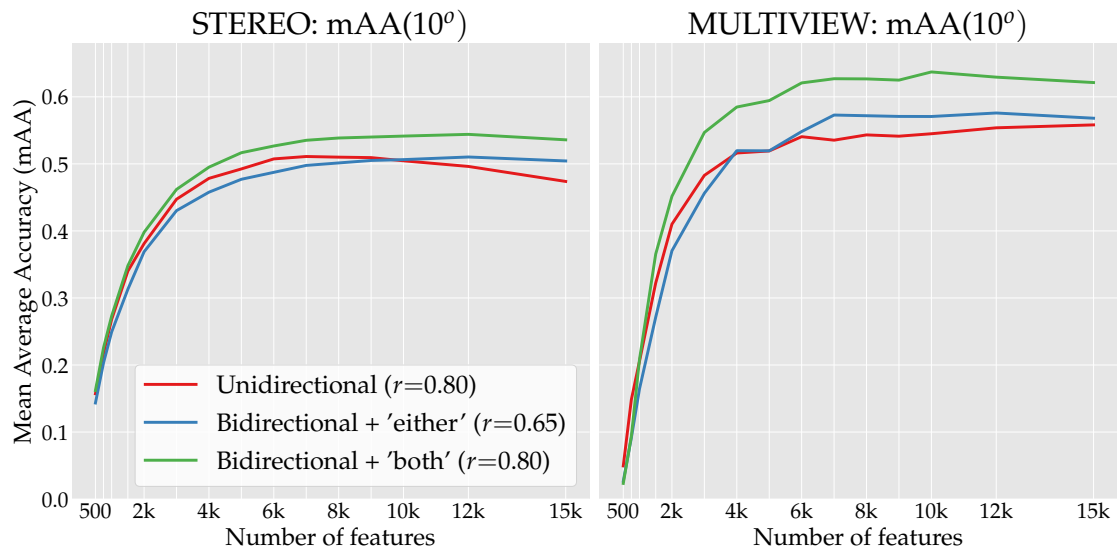


Figure 6.13: **Validation – Number of features.** Performance on the stereo and multi-view tasks while varying the number of SIFT features, with three matching strategies, and reasonable defaults for the ratio test r .

Interestingly, D2-Net is the *only* method that clearly performs best without the ratio test. It also performs poorly overall in our evaluation, despite reporting state-of-the-art results in other benchmarks [151, 16, 219, 250] – without the ratio test, the number of tentative matches might be too high for RANSAC or COLMAP to perform well.

Additionally, we implement the first-geometric-inconsistent ratio threshold, or FGINN [162]. We find that although it improves over unidirectional matching, its gains mostly disappear against matching with “both”. We report these results in Section 6.7.2.

6.5.4 Choosing the number of features

The ablation tests in this section use (up to) $K=8000$ feature (2k for SuperPoint and LF-Net, as they are trained to extract fewer keypoints). This number is commensurate with that used by SfM frameworks [281, 226]. We report performance for different values of K in Fig. 6.13. We use PyRANSAC with reasonable defaults for all three matching strategies, with SIFT features.

As expected, performance is strongly correlated with the number of features. We find 8k to be a good compromise between performance and cost, and also consider 2k (actually 2048) as a ‘cheaper’ alternative – this also provides a fair comparison with some learned methods which only operate on that regime. We choose these two values as valid categories for the open challenge² linked to the benchmark, and do the same here for consistency.

6.5.5 Additional experiments

Some methods require additional considerations before evaluating them on the test set. We briefly discuss them in this section. Further experiments are available in Section 6.7.

Binary features (Fig. 6.14). We consider three binary descriptors: ORB [212], AKAZE [7], and FREAK [6]. Binary descriptor papers historically favour a distance threshold in place of the ratio test to reject non-discriminative matches [212], although some papers have used the ratio test for ORB descriptors [9]. We evaluate both in Fig. 6.14 – as before, we use up to 8k features and matching with the “both” strategy. The ratio test works better for all three methods – we use it instead of a distance threshold for all experiments in the Chapter, including those in the previous sections.

On the influence of the detector (Fig. 6.15). We embed several popular blob and corner detectors into our pipeline, with OpenCV’s DoG [140] as a baseline. We combine multiple methods, taking advantage of the VLFeat library: Difference of Gaussians (DoG), Hessian [28], HessianLaplace [153], HarrisLaplace [153], MSER [148], DoGAffine, Hessian-Affine [153, 25],

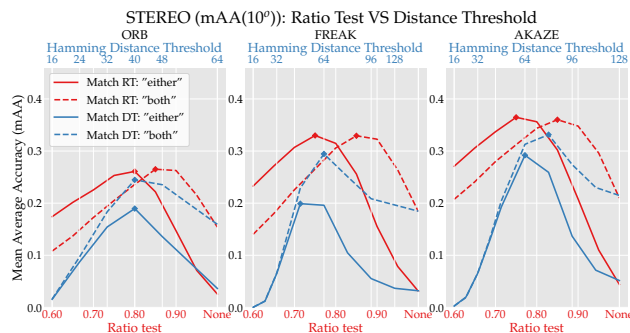


Figure 6.14: **Validation – Matching binary descriptors.** We filter out non-discriminative matches with the ratio test or a distance threshold. The latter (the standard) performs worse in our experiments.

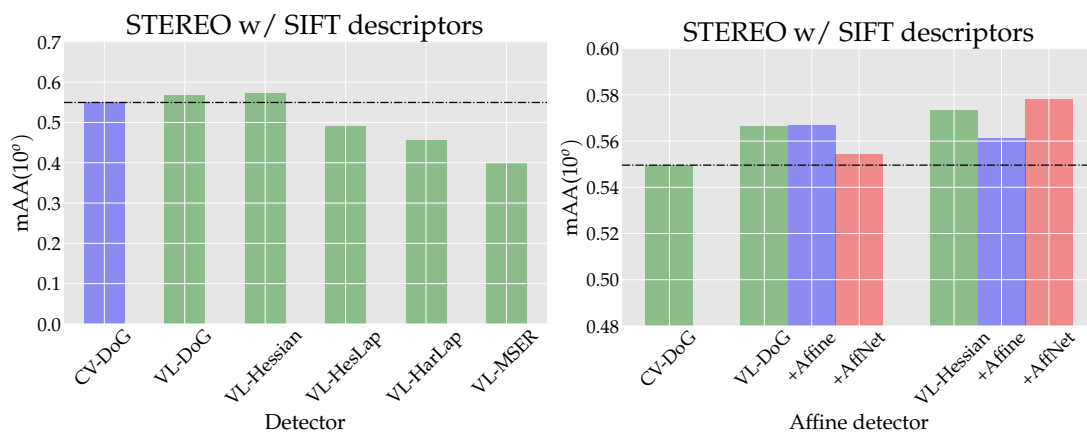


Figure 6.15: **Validation – Benchmarking detectors.** We evaluate the performance on the stereo task while pairing different detectors with SIFT descriptors. The dashed, black line indicates OpenCV SIFT – the baseline. **Left:** OpenCV DoG vs. VLFeat implementations of blob detectors (DoG, Hessian, HesLap) and corner detectors (Harris, HarLap), and MSER. **Right:** Affine shape estimation for DoG and Hessian keypoints, against the plain version. We consider a classical approach, Baumberg (Affine) [25], and the recent, learned AffNet [161] – they provide a small but inconsistent boost.

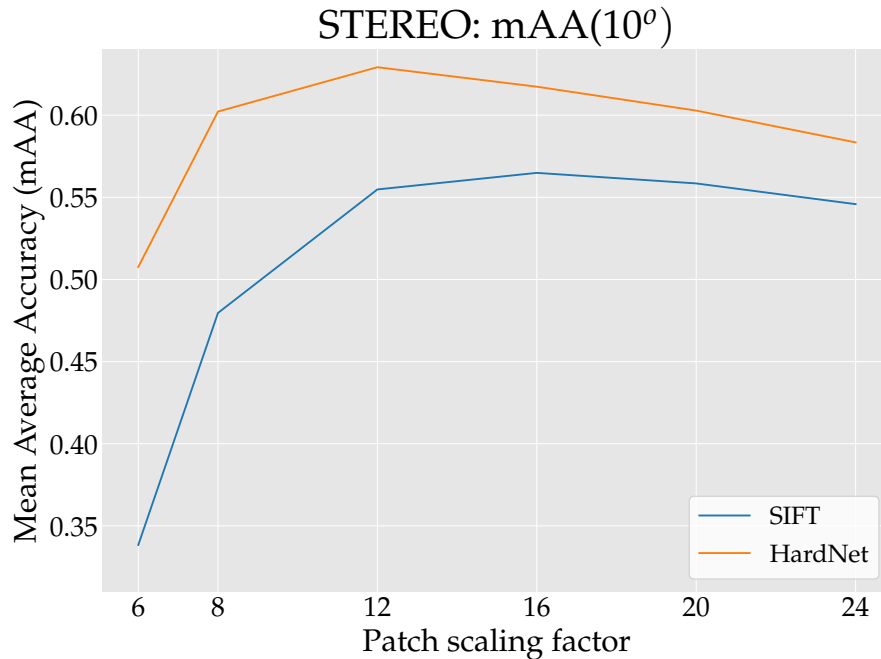


Figure 6.16: **Validation – Scaling the descriptor support region.** Performance with SIFT and HardNet descriptors while applying a scaling factor λ to the keypoint scale (note that OpenCV’s default value is $\lambda=12$). We consider SIFT and HardNet. Default values are optimal or near-optimal.

DoG-AffNet [161], and Hessian-AffNet [161]. We pair them with SIFT descriptors, also computed with VLFeat, as OpenCV cannot process affine keypoints, and report the results in Fig. 6.15. VLFeat’s DoG performs marginally better than OpenCV’s. Its affine version gives a small boost. Given the small gain and the infrastructure burden of interacting with a Matlab/C library, we use OpenCV’s DoG implementation for most of this Chapter.

On increasing the support region (Fig. 6.16). The size (“scale”) of the support region used to compute a descriptor can significantly affect its performance [65, 289, 2]. We experiment with different scaling factors, using DoG with SIFT and HardNet [158], and find that $12\times$ the OpenCV scale (the default value) is already nearly optimal, confirming the findings reported in [70]. We show these results in Fig. 6.16. Interestingly, SIFT descriptors do benefit from increasing the scaling factor from 12 to 16, but the difference is very small – we thus use the recommended value of 12 for the rest of the experiments. This, however, suggests that deep descriptors such as HardNet might be able to increase performance slightly by training on larger patches.

6.6 Establishing the State of the Art

With the findings and the optimal parameters found in Section 6.5, we move on to the test set, evaluating many methods with their optimal settings. All experiments in this section use bidirectional matching with the “both” strategy. We consider a large feature budget (up to 8k features) and a smaller one (up to 2k), and evaluate many detector/descriptor combinations.

We make three changes with respect to the validation experiments of the previous section. (1) We double the RANSAC budget from 0.5 seconds (used for validation) to 1 second per image pair, and adjust the maximum number of iterations Γ accordingly – we made this decision to encourage participants to the challenge based on this benchmark to use built-in methods rather than run RANSAC themselves to squeeze out a little extra performance, and use the same values for consistency. (2) We run each stereo and multiview evaluation three times and average the results, in order to decrease the potential randomness in the results – in general, we found the variations within these three runs to be negligible. (3) We use brute-force to match descriptors instead of FLANN, as we observed a drop in performance. For more details, see Section 6.6.6 and Table 6.12.

| Method | PyRANSAC | | | DEGENSAC | | MAGSAC | | Rank |
|--------------------------|----------|-----------------|------------------------------------|-----------------|------------------------------------|-----------------|------------------------------------|------|
| | NF | NI [†] | mAA(10 ^o) [†] | NI [†] | mAA(10 ^o) [†] | NI [†] | mAA(10 ^o) [†] | |
| CV-SIFT | 7861.1 | 167.6 | .3996 | 243.6 | .4584 | 297.4 | .4583 | 14 |
| VL-SIFT | 7880.6 | 179.7 | .3999 | 261.6 | .4655 | 326.2 | .4633 | 13 |
| VL-Hessian-SIFT | 8000.0 | 204.4 | .3695 | 290.2 | .4450 | 348.9 | .4335 | 15 |
| VL-DoGAff-SIFT | 7892.1 | 171.6 | .3984 | 250.1 | .4680 | 317.1 | .4666 | 11 |
| VL-HesAffNet-SIFT | 8000.0 | 209.3 | .3933 | 299.0 | .4679 | 350.0 | .4626 | 12 |
| CV- $\sqrt{\text{SIFT}}$ | 7860.8 | 192.3 | .4228 | 281.7 | .4930 | 347.5 | .4941 | 10 |
| CV-SURF | 7730.0 | 107.9 | .2280 | 113.6 | .2593 | 145.3 | .2552 | 19 |
| CV-AKAZE | 7857.1 | 131.4 | .2570 | 246.8 | .3074 | 301.8 | .3036 | 17 |
| CV-ORB | 7150.2 | 123.7 | .1220 | 150.0 | .1674 | 178.9 | .1570 | 22 |
| CV-FREAK | 8000.0 | 123.3 | .2273 | 131.0 | .2711 | 196.7 | .2656 | 18 |
| L2-Net | 7861.1 | 213.8 | .4621 | 366.0 | .5295 | 481.0 | .5252 | 5 |
| DoG-HardNet | 7861.1 | 286.5 | .4801 | 432.3 | .5543 | 575.1 | .5502 | 2 |
| DoG-HardNetAmos+ | 7861.0 | 265.7 | .4607 | 398.6 | .5385 | 528.7 | .5329 | 3 |
| Key.Net-HardNet | 7997.6 | 448.1 | .3997 | 598.3 | .4986 | 815.4 | .4739 | 9 |
| Key.Net-SOSNet | 7997.6 | 275.5 | .4236 | 587.4 | .5019 | 766.4 | .4780 | 8 |
| GeoDesc | 7861.1 | 205.4 | .4328 | 348.5 | .5111 | 453.4 | .5056 | 7 |
| ContextDesc | 7859.0 | 278.2 | .4684 | 493.6 | .5098 | 544.1 | .5143 | 6 |
| DoG-SOSNet | 7861.1 | 281.6 | .4784 | 424.6 | .5587 | 563.3 | .5517 | 1 |
| LogPolarDesc | 7861.1 | 254.4 | .4574 | 441.8 | .5340 | 591.2 | .5238 | 4 |
| D2-Net (SS) | 5665.3 | 280.8 | .1933 | 482.3 | .2228 | 781.3 | .2032 | 21 |
| D2-Net (MS) | 6924.1 | 278.2 | .2160 | 470.6 | .2506 | 741.2 | .2321 | 20 |
| R2D2 (wasf-n8-big) | 7940.5 | 457.6 | .3683 | 842.2 | .4437 | 998.9 | .4236 | 16 |
| DoG-AffNet-HardNet | 7834.0 | 267.9 | .4505 | 403.4 | .5447 | 516.8 | .5412 | 3* |
| DoG-MKD-Concat | 7860.8 | 208.0 | .4061 | 305.8 | .4846 | 381.4 | .4810 | 11* |
| DoG-TFeat | 7860.8 | 160.8 | .4008 | 234.8 | .4649 | 292.6 | .4668 | 13* |

Table 6.5: **Test – Stereo results with 8k features.** We report: (NF) Number of Features; (NI) Number of Inliers produced by RANSAC; and mAA(10^o). Top three methods by mAA marked in red, green and blue. * The last group of results is obtained after the paper publication and not described in the text of the paper. Their rank does not influence other entries ranks.

For stereo, we consider DEGENSAC and MAGSAC, which perform the best in the validation set, and PyRANSAC as a ‘baseline’ RANSAC. We report the results with both 8k features and 2k features in the following subsections. All observations are in terms of mAA, our primary metric, unless stated otherwise.

6.6.1 Results with 8k features — Tables 6.5 and 6.6

On the stereo task, deep descriptors extracted on DoG keypoints are at the top in terms of mAA, with SOSNet being #1, closely followed by HardNet. Interestingly, ‘HardNetAmos+’ [198], a version trained on more datasets – Brown [36], HPatches [16], and AMOS-patches [198] – performs worse than the original models, trained only on the “Liberty” scene from Brown’s dataset. On the multiview task, HardNet edges out ContextDesc, SOSNet and LogpolarDesc by a small margin. Affine features bring benefits in the multiview scenario, where AffNet improves DoG-HardNet results in terms of both mAA and success rate. However, the plain DoG detector is better for the stereo task. We also pair HardNet and SOSNet with Key.Net, a learned detector, which performs worse than with DoG when extracting a large number of features, with the exception of Key.Net + SOSNet on the multiview task.

R2D2, the best performing end-to-end method, does well on multiview (#7), but performs worse than SIFT on stereo – it produces a much larger number of “inliers” (which may be correct or incorrect) than most other methods. This suggests that, like D2-Net, its lack of compatibility with the ratio test may be a problem when paired with sample-based robust estimators, due to a lower inlier ratio. Note that D2-net performs poorly on our benchmark, despite state-of-the-art results on others. On the multiview task it creates many more 3D landmarks than any other

| Method | NL [↑] | SR [↑] | RC [↑] | TL [↑] | mAA(5°) [↑] | mAA(10°) [↑] | ATE [↓] | Rank |
|--------------------------|-----------------|-----------------|-----------------|-----------------|----------------------|-----------------------|------------------|------|
| CV-SIFT | 2577.6 | 96.7 | 94.1 | 3.95 | .5309 | .6261 | .4721 | 14 |
| VL-SIFT | 3030.7 | 97.9 | 95.4 | 4.17 | .5273 | .6283 | .4669 | 13 |
| VL-Hessian-SIFT | 3209.1 | 97.4 | 94.1 | 4.13 | .4857 | .5866 | .5175 | 16 |
| VL-DoGAff-SIFT | 3061.5 | 98.0 | 96.2 | 4.11 | .5263 | .6296 | .4751 | 12 |
| VL-HesAffNet-SIFT | 3327.7 | 97.7 | 95.2 | 4.08 | .5049 | .6069 | .4897 | 15 |
| CV- $\sqrt{\text{SIFT}}$ | 3312.1 | 98.5 | 96.6 | 4.13 | .5778 | .6765 | .4485 | 9 |
| CV-SURF | 2766.2 | 94.8 | 92.6 | 3.47 | .3897 | .4846 | .6251 | 18 |
| CV-AKAZE | 4475.9 | 99.0 | 95.4 | 3.88 | .4516 | .5553 | .5715 | 17 |
| CV-ORB | 3260.3 | 97.2 | 91.1 | 3.45 | .2697 | .3509 | .7377 | 22 |
| CV-FREAK | 2859.1 | 92.9 | 91.7 | 3.53 | .3735 | .4653 | .6229 | 20 |
| L2-Net | 3424.9 | 98.6 | 96.2 | 4.21 | .5661 | .6644 | .4482 | 10 |
| DoG-HardNet | 4001.4 | 99.5 | 97.7 | 4.34 | .6090 | .7096 | .4187 | 1 |
| DoG-HardNetAmos+ | 3550.6 | 98.8 | 96.9 | 4.28 | .5879 | .6888 | .4428 | 6 |
| Key.Net-HardNet | 3366.0 | 98.9 | 96.7 | 4.32 | .5391 | .6483 | .4622 | 11 |
| Key.Net-SOSNet | 5505.5 | 100.0 | 98.7 | 4.46 | .5989 | .7038 | .4286 | 2 |
| GeoDesc | 3839.0 | 99.1 | 97.2 | 4.26 | .5782 | .6803 | .4445 | 8 |
| ContextDesc | 3732.5 | 99.3 | 97.6 | 4.22 | .6036 | .7035 | .4228 | 3 |
| DoG-SOSNet | 3796.0 | 99.3 | 97.4 | 4.32 | .6032 | .7021 | .4226 | 4 |
| LogPolarDesc | 4054.6 | 99.0 | 96.4 | 4.32 | .5928 | .6928 | .4340 | 5 |
| D2-Net (SS) | 5893.8 | 99.8 | 97.5 | 3.62 | .3435 | .4598 | .6361 | 21 |
| D2-Net (MS) | 6759.3 | 99.7 | 98.2 | 3.39 | .3524 | .4751 | .6283 | 19 |
| R2D2 (wasf-n8-big) | 4432.9 | 99.7 | 97.2 | 4.59 | .5775 | .6832 | .4333 | 7 |
| DoG-AffNet-HardNet | 4671.3 | 99.9 | 98.1 | 4.56 | .6296 | .7267 | .4021 | 1* |
| DoG-MKD-Concat | 3507.4 | 98.5 | 96.1 | 4.17 | .5461 | .6476 | .4668 | 11* |
| DoG-TFeat | 2905.3 | 97.1 | 94.8 | 4.04 | .5270 | .6261 | .4873 | 14* |

Table 6.6: **Test – Multiview results with 8k features.** We report: **(NL)** Number of 3D Landmarks; **(SR)** Success Rate (%) in the 3D reconstruction across “bags”; **(RC)** Ratio of Cameras (%) registered in a “bag”; **(TL)** Track Length or number of observations per landmark; **mAA** at 5 and 10°; and **(ATE)** Absolute Trajectory Error. All metrics are averaged across different “bag” sizes, as explained in Section 6.4. We rank them by mAA at 10° and color-code them as in Table 6.5. * **The last group of results is obtained after the paper publication and not described in the text of the paper. Their rank does not influence other entries ranks.**

method. Both issues may be related to its poor localization (pixel) accuracy, due to operating on downsampled feature maps.

Out of the handcrafted methods, SIFT – RootSIFT specifically – remains competitive, being #10 on stereo and #9 on multiview, within 13.1% and 4.9% relative of the top performing method, respectively, while previous benchmarks report differences in performance of *orders of magnitude*. Other “classical” features do not fare so well. One interesting observation is that among these, their ranking on validation and test set is not consistent – Hessian is better on validation than DoG, but significantly worse on the test set, especially in the multiview setup. While this is a special case, this nonetheless demonstrates that a small-scale benchmark can be misleading, and a method needs to be tested on a variety of scenes, which is what our test set aims to provide.

Regarding the robust estimators, DEGENSAC and MAGSAC both perform very well, with the former edging out the latter for most local feature methods. This may be due to the nature of the scenes, which often contain dominant planes.

6.6.2 Results with 2k features — Tables 6.7 and 6.8

Results change slightly on the low-budget regime, where the top two spots on both tasks are occupied by Key.Net+SOSNet and DoG-AffNet-HardNet. They are closely followed by LogPolarDesc (#3 on stereo and #4 on multiview), a method trained on DoG keypoints – but using a much larger support region, resampled into log-polar patches. R2D2 performs very well on the multiview task (#3), while once again falling a bit short on the stereo task (#8, and 14.5% relative below the #1 method), for which it retrieves a number of inliers significantly larger than its competitors. The rest of the end-to-end methods do not perform so well, other than SuperPoint, which obtains competitive results on the multiview task.

The difference between classical and learned methods is more pronounced than with 8k points, with RootSIFT once again at the top, but now within 31.4% relative of the #1 method on stereo, and 26.9% on multiview. This is somewhat to be expected, given that with fewer keypoints, the quality of each individual point matters more.

6.6.3 2k features vs 8k features — Figs. 6.17 and 6.18

We compare the results between the low- and high-budget regimes in Fig. 6.17, for stereo (with DEGENSAC), and Fig. 6.18, for multiview. Note how methods can behave quite differently. Those based on DoG significantly benefit from an increased feature budget, whereas those learned end-to-end may require re-training – this is exemplified by the difference in performance between 2k and 8k for Key.Net+Hardnet, specially on multiview, which is very narrow despite quadrupling the budget. Overall, learned detectors – KeyNet, SuperPoint, R2D2, LF-Net – show relatively better results on multiview setup than on stereo. Our hypothesis is that they have good robustness, but low localization precision which is later corrected during bundle adjustment.

6.6.4 Outlier pre-filtering with deep networks — Table 6.9

Next, we study the performance of CNe [284] for outlier rejection, paired with PyRANSAC, DEGENSAC, and MAGSAC. Its training data does not use the ratio test, so we omit it here too – note that because of this, it expects a relatively large number of input matches. We thus evaluate it only for the 8k feature setting, while using the “both” matching strategy.

Our experiments with SIFT, the local feature used to train CNe, are encouraging: CNe aggressively filters out about 80% of the matches in a single forward pass, boosting mAA at 10° by 2-4% relative for stereo task and 8% for multiview task. In fact, it is surprising that nearly all classical methods benefit from it, with gains of up to 20% relative. By contrast, it damages performance with most learned descriptors, even those operating on DoG keypoints, and particularly for methods learned end-to-end, such as D2-Net and R2D2. We hypothesize this might be because the models performed better on the “classical” keypoints it was trained with – [248] reports that re-training them for a specific feature helps.

6.6.5 On the effect of local feature orientation estimation — Tables 6.10 and 6.11

In contrast with classical methods, which estimate the orientation of each keypoint, modern, end-to-end pipelines [62, 69, 207] often skip this step, assuming that the images are roughly

| Method | PyRANSAC | | | DEGENSAC | | MAGSAC | | Rank |
|--------------------------|----------|-----------------|-----------------------|-----------------|-----------------------|-----------------|-----------------------|------|
| | NF | NI [†] | mAA(10°) [†] | NI [†] | mAA(10°) [†] | NI [†] | mAA(10°) [†] | |
| CV-SIFT | 2048.0 | 84.9 | .2489 | 79.0 | .2875 | 99.2 | .2805 | 12 |
| CV- $\sqrt{\text{SIFT}}$ | 2048.0 | 84.2 | .2724 | 88.3 | .3149 | 106.8 | .3125 | 10 |
| CV-SURF | 2048.0 | 37.9 | .1725 | 72.7 | .2086 | 87.0 | .2081 | 15 |
| CV-AKAZE | 2048.0 | 96.1 | .1780 | 91.0 | .2144 | 115.5 | .2127 | 14 |
| CV-ORB | 2031.8 | 56.3 | .0610 | 63.5 | .0819 | 71.5 | .0765 | 19 |
| CV-FREAK | 2048.0 | 62.5 | .1461 | 65.6 | .1761 | 78.4 | .1698 | 17 |
| L2-Net | 1936.3 | 66.1 | .3131 | 92.4 | .3752 | 114.7 | .3691 | 6 |
| DoG-HardNet | 1936.3 | 111.9 | .3508 | 117.7 | .4029 | 150.5 | .4033 | 5 |
| Key.Net-HardNet | 2048.0 | 134.4 | .3272 | 174.8 | .4139 | 228.4 | .3897 | 1 |
| Key.Net-SOSNet | 2048.0 | 88.0 | .3279 | 171.9 | .4132 | 212.9 | .3928 | 2 |
| GeoDesc | 1936.3 | 98.9 | .3127 | 103.9 | .3662 | 129.7 | .3640 | 7 |
| ContextDesc | 2048.0 | 118.8 | .2965 | 124.1 | .3510 | 146.4 | .3485 | 9 |
| DoG-SOSNet | 1936.3 | 111.1 | .3536 | 132.1 | .3976 | 149.6 | .4092 | 4 |
| LogPolarDesc | 1936.3 | 118.8 | .3569 | 124.9 | .4115 | 161.0 | .4064 | 3 |
| D2-Net (SS) | 2045.6 | 107.6 | .1157 | 134.8 | .1355 | 259.3 | .1317 | 18 |
| D2-Net (MS) | 2038.2 | 149.3 | .1524 | 188.4 | .1813 | 302.9 | .1703 | 16 |
| LF-Net | 2020.3 | 100.2 | .1927 | 106.5 | .2344 | 141.0 | .2226 | 13 |
| SuperPoint | 2048.0 | 120.1 | .2577 | 126.8 | .2964 | 127.3 | .2676 | 11 |
| R2D2 (wasf-n16) | 2048.0 | 191.0 | .2829 | 215.6 | .3614 | 215.6 | .3614 | 8 |
| DoG-AffNet-HardNet | 2047.8 | 105.6 | .3589 | 152.1 | .4197 | 195.2 | .4175 | 1* |

Table 6.7: Test – Stereo results with 2k features. Same as Table 6.5. * The last group of results is obtained after the paper publication and not described in the text of the paper. Their rank does not influence other entries ranks.

| Method | NL [†] | SR [†] | RC [†] | TL [†] | mAA(5°) [†] | mAA(10°) [†] | ATE [↓] | Rank |
|--------------------------|-----------------|-----------------|-----------------|-----------------|----------------------|-----------------------|------------------|------|
| CV-SIFT | 1081.2 | 87.6 | 87.4 | 3.70 | .3718 | .4562 | .6136 | 13 |
| CV- $\sqrt{\text{SIFT}}$ | 1174.7 | 90.3 | 89.4 | 3.82 | .4074 | .4995 | .5589 | 12 |
| CV-SURF | 1186.6 | 90.2 | 88.6 | 3.55 | .3335 | .4184 | .6701 | 15 |
| CV-AKAZE | 1383.9 | 94.7 | 90.9 | 3.74 | .3393 | .4361 | .6422 | 14 |
| CV-ORB | 683.3 | 74.9 | 73.0 | 3.21 | .1422 | .1914 | .8153 | 19 |
| CV-FREAK | 1075.2 | 87.2 | 86.3 | 3.52 | .2578 | .3297 | .7169 | 17 |
| L2-Net | 1253.3 | 94.7 | 92.6 | 3.96 | .4369 | .5392 | .5419 | 9 |
| DoG-HardNet | 1338.2 | 96.3 | 93.7 | 4.03 | .4624 | .5661 | .5093 | 6 |
| Key.Net-HardNet | 1276.3 | 97.8 | 95.7 | 4.49 | .5050 | .6161 | .4902 | 2 |
| Key.Net-SOSNet | 1475.5 | 99.3 | 96.5 | 4.42 | .5229 | .6340 | .4853 | 1 |
| GeoDesc | 1133.6 | 93.6 | 91.3 | 4.02 | .4246 | .5244 | .5455 | 10 |
| ContextDesc | 1504.9 | 95.6 | 93.3 | 3.92 | .4529 | .5568 | .5327 | 7 |
| DoG-SOSNet | 1317.4 | 96.0 | 93.8 | 4.05 | .4739 | .5784 | .5194 | 5 |
| LogPolarDesc | 1410.2 | 96.0 | 93.8 | 4.05 | .4794 | .5849 | .5090 | 4 |
| D2-Net (SS) | 2357.9 | 98.9 | 94.7 | 3.39 | .2875 | .3943 | .7010 | 16 |
| D2-Net (MS) | 2177.3 | 98.2 | 93.4 | 3.01 | .1921 | .3007 | .7861 | 20 |
| LF-Net | 1385.0 | 95.6 | 90.4 | 4.14 | .4156 | .5141 | .5738 | 11 |
| SuperPoint | 1184.3 | 95.6 | 92.4 | 4.34 | .4423 | .5464 | .5457 | 8 |
| R2D2 (wasf-n16) | 1228.4 | 99.4 | 96.2 | 4.29 | .5045 | .6149 | .4956 | 3 |
| DoG-AffNet-HardNet | 1788.7 | 98.7 | 95.7 | 4.19 | .4771 | .5854 | .5114 | 4* |

Table 6.8: Test – Multiview results with 2k features. Same as Table 6.6. * The last group of results is obtained after the paper publication and not described in the text of the paper. Their rank does not influence other entries ranks.

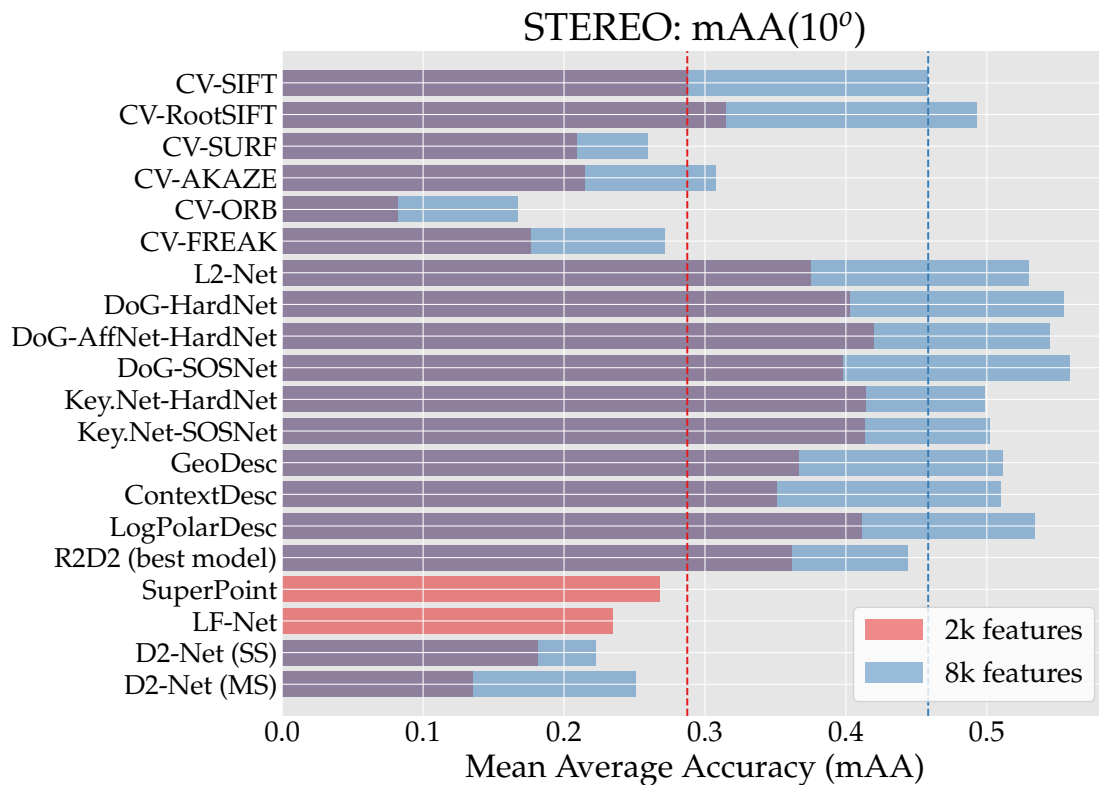


Figure 6.17: **Test – Stereo performance: 2k vs 8k features.** We compare the results obtained with different methods using either 2k or 8k features – we use DEGENSAC, which performs better than other RANSAC variants under most circumstances. Dashed lines indicate SIFT’s performance. For LF-Net and SuperPoint we do not include results with 8k features, as we failed to obtain meaningful results. For R2D2, we use the best model for each setting.

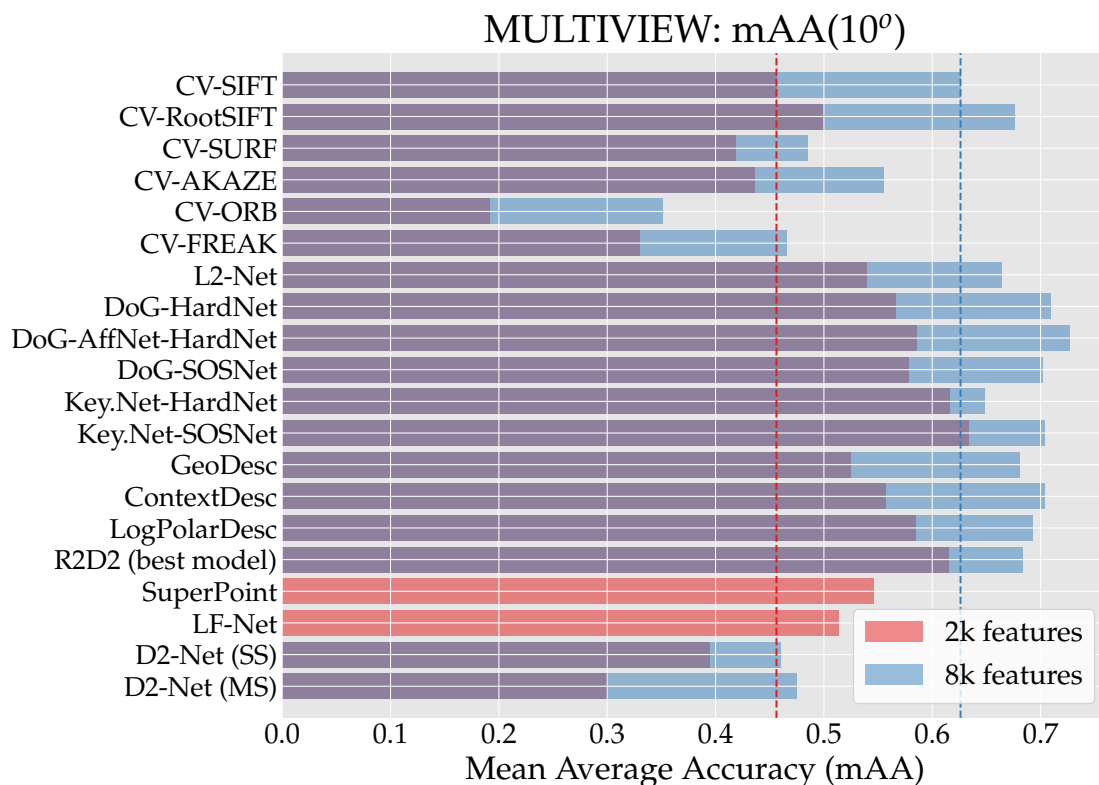


Figure 6.18: **Test – Multiview performance: 2k vs 8k features.** Same as Fig. 6.17, for multiview.

| Method | PyRANSAC | | Stereo Task DEGENSAC | | MAGSAC | | Multi-view Task | |
|--------------------|-----------|--------|-------------------------|--------|--------------|--------|-----------------|--------|
| | mAA(10°)↑ | Δ(%)↑ | mAA(10°)↑ | Δ(%)↑ | mAA(10°)↑ | Δ(%)↑ | mAA(10°)↑ | Δ(%)↑ |
| CV-SIFT | .4086 | +2.24 | .4751 | +3.65 | .4694 | +2.42 | .6815 | +8.85 |
| CV-√SIFT | .4205 | -0.53 | .4927 | -0.06 | .4848 | -1.87 | .6978 | +3.16 |
| CV-SURF | .2490 | +9.18 | .3071 | +18.42 | .2954 | +15.75 | .5750 | +18.67 |
| CV-AKAZE | .2857 | +11.18 | .3417 | +11.18 | .3316 | +9.23 | .6026 | +8.51 |
| CV-ORB | .1323 | +8.49 | .1856 | +10.87 | .1748 | +11.34 | .4171 | +18.88 |
| CV-FREAK | .2532 | +11.36 | .3204 | +18.18 | .3053 | +14.93 | .5574 | +19.79 |
| L2-Net | .4377 | -5.27 | .5012 | -5.35 | .4937 | -5.99 | .6951 | +4.62 |
| DoG-HardNet | .4427 | -7.80 | .5156 | -6.98 | .5056 | -8.11 | .7061 | -.50 |
| Key.Net-HardNet | .3081 | -22.92 | .4226 | -15.23 | .4012 | -15.36 | .6620 | +2.11 |
| GeoDesc | .4239 | -2.05 | .4924 | -3.67 | .4807 | -4.93 | .6956 | +2.25 |
| ContextDesc | .3976 | -15.11 | .4482 | -12.09 | .4535 | -11.83 | .6900 | -1.91 |
| DoG-SOSNet | .4439 | -7.21 | .5187 | -7.15 | .5073 | -8.04 | .7103 | +1.18 |
| LogPolarDesc | .4259 | -6.89 | .4898 | -8.27 | .4808 | -8.22 | .6871 | -.82 |
| D2-Net (SS) | .1231 | -36.32 | .1717 | -22.95 | .1608 | -20.86 | .4639 | +0.89 |
| D2-Net (MS) | .0998 | -53.78 | .1370 | -45.33 | .1316 | -43.29 | .4132 | -13.02 |
| R2D2 (wasf-n8-big) | .2218 | -39.78 | .3141 | -29.21 | .3032 | -28.43 | .6229 | -8.83 |

Table 6.9: **Test – Outlier pre-filtering with CNe (8k features)**. We report mAP at 10° with CNe, on stereo and multi-view, and its increase in performance w.r.t. Table 6.6 – positive Δ meaning CNe helps. When using CNe, we disable the ratio test.

| | CV-√SIFT | | HardNet | | SOSNet | | LogPolarDesc | |
|-----------|----------|-----------|---------|-----------|--------|-----------|--------------|-----------|
| | NI↑ | mAA(10°)↑ | NI↑ | mAA(10°)↑ | NI↑ | mAA(10°)↑ | NI↑ | mAA(10°)↑ |
| Standard | 281.7 | 0.4930 | 432.3 | 0.5543 | 424.6 | 0.5587 | 441.8 | 0.5340 |
| Upright | 270.0 | 0.4878 | 449.2 | 0.5542 | 432.9 | 0.5554 | 461.8 | 0.5409 |
| Δ (%) | -4.15 | -1.05 | +3.91 | -0.02 | +1.95 | -0.59 | +4.53 | +1.29 |
| Upright++ | 358.9 | 0.5075 | 527.6 | 0.5728 | 508.4 | 0.5738 | 543.2 | 0.5510 |
| Δ (%) | +27.41 | +2.94 | +22.04 | +3.34 | +19.74 | +2.70 | +22.95 | +3.18 |

Table 6.10: **Test – Stereo performance with upright descriptors (8k features)**. We report (NI) the number of inliers and mAA at 10° for the stereo task, using DEGENSAC. As DoG may return multiple orientations for the same point [140] (up to 30%), we report: (**top**) with orientation estimation; (**middle**) setting the orientation to zero while removing duplicates; and (**bottom**) adding new points until hitting the 8k-feature budget.

| | CV-√SIFT | | HardNet | | SOSNet | | LogPolarDesc | |
|-----------|----------|-----------|---------|-----------|--------|-----------|--------------|-----------|
| | NL↑ | mAA(10°)↑ | NL↑ | mAA(10°)↑ | NL↑ | mAA(10°)↑ | NL↑ | mAA(10°)↑ |
| Standard | 3312.1 | 0.6765 | 4001.4 | 0.7096 | 3796.0 | 0.7021 | 4054.6 | 0.6928 |
| Upright | 3485.1 | 0.6572 | 3594.6 | 0.6962 | 4025.1 | 0.7054 | 3737.4 | 0.6934 |
| Δ (%) | +5.22 | -2.85 | -10.17 | -1.89 | +6.04 | +0.47 | -7.82 | +0.09 |
| Upright++ | 4404.6 | 0.6792 | 4250.4 | 0.7231 | 3988.6 | 0.7129 | 4414.1 | 0.7109 |
| Δ (%) | +32.99 | +0.40 | +6.22 | +1.90 | +5.07 | +1.54 | +8.87 | +2.61 |

Table 6.11: **Test – Multiview performance with upright descriptors (8k features)**. Analogous to Table 6.10, on the multiview task. We report the number of landmarks (NL) instead of the number of inliers (NI).

| | CV-√SIFT | | HardNet | | SOSNet | | D2Net | |
|-------|----------|-----------|---------|-----------|--------|-----------|--------|-----------|
| | NI↑ | mAA(10°)↑ | NI↑ | mAA(10°)↑ | NI↑ | mAA(10°)↑ | NI↑ | mAA(10°)↑ |
| Exact | 281.7 | 0.4930 | 432.0 | 0.5532 | 424.3 | 0.5575 | 470.6 | 0.2506 |
| FLANN | 274.6 | 0.4879 | 363.3 | 0.5222 | 339.8 | 0.5179 | 338.9 | 0.2046 |
| Δ (%) | -2.52 | -1.03 | -15.90 | -5.60 | -19.92 | -7.10 | -27.99 | -18.36 |

Table 6.12: **Test – Stereo performance with OpenCV FLANN, using kd-tree approximate nearest neighbors (8k features)**. kd-tree parameters are: 4 trees, 128 checks. We report (NI) the number of inliers and mAA at 10° for the stereo task, using DEGENSAC.

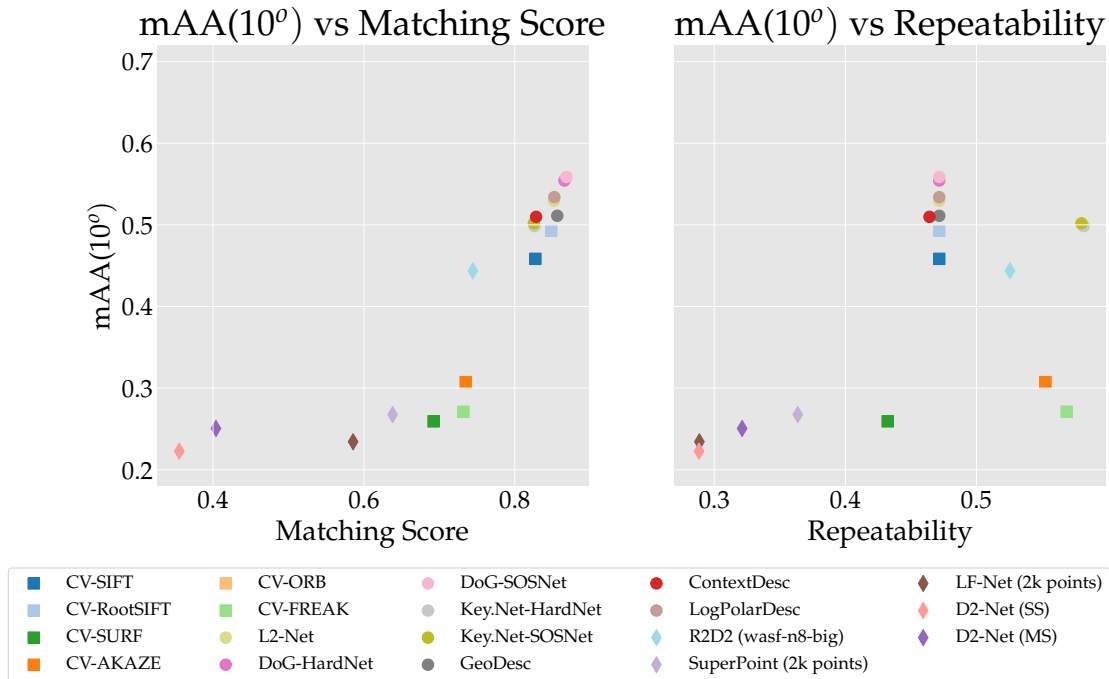


Figure 6.19: **Test – Downstream vs. traditional metrics (8k features)**. We cross-reference stereo mAA at 10° with repeatability and matching score, with a 3-pixel threshold.

aligned (upright), with the descriptor shouldering the increased invariance requirements. As our images meet this condition, we experiment with setting the orientation of keypoints to a fixed value (zero). DoG often returns multiple orientations for the same keypoint, so we consider two variants: one where we simply remove keypoints which become duplicates after setting the orientation to a constant value (Upright), and a second one where we fill out the budget with new keypoints (Upright++). We list the results in Table 6.10, for stereo, and Table 6.11, for multiview. Performance increases across the board with Upright++ – albeit by a small margin.

6.6.6 On the effect of approximate nearest neighbor matching — Table 6.12

While it is known that approximate nearest neighbor search algorithms have non-perfect recall [171], it is not clear how their usage influences the downstream performance. We thus compare the exact (brute-force) nearest neighbor search with a popular choice for the approximate nearest neighbor search, FLANN [171], as implemented in OpenCV. We experimented with different parameters and found that 4 trees and 128 checks provide a reasonable trade-off between precision and runtime.

We report the results with and without FLANN in Table 6.12.

The performance drop varies for different methods: from moderate 1% for RootSIFT, to 5-7% for HardNet and SOSNet, and 18% for D2-Net.

6.6.7 Pose mAA vs. traditional metrics — Fig. 6.19

To examine the relationship between our pose metric and traditional metrics, we compare mAA against repeatability and matching score on the stereo task, with DEGENSAC. While the matching score seems to correlate with mAA, repeatability is harder to interpret. However, note that even for the matching score, which shows correlation, higher value does not guarantee high mAA – see *e.g.* RootSIFT vs ContextDesc. We remind the reader that, as explained in Section 6.4, our implementation differs from the classical formulation, as many methods do not have a strict notion of a support region. We compute these metrics at a 3-pixel threshold, and provide more granular results in Section 6.7.

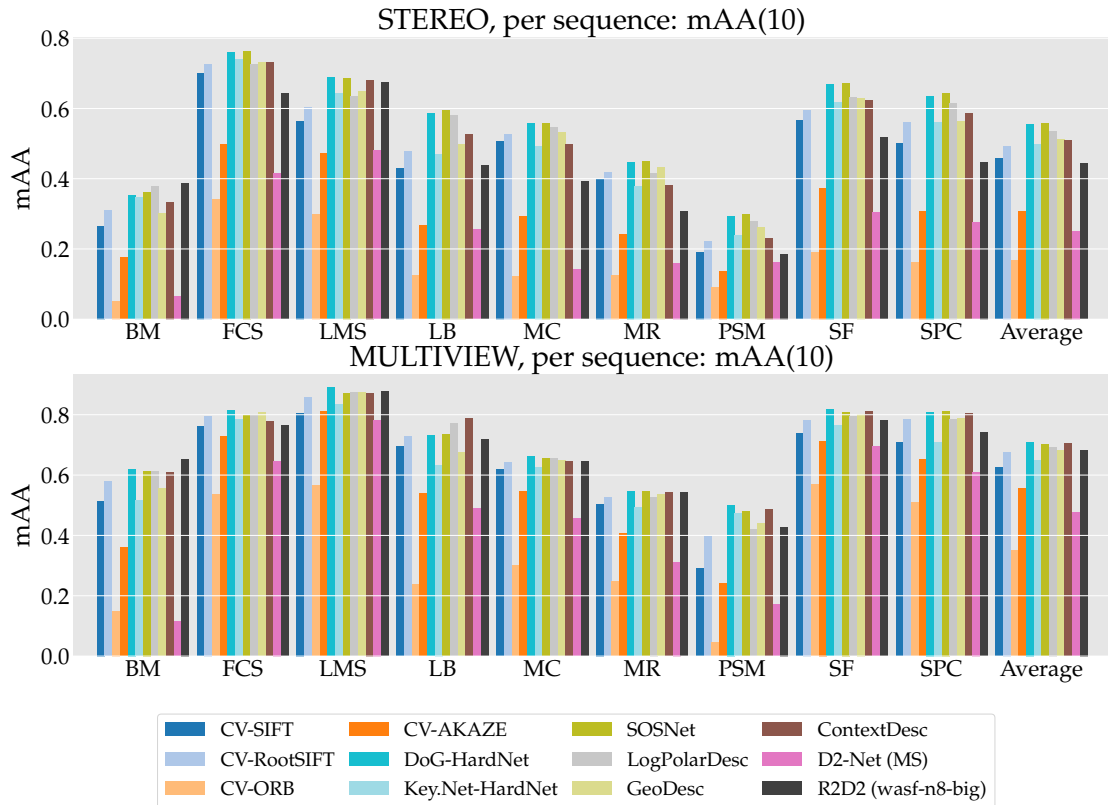


Figure 6.20: **Test – Breakdown by scene (8k features)**. For the stereo task, we use DEGENSAC. Note how performance can vary drastically between scenes, and the relative rank of a given local feature fluctuates as well. Please refer to Table 6.1 for a legend.

As shown, all methods based on DoG are clustered, as they operate on the same keypoints. Key.Net obtains the best repeatability, but performs worse than DoG in terms of mAA with the same descriptors (HardNet). AKAZE and FREAK perform surprisingly well in terms of repeatability – #2 and #3, respectively – but obtain a low mAA, which may be related to their descriptors, which are binary. R2D2 shows good repeatability but a poor matching score and is outperformed by DoG-based features.

6.6.8 Breakdown by scene — Fig. 6.20

Results may vary drastically from scene to scene, as shown in Fig. 6.20. A given method may also perform better on some than others – for instance, D2-Net nears the state of the art on “Lincoln Memorial Statue”, but is 5x times worse on “British Museum”. AKAZE and ORB show similar behaviour. This can provide valuable insights on limitations and failure cases.

6.6.9 Breakdown by co-visibility — Figs. 6.21 and 6.22

Next, in Fig. 6.21 we evaluate stereo performance at different co-visibility thresholds, for several local feature methods, using DEGENSAC. Bins are encoded as $v+$, with v the co-visibility threshold, and include all image pairs with a co-visibility value larger or equal than v . This means that the first bin may contain unmatchable images – we use 0.1+ for all other experiments in the Chapter. We do not report values above 0.6+ as there are only a handful of them and thus the results are very noisy.

Performance for all local features and RANSAC variants increases with the co-visibility threshold, as expected. Results are consistent, with end-to-end methods performing better at higher than co-visibility than lower, and single-scale D2-Net outperforming its multi-scale counterpart at 0.4+ and above, where the images are more likely aligned in terms of scale.

We also break down different RANSAC methods in Fig. 6.22, along with different local features, including AKAZE (binary), SIFT (also handcrafted), HardNet (learned descriptor on DoG points)

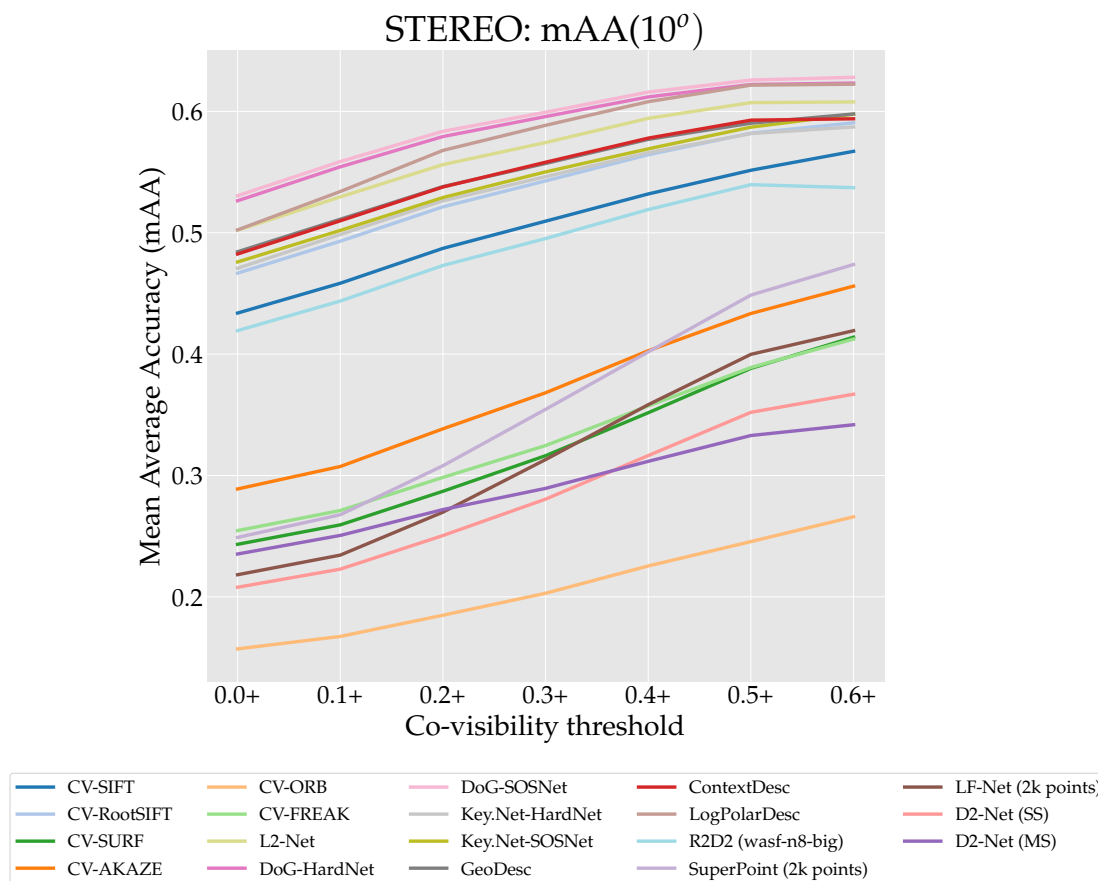


Figure 6.21: **Test – Local features vs. co-visibility (8k features)**. We plot mAA at 10° on the stereo task – using DEGENSAC – at different co-visibility thresholds for different local feature types. Bin “0+” consists of all possible image pairs, including potentially unmatchable ones. Bin “0.1+” includes pairs with a minimum co-visibility value of 0.1 – this is the default value we use for all other experiments in this paper – and so forth. Results are mostly consistent, with end-to-end methods performing better at higher than lower co-visibility.

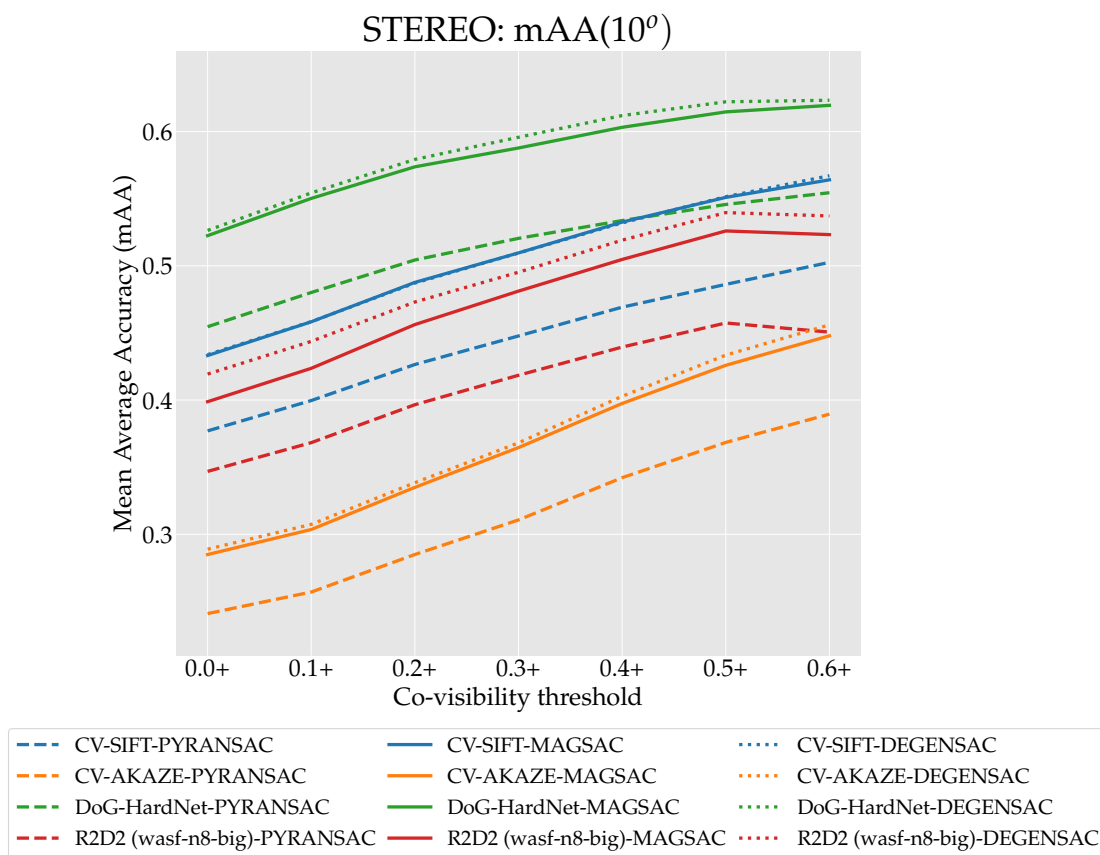


Figure 6.22: **Test – RANSAC vs. co-visibility (8k features)**. We plot mAA at 10^0 on the stereo task at different co-visibility thresholds for different RANSAC variants, binning the results as in Fig. 6.21. We pair them with different local features methods. The difference in performance between RANSAC variants seems consistent across pairs, regardless of their difficulty.

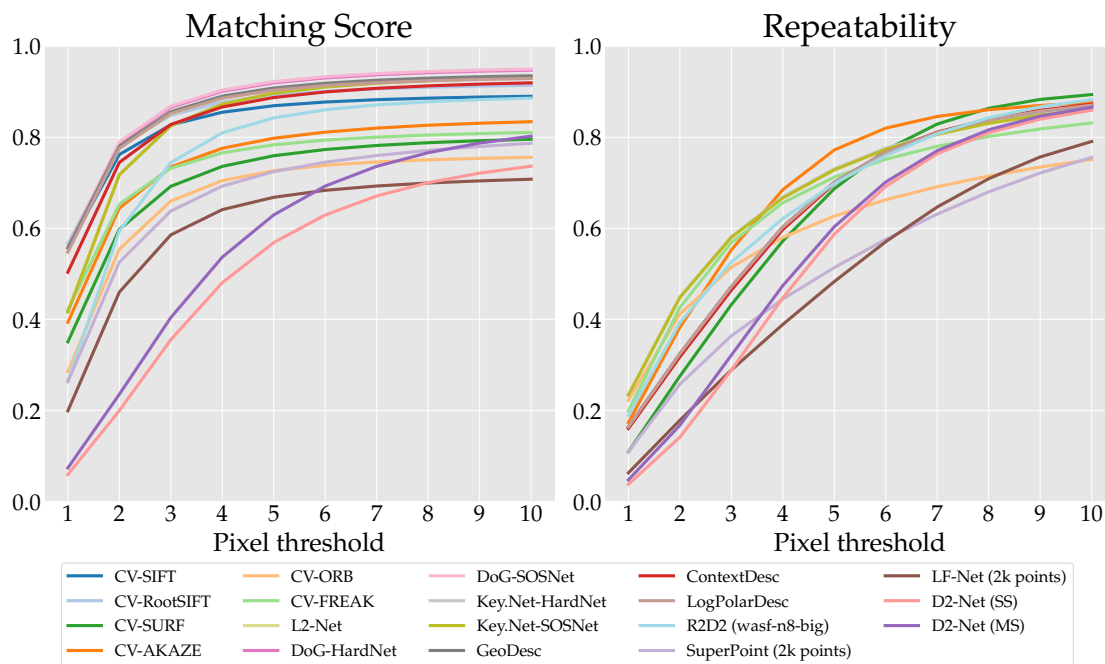


Figure 6.23: **Test – Classical metrics, by pixel threshold (8k features)**. Repeatability and matching score computed at different pixel thresholds are shown.

and R2D2 (learned end-to-end). We do not observe significant variations in the trend of each curve as we swap RANSAC and local feature methods. DEGENSAC and MAGSAC show very similar performance.

6.6.10 Classical metrics vs. pixel threshold — Fig. 6.23

In Fig. 6.19 we plot repeatability and matching score against mAA at a fixed error threshold of 3 pixels. In Fig. 6.23 we show them at different pixel thresholds. End-to-end methods tend to perform better at higher pixel thresholds, which is expected – D2-Net in particular extracts keypoints from downsampled feature maps. These results are computed from the depth maps estimated by COLMAP, which are not pixel-perfect, so the results for very low thresholds are not completely trustworthy.

Note that repeatability is typically lower than matching score, which might be counter-intuitive as the latter is more strict – it requires two features to be nearest-neighbors in descriptor space in addition to being physically close (after reprojection). We compute repeatability with the raw set of keypoints, whereas the matching score is computed with optimal matching settings – bidirectional matching with the “both” strategy and the ratio test. This results in a much smaller pool – 8k features are typically narrowed down to 200–400 matches (see Table 6.13). This better isolates the performance of the detector and the descriptor where it matters.

6.6.11 Breakdown by angular threshold — Figs. 6.24 and 6.25

We summarize pose accuracy by mAA at 10° in order to have a single, easy-to-interpret number. In this section we show how performance varies across different error thresholds – we look at the *average accuracy* at every angular error threshold, rather than the mean Average Accuracy. Fig. 6.24 plots performance on stereo and multiview for different local feature types, showing that the ranks remain consistent across thresholds. Fig. 6.25 shows how it affects different RANSAC variants, with four different local features. Again, ranks do not change. DEGENSAC and MAGSAC perform nearly identically for all features, except for R2D2. The consistency in the ranks demonstrate that summarizing the results with a single number, mAA at 10° , is reasonable.

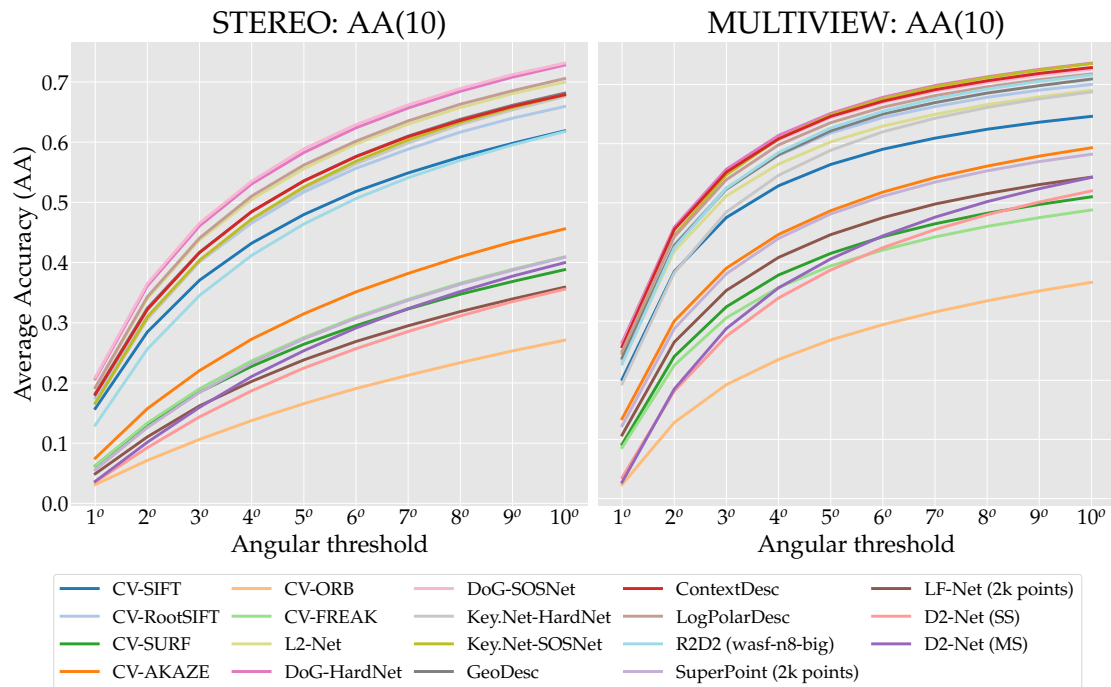


Figure 6.24: **Test – Breakdown by angular threshold, for local features (8k features).** Accuracy in pose estimation is evaluated at every error threshold. Mean Average Accuracy used in the rest of paper is the area under this curve. Ranks are consistent across thresholds.

6.6.12 Qualitative results — Figs. 6.28, 6.29, 6.30, and 6.31

Figs. 6.28 and 6.29 show qualitative results for the stereo task. We draw the inliers produced by DEGENSAC and color-code them using the ground truth depth maps, from green (0) to yellow (5 pixels off) if they are correct, and in red if they are incorrect (more than 5 pixels off). Matches including keypoints which fall on occluded pixels are drawn in blue. Note that while the depth maps are somewhat noisy and not pixel-accurate, they are sufficient for this purpose. Notice how the best performing methods have more correct matches that are well spread across the overlapping region.

Fig. 6.30 shows qualitative results for the multiview task, for handcrafted detectors. We illustrate it by drawing keypoints used in the SfM reconstruction in blue and the rest in red. It showcases the importance of the detector, specially on unmatched regions such as the sky. ORB and Hessian keypoints are too concentrated in the high-contrast regions, failing to provide evenly-distributed features. In contrast, SURF fails to filter-out background and sky points. Fig. 6.31 shows results for learned methods: DoG+HardNet, Key.Net+HardNet, SuperPoint, R2D2, and D2-Net (multiscale). They have different detection patterns: Key.Net resembles a “cleaned” version of DoG, while R2D2 seems to be evenly distributed. D2Net looks rather noisy, while SuperPoint fires precisely on corner points on structured parts, and sometimes form a regular grid on sky-like homogeneous regions, which might be due to the method running out of locations to place points at – note however that the best results were obtained with larger NMS (non-maxima suppression) thresholds.

6.7 Further results and considerations

In this section we provide additional results on the validation set. These include a study of the typical outlier ratios under optimal settings in Section 6.7.1, matching with FGINN in Section 6.7.2, image-preprocessing techniques for feature extraction in Section 6.7.3, and a breakdown of the optimal settings in Section 6.7.4, provided to serve as a reference.

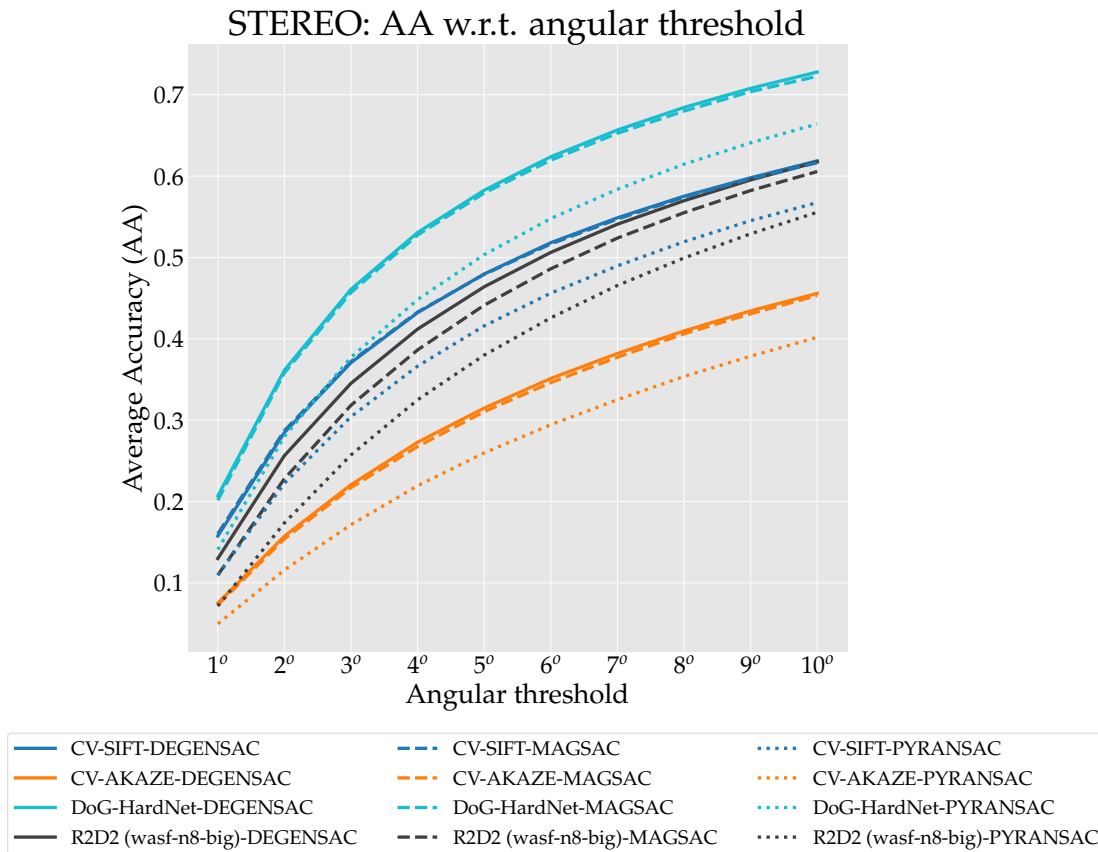


Figure 6.25: **Test – Breakdown by angular threshold, for RANSAC (8k features)**. We plot performance on the stereo task at different angular error thresholds, for four different local features and three RANSAC variants: MAGSAC (solid line), DEGENSAC (dashed line), and PyRANSAC (dotted line). We observe a similar behaviour across all thresholds.

| Method | # matches | # inliers | Ratio (%) | mAA(10°) |
|--------------------------|-----------|-----------|-----------|----------|
| CV-SIFT | 328.3 | 113.0 | 34.4 | 0.548 |
| CV- $\sqrt{\text{SIFT}}$ | 331.6 | 131.4 | 39.6 | 0.584 |
| CV-SURF | 221.5 | 77.4 | 35.0 | 0.309 |
| CV-AKAZE | 369.1 | 143.5 | 38.9 | 0.360 |
| CV-ORB | 193.4 | 74.7 | 38.6 | 0.265 |
| CV-FREAK | 216.6 | 75.5 | 34.8 | 0.329 |
| DoG-HardNet | 433.1 | 173.1 | 40.0 | 0.627 |
| Key.Net-HardNet | 644.1 | 166.6 | 25.9 | 0.580 |
| L2Net | 368.0 | 144.0 | 39.1 | 0.601 |
| GeoDesc | 322.6 | 133.4 | 41.3 | 0.564 |
| ContextDesc | 560.8 | 165.9 | 29.6 | 0.587 |
| SOSNet | 391.7 | 161.8 | 41.3 | 0.621 |
| LogPolarDesc | 522.8 | 175.2 | 33.5 | 0.618 |
| SuperPoint (2k points) | 202.0 | 62.6 | 31.0 | 0.312 |
| LF-Net (2k points) | 165.9 | 73.2 | 44.1 | 0.293 |
| D2-Net (SS) | 657.3 | 148.9 | 22.7 | 0.263 |
| D2-Net (MS) | 601.7 | 161.7 | 26.9 | 0.343 |
| R2D2 (wasf-n8-big) | 901.5 | 477.3 | 52.9 | 0.473 |

Table 6.13: **Validation – Number of inliers with optimal settings**. We use 8k features with optimal parameters, and PyRANSAC as a robust estimator. The number of inliers varies significantly between methods, despite tuning the matcher and the ratio test, and lower inlier ratios tend to correlate with low performance.

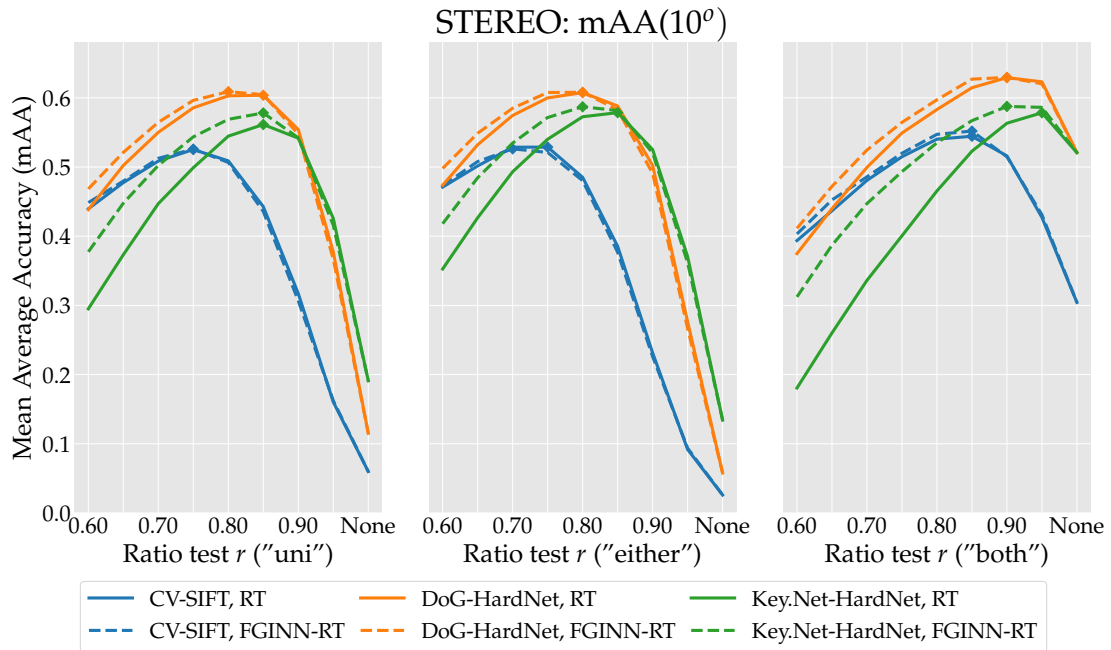


Figure 6.26: **Validation – FGINN vs. ratio test (8k features)**. We evaluate the ratio test with FGINN [162] (dashed line), and the standard ratio test (solid line). With FGINN, the valid range for r – the ratio test threshold – is significantly wider, but the best performance with the “both” matching strategy is not significantly better than for the standard ratio test.

6.7.1 Number of inliers per step — Table 6.13

We list the number of input matches and their *resulting* inliers for the stereo task, in Table 6.13. As before, we remind the reader that these inliers are what each method reports, *i.e.*, the matches that are actually used to estimate the poses. We list the number of input matches, the number of inliers produced by each method (which may still contain outliers), their ratio, and the mAA at 10° . We use PyRANSAC with optimal settings for each method, the ratio test, and bidirectional matching with the “both” strategy.

We see that inlier-to-outlier ratios hover around 35–40% for all features relying on classical detectors. Key.Net with HardNet descriptors sees a significant drop in inlier ratio and mAA, when compared to its DoG counterpart. D2-Net similarly has inlier ratios around 25%. R2D2 has the largest inlier ratio by far at 53%, but is outperformed by many other methods in terms of mAA, suggesting that many of these are not actual inliers. In general, we observe that the methods which produce a large number of matches, such as Key.Net (600+), D2-Net (600+) or R2D2 (900+) are less accurate in terms of pose estimation.

6.7.2 Feature matching with an advanced ratio test — Fig. 6.26

We also compare the benefits of applying first-geometrically-inconsistent-neighbor-ratio (FGINN) [162] to DoG/SIFT, DoG/HardNet and Key.Net/HardNet, against Lowe’s standard ratio test [140]. FGINN performs the ratio-test with second-nearest neighbors that are “far enough” from the tentative match (10 pixels in [162]). In other words, it loosens the test to allow for nearby-thus-similar points. We test it for 3 matching strategies: unidirectional (“uni”), “both” and “either”. We report the results in Fig. 6.26. As shown, FGINN provides minor improvements over the standard ratio test in case of unidirectional matching, and not as much when “both” is used. It also behaves differently compared to the standard strategy, in that performance at stricter thresholds degrades less.

6.7.3 Image pre-processing — Fig. 6.27

Contrast normalization is key to invariance against illumination changes – local feature methods typically apply some normalization strategy over small patches [140, 158]. Therefore, we experiment

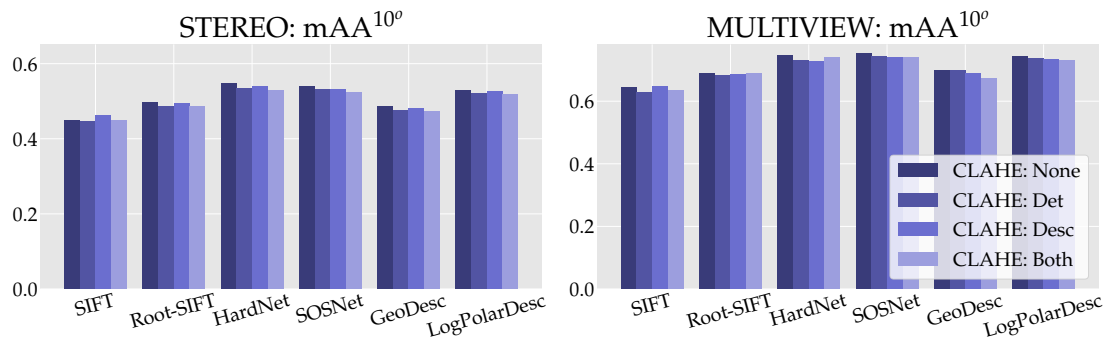


Figure 6.27: **Validation – Image pre-processing with CLAHE (8k features).** We experiment with contrast normalization for keypoint and descriptor extraction. Results are very similar with or without it.

with contrast-limited adaptive histogram equalization (CLAHE) [302], as implemented in OpenCV. We apply it prior to feature detection and/or description with SIFT and several learned descriptors, and display the results in Fig. 6.27. Performance decreases for all learned methods, presumably because they are not trained for it. Contrary to our initial expectations, SIFT does not benefit much from it either: the only increase in performance comes from applying it for descriptor extraction, at 2.5% relative for stereo task and 0.56% relative for multi-view. This might be due to the small number of night-time images in our data. It also falls in line with the observations in [64], which show that SIFT descriptors are actually optimal under certain assumptions.

6.7.4 Optimal settings breakdown — Tables 6.14 and 6.15

For the sake of clarity, we summarize the optimal hyperparameter combinations from Figs. 6.9, 6.10 and 6.11 in Table 6.14 (for 8k features) and Table 6.15 (for 2k features). We set the confidence value to $\tau=0.999999$ for all RANSAC variants. Notice how it is better to have more features and a stricter ratio test threshold to filter them out, than having fewer features from the beginning.

6.8 Summary

We introduce a comprehensive benchmark for local features and robust estimation algorithms. The modular structure of its pipeline allows to easily integrate, configure, and combine methods and heuristics. We demonstrate this by evaluating dozens of popular algorithms, from seminal works to the cutting edge of machine learning research, and show that classical solutions may still outperform the perceived state of the art with proper settings.

The experiments carried out through the benchmark and reported in this paper have already revealed unexpected, non-intuitive properties of various components of the SfM pipeline, which will benefit SfM development, *e.g.*, the need to tune RANSAC to the particular feature detector *and* descriptor and to select specific settings for a particular RANSAC variant. Other interesting facts have been uncovered by our tests, such as that the optimal set-ups across different tasks (stereo and multiview) may differ, or that methods that perform better on proxy tasks, like patch retrieval or repeatability, may be inferior on the downstream task. Our work is open-sourced and makes the basis of an open challenge for image matching with sparse methods.

| | η_{PyR} | η_{DEGEN} | η_{GCR} | η_{MAG} | r_{stereo} | $r_{\text{multiview}}$ |
|--------------------------|---------------------|-----------------------|---------------------|---------------------|---------------------|------------------------|
| CV-SIFT | 0.25 | 0.5 | 0.5 | 1.25 | 0.85 | 0.75 |
| CV- $\sqrt{\text{SIFT}}$ | 0.25 | 0.5 | 0.5 | 1.25 | 0.85 | 0.85 |
| CV-SURF | 0.75 | 0.75 | 0.75 | 2 | 0.85 | 0.90 |
| CV-AKAZE | 0.25 | 0.75 | 0.75 | 1.5 | 0.85 | 0.90 |
| CV-ORB | 0.75 | 1 | 1.25 | 2 | 0.85 | 0.95 |
| CV-FREAK | 0.5 | 0.5 | 0.75 | 2 | 0.85 | 0.85 |
| VL-DoG-SIFT | 0.25 | 0.5 | 0.5 | 1.5 | 0.85 | 0.80 |
| VL-DoGAff-SIFT | 0.25 | 0.5 | 0.5 | 1.5 | 0.85 | 0.80 |
| VL-Hess-SIFT | 0.2 | 0.5 | 0.5 | 1.5 | 0.85 | 0.80 |
| VL-HessAffNet-SIFT | 0.25 | 0.5 | 0.5 | 1 | 0.85 | 0.80 |
| CV-DoG/HardNet | 0.25 | 0.5 | 0.5 | 1.5 | 0.90 | 0.80 |
| KeyNet/Hardnet | 0.5 | 0.75 | 0.75 | 2 | 0.95 | 0.85 |
| KeyNet/SOSNet | 0.25 | 0.75 | 0.75 | 1.5 | 0.95 | 0.95 |
| CV-DoG/L2Net | 0.2 | 0.5 | 0.5 | 1.5 | 0.90 | 0.80 |
| CV-DoG/GeoDesc | 0.2 | 0.5 | 0.75 | 1.5 | 0.90 | 0.85 |
| ContextDesc | 0.25 | 0.75 | 0.5 | 1 | 0.95 | 0.85 |
| CV-DoG/SOSNet | 0.25 | 0.5 | 0.75 | 1.5 | 0.90 | 0.80 |
| CV-DoG/LogPolarDesc | 0.2 | 0.5 | 0.5 | 1.5 | 0.90 | 0.80 |
| D2-Net (SS) | 1 | 2 | 2 | 7.5 | — | — |
| D2-Net (MS) | 1 | 2 | 2 | 5 | — | — |
| R2D2 (wasf-n8-big) | 0.75 | 1.25 | 1.25 | 2 | — | 0.95 |
| CV-DoG/TFeat | 0.25 | 0.5 | — | 1.25 | 0.85 | 0.80 |
| CV-DoG/MKD-Concat | 0.25 | 0.5 | — | 1.25 | 0.85 | 0.80 |
| CV-DoGAffNet/HardNet | 0.25 | 0.5 | — | 1.25 | 0.85 | 0.85 |

Table 6.14: **Optimal hyper-parameters with 8k features.** We summarize the optimal hyperparameters – the maximum number of RANSAC iterations η and the ratio test threshold r – for each combination of methods. The number of RANSAC iterations Γ is set to 250k for PyRANSAC, 50k for DEGENSAC, and 10k for both GC-RANSAC and MAGSAC (for the experiments on the validation set). We use bidirectional matching with the “both” strategy.

| | η_{PyR} | η_{DEGEN} | η_{MAG} | r_{stereo} | $r_{\text{multiview}}$ |
|--------------------------|---------------------|-----------------------|---------------------|---------------------|------------------------|
| CV-SIFT | 0.75 | 0.5 | 2 | 0.90 | 0.90 |
| CV- $\sqrt{\text{SIFT}}$ | 0.5 | 0.5 | 1.25 | 0.90 | 0.90 |
| CV-SURF | 0.25 | 1 | 3 | 0.90 | 0.95 |
| CV-AKAZE | 1 | 0.75 | 2 | 0.90 | 0.95 |
| CV-ORB | 1 | 1.25 | 3 | 0.90 | 0.90 |
| CV-FREAK | 0.75 | 0.75 | 2 | 0.9 | 0.95 |
| CV-DoG/HardNet | 0.5 | 0.5 | 1.5 | 0.95 | 0.90 |
| KeyNet/HardNet | 0.5 | 0.75 | 2.5 | 0.95 | 0.90 |
| KeyNet/SOSNet | 0.5 | 0.75 | 1.5 | 0.95 | 0.95 |
| CV-DoG/L2Net | 0.25 | 0.5 | 1.5 | 0.90 | 0.90 |
| CV-DoG/GeoDesc | 0.5 | 0.5 | 1.5 | 0.95 | 0.90 |
| ContextDesc | 0.75 | 0.75 | 2 | 0.95 | 0.95 |
| CV-DoG/SOSNet | 0.5 | 0.75 | 1.5 | 0.95 | 0.90 |
| CV-DoG/LogPolarDesc | 0.5 | 0.5 | 1.5 | 0.95 | 0.90 |
| D2-Net (SS) | 1.5 | 2 | 7.5 | — | — |
| D2-Net (MS) | 1.5 | 2 | 10 | — | — |
| SuperPoint | 0.75 | 1 | 3 | 0.95 | 0.90 |
| LF-Net | 1 | 1 | 4 | 0.95 | 0.95 |
| R2D2 (wasf-n16) | 1.5 | 1.25 | 2 | — | — |
| CV-DoGAffNet/HardNet | 0.25 | 0.5 | 1.25 | 0.95 | 0.95 |

Table 6.15: **Optimal hyper-parameters with 2k features.** Equivalent to Table 6.14. We do not evaluate GC-RANSAC, as it is always outperformed by DEGENSAC and MAGSAC, but keep PyRANSAC as a baseline RANSAC.

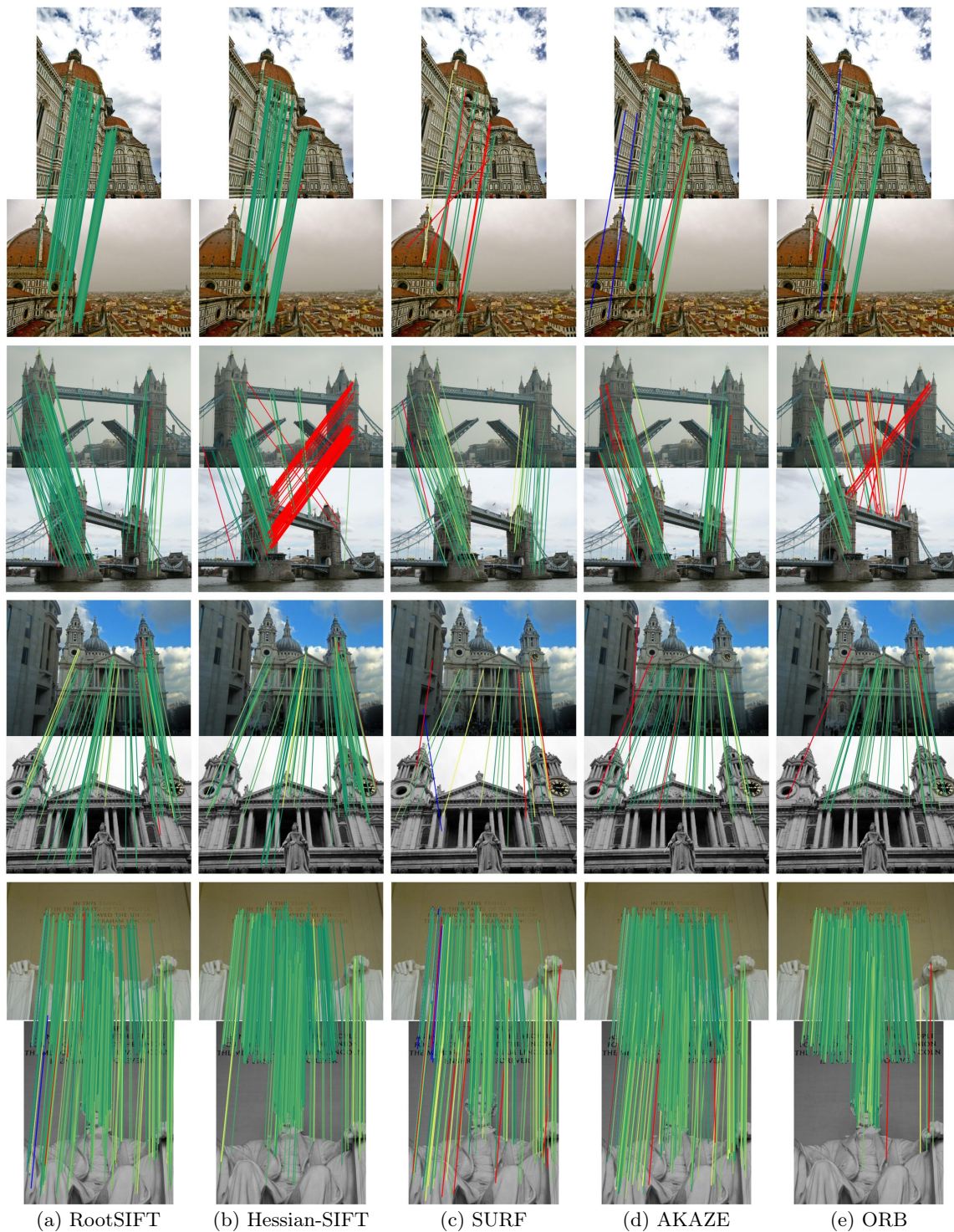


Figure 6.28: **Qualitative results for the stereo task – “Classical” features.** We plot the matches predicted by each local feature, with DEGENSAC. Matches above a 5-pixel error threshold are drawn in **red**, and those below are color-coded by their error, from 0 (**green**) to 5 pixels (**yellow**). Matches for which we do not have depth estimates are drawn in **blue**.

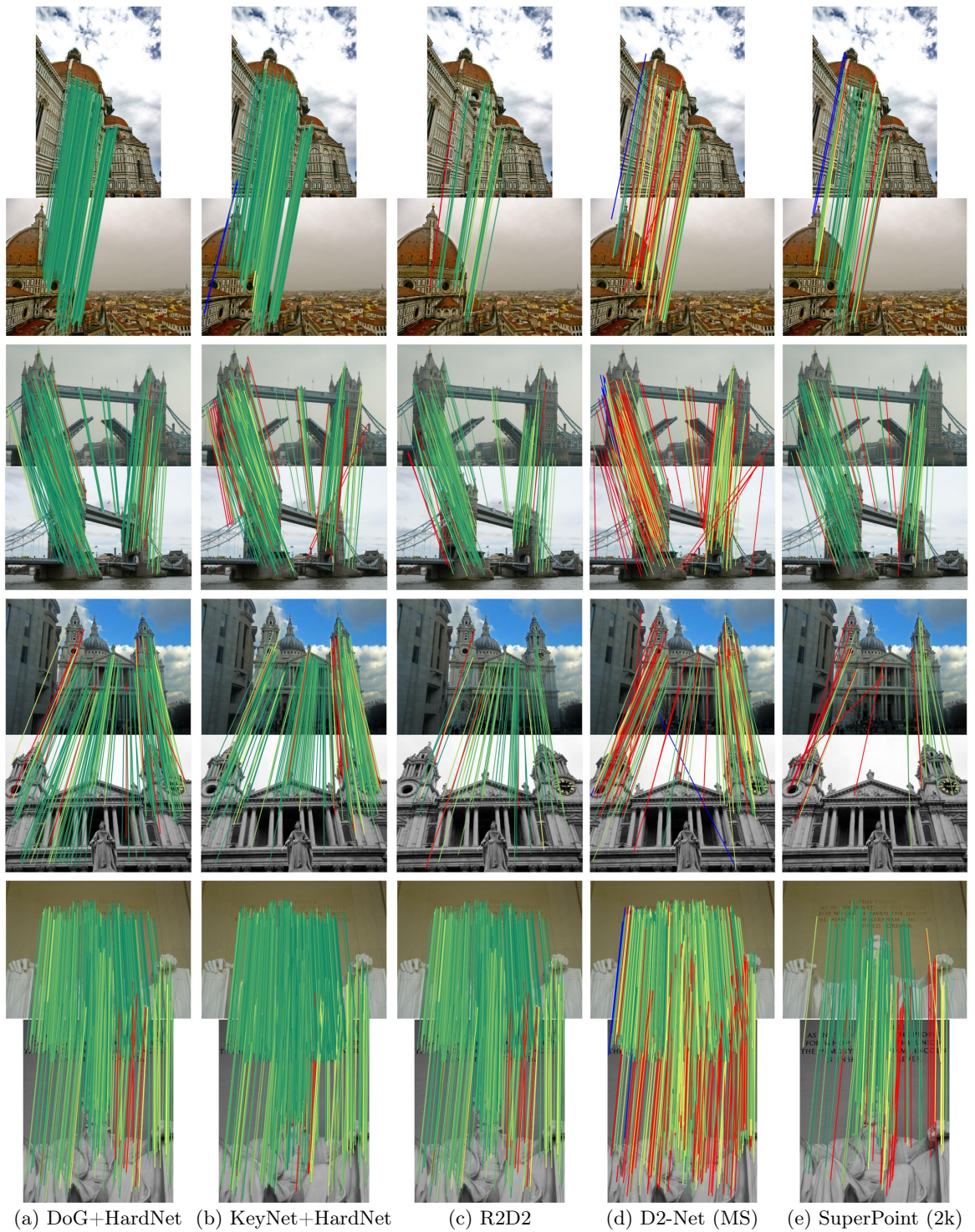


Figure 6.29: **Qualitative results for the stereo task – Learned features.** Color-coded as in Fig. 6.28.

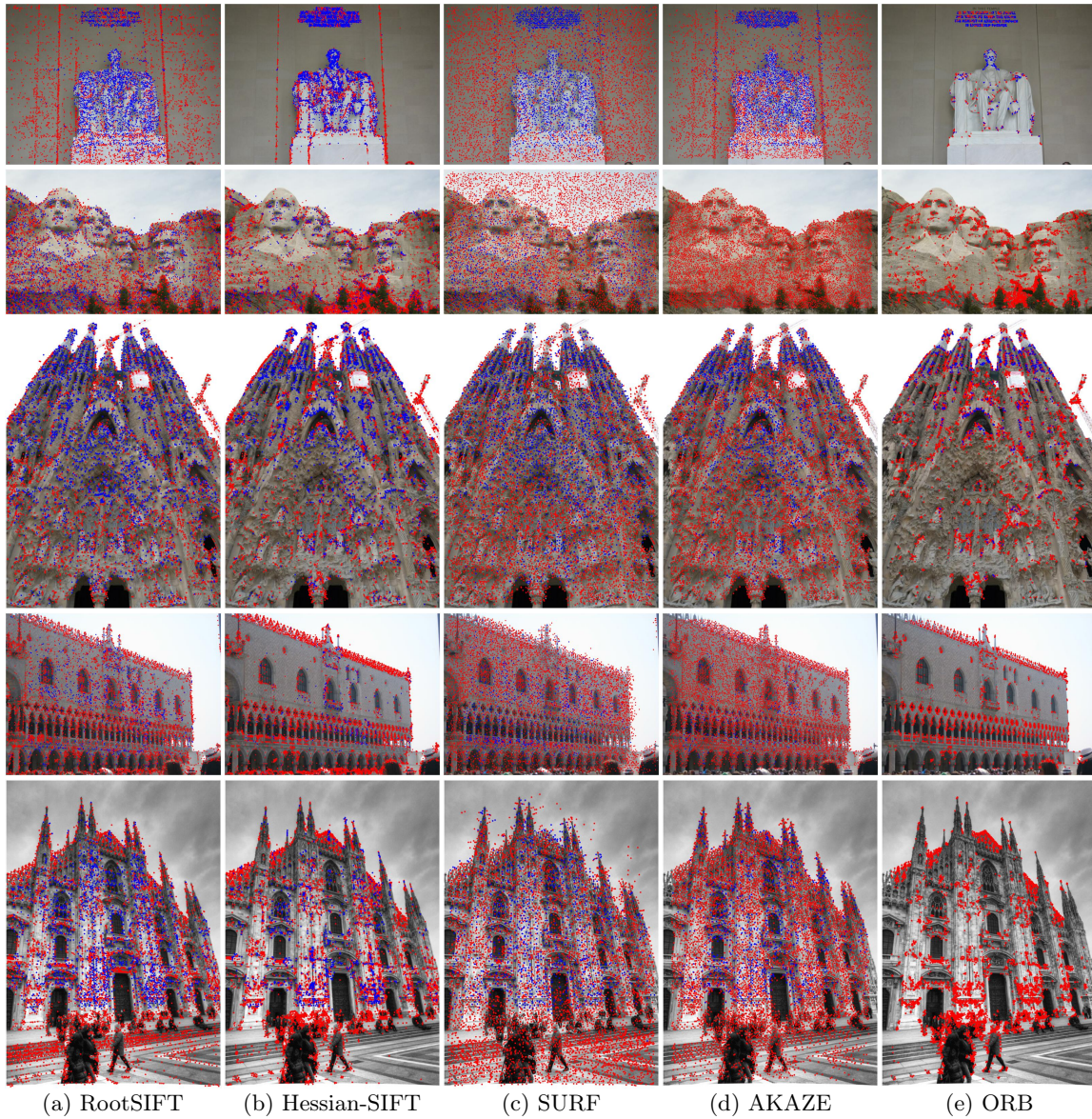


Figure 6.30: **Qualitative results for the multi-view task – “Classical” features.** We show images and the keypoints detected on them for different methods, with the points reconstructed by COLMAP in **blue**, and the ones that are not used in the 3D model in **red** – more blue points indicate denser 3D models. These results correspond to the multiview-task for a 25-image subset.

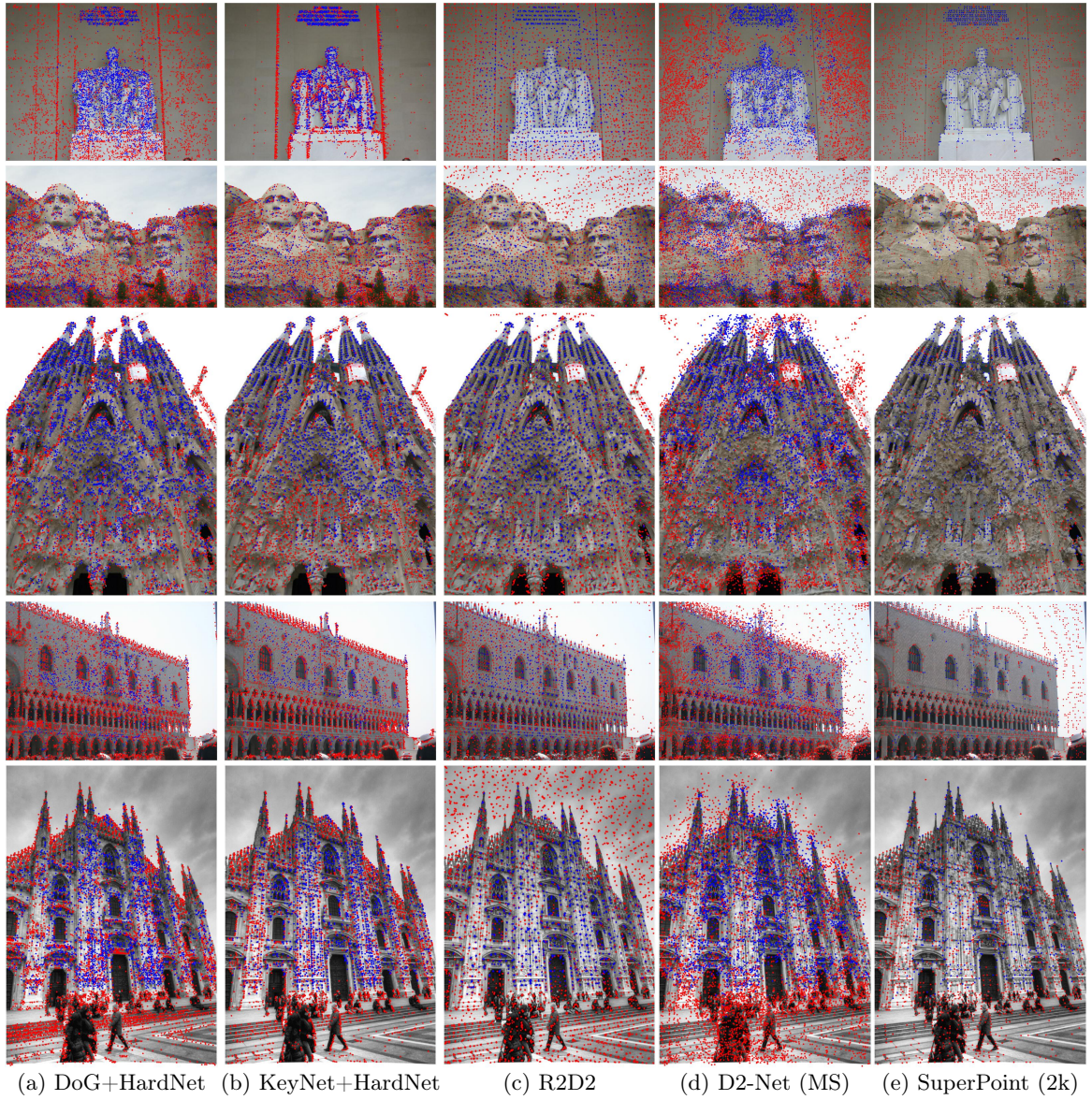


Figure 6.31: **Qualitative results for the multi-view task – Learned features.** Color-coded as in Fig. 6.30.

CHAPTER 7

Questions from Past and Ideas for Future Research

Each Chapter ends with its conclusion. To conclude the thesis as a whole, I would like to present some open questions that I find to be puzzling and important as well as research directions, which I find to be worth exploring in the short and long term. These questions and ideas arise while I was working on the thesis and are directly related to it. As you may noticed, I have switched “we” to “I” to underline the subjectiveness of the open directions I am going to present.

7.1 Questions

7.1.1 Should one make the training-time setup as close as possible to the test time setup?

While it is believed to be a good idea to remove the gap between training and test conditions with an evidence for doing so [266], it has not always led to the best results. For example, HardNet [158] borrows the idea of using the second nearest neighbor (SNN) descriptor ratio [140] from the SIFT paper. However, we have discovered that using SNN ratio for descriptor learning leads to inferior results, confirming the results first reported in [18]. We had to modify the loss function to become the triplet margin loss, which optimizes the distance difference instead of the ratio. Another example is the patch sampling procedure for HardNetAmos training [199] – there is little to no difference between random points as “keypoints” and actual detections of the Hessian detector when used for local descriptor training. Yet another example is the non-maxima suppression method applied to the DISK [271] local features. The paper reports that it is beneficial to replace train-time grid-based approach with a more classical window non-maxima suppression. The question is – could we tell when to reduce train-test gap and when not to, and to what extent?

7.1.2 End-to-end or stop gradients?

Some papers – LIFT [181], SuperPoint [62], SuperGlue [216] – report the benefits of end-to-end gradient propagation for model performance. Others – AffNet [161], DELF [179], DELG [44] use the stop-gradient operation and report problems otherwise. DELF and DELG stop the gradients of the descriptor-related loss function and not propagate them to the detector part. Otherwise, the overall representation quality is worse and overfitting is observed. AffNet does not propagate the gradient to the negative examples in its modified triplet margin loss. Otherwise, AffNet training does not converge when trained with HardNet, see Chapter 4 for more details.

On the other hand, SuperPoint [62] and R2D2 [207] optimize both detector and descriptor losses jointly without stopping the gradients. Why do some methods benefit from the joint optimization and other methods benefit from the stop gradient? How does stop-gradient operation help and why? A recent paper by Tian *et al.* [255] starts exploring the influence of the stop-gradient operation in the context of a specific case of contrastive learning. However, there are more questions than answers in that direction so far.

7.1.3 Evaluation: typical use-case sampling or focus on corner cases?

In the past, there was a tendency to create small datasets, which were explicitly curated w.r.t. of the nuisance factors they contain. Oxford-Affine dataset [151, 152] is a good example of it – the problems were split into “categories” like “blur”, “rotation”, “viewpoint change”. If the model or algorithm can deal with the hardest cases, one could assume that it will work in the less challenging conditions as well. This might not be the case, because the combination of mild illumination and viewpoint change could be harder to handle than the case, which contains extreme changes, but only in a single nuisance factor. In other words, one might be able to match a day versus night photo taken from the static camera, but not able to match day versus evening, when the camera moved slightly.

Now the tendency is to gather as much data as possible from the “real world” and hope that it would cover the nuisance factors, which are common in the real world [179, 109]. Sometimes it is true. For example, early research in image matching tends to design methods to be rotation invariant [140, 212]. Later, the introduction of the larger scale datasets, scraped from the internet [188, 179, 109] led to the spread of “up-is-up” assumption [186], baked into the local features pipeline [62, 69]. Rotational invariance in such scenarios hurts more than helps – the most general method would always work worse than those, which are tailored to the specific situation.

The question is – how best to combine both approaches – curated datasets, which contain corner cases and the “typical use-case” datasets?

7.2 Future research directions

7.2.1 Application-specific local features

Current local features are designed to be general. SIFT works reasonably well for various environments, as well as applications: 3D reconstruction, SLAM, image retrieval, etc. Learned features, like SuperPoint or R2D2 are biased towards the data they are trained on, but still have nothing domain specific in their design.

Despite the lack of domain-specific design, various local features have different properties: they are more or less robust to nuisance factors like illumination and camera position. Some of them, e.g., DELF, are poorly localized, while others – SIFT, SuperPoint – are localized well. They can be more or less dense and/or evenly distributed over the image. For example, in image retrieval, one does not really care about precise localization, the robustness is much more important. In 3D reconstruction, it is beneficial to have a lot of features for better reconstruction. On the other hand, for the SLAM/relocalization application, sparse features would be more advantageous because of the smaller memory footprint and computational cost. There are actually some steps in that direction. Let me name a few.

1. HOW local features [262] are designed for image retrieval. Localization precision is not considered at all and repeated patterns, unlike for WxBS are encouraged, not discouraged, because they increase robustness of the representation. For the image retrieval it is not a problem, if one mismatch one window for almost identical neighboring window, as long as it helps finding the image.
2. SIPs [52] – as sparse as possible local features for the SLAM.
3. Reinforced Feature Points: Optimizing Feature Detection and Description for a High-Level Task [32] – finetuning the existing local feature with reinforcement learning for the downstream task.

I believe that it is only the beginning and we are yet to experience AlphaZero moment for the local features.

7.2.2 Rethinking the overall wide baseline stereo pipeline, optimized for specific applications

It is often perceived that image matching of an unordered collection of images is a task with quadratic complexity w.r.t. number of images. Some operations can be done separately, e.g. feature detection, but others, like feature matching and RANSAC cannot. However, it turns out that

the quadratic part can be done on the level of nearest neighbor search among global descriptors alone. This part scales very well [110]. The rest of the steps – feature matching and RANSAC can be avoided for more than 90% of image pairs with clever preprocessing, ordering, and re-using results from the previous matching. Moreover, to do the whole task faster (matching image collection), one may need to introduce additional steps, which are slowing things down for the two images case, e.g., calculating a global descriptor of the image. That is what we have done for the initial pose estimation for the global SfM [21], which reduced the matching runtime on 1dSfM dataset [280] from 200 hours to 29. Another example would be SuperGlue [216], where the authors abandoned traditional descriptor matching and instead leveraged all information about keypoints and descriptors from both images altogether processed by the graph neural network.

7.2.3 Rethinking and improving the process of training data generation for WxBS

So far, all local feature papers I have read rely on one of these sources of supervision:

1. SfM data, obtained using COLMAP with, possibly, a cleaned depth information.
2. Affine and color augmentation.
3. Synthetic images (e.g. corners for SuperPoint).

There are several problems with these sources of supervision.

SfM data assume that the data is matchable by existing methods, at least to an extent. That might not always be true for cross-seasonal, medical and other kinds of data. It is also not applicable for historical photography and other types of data. Moreover, SfM data takes a significant time to compute and the significant storage space.

Affine and color augmentation can take us only this far – we actually want our detectors and descriptors to be robust to the changes, which we do not know how to simulate/augment.

Synthetic images as in, for example, the CARLA [67] simulator, lack fine details and photorealism. However, I am optimistic about using neural renderers and learned wide multiple baseline stereo in a GAN-like self-improving loop, where WxBS is used for improving the rendering part, which is, in its turn, provides the WxBS with new, more challenging examples to train on.

What do I propose instead? The interesting directions to explore are the following. First – photorealistic renderings of the scenes, like the ones being used for object detection [98] and semantic segmentation [209]. Currently, they are not tuned to provide the information required for the wide baseline stereo. Second – probably more expensive – approach would be to use laser scans registered together with RGB images to get the perfect 3D reconstruction and use it for local features training. One of such examples is the Tanks and Temples dataset [118] currently mostly used for benchmarking and training geometry estimation models.

7.2.4 Matching with On-Demand View Synthesis revisited

I like the “on-demand” principle a lot and I think we can explore it much more than we are now. So far, we have either affine view synthesis (ASIFT [170], MODS [162]), or GAN-based stylizations [12] for the day-night matching. That is why I am glad to see papers like “Single-Image Depth Prediction Makes Feature Matching Easier” [258], which generate normalized views based on depth to help the matching. Why not go further? Combine viewpoint, illumination, season, and sensor synthesis?

7.2.5 More inputs!

I have mentioned above that monocular depth may help feature matching or camera pose estimation. However, why stop here? Let us use other networks as well, especially given that we will need them on the robot or vehicle anyway. Semantic segmentation? Yes, please. Surface normals? Why not? Intrinsic images? Let it be!

Bibliography

- [1] H. Aanaes, A.L. Dahl, and K. Steenstrup Pedersen. Interesting interest points. *International Journal of Computer Vision*, 97:18–35, 2012. [64](#), [87](#)
- [2] H. Aanaes and F. Kahl. Estimation of Deformable Structure and Motion. In *Vision and Modelling of Dynamic Scenes Workshop*, 2002. [101](#)
- [3] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M. Seitz, and Rick Szeliski. Building rome in a day. *Communications of the ACM*, 54:105–112, 2011. [16](#), [83](#), [86](#)
- [4] Cristhian Aguilera, Fernando Barrera, Felipe Lumbreras, Angel D. Sappa, and Ricardo Toledo. Multispectral image feature points. *Sensors*, 12(9):12661–12672, 2012. [24](#), [26](#), [28](#), [70](#), [73](#), [75](#)
- [5] Timo Ahonen, Jiří Matas, Chu He, and Matti Pietikäinen. Rotation invariant image description with local binary pattern histogram fourier features. In *Image Analysis*, pages 61–70, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. [46](#)
- [6] Alexandre Alahi, Raphaël Ortiz, and Pierre Vandergheynst. Freak: Fast retina keypoint. *IEEE Conference on Computer Vision and Pattern Recognition*, 2012. [26](#), [29](#), [93](#), [99](#)
- [7] P. F. Alcantarilla, J. Nuevo, and A. Bartoli. Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces. 2013. [26](#), [28](#), [86](#), [93](#), [99](#)
- [8] Javier Aldana-Iuit, Dmytro Mishkin, Ondrej Chum, and Jiri Matas. In the saddle: Chasing fast and repeatable features. In *23rd International Conference on Pattern Recognition, ICPR 2016, Cancún, Mexico, December 4-8, 2016*, pages 675–680, 2016. [21](#), [45](#)
- [9] Javier Aldana-Iuit, Dmytro Mishkin, Ondřej Chum, and Jiří Matas. Saddle: Fast and repeatable features with good coverage. *Image and Vision Computing*, 2019. [21](#), [45](#), [99](#)
- [10] Hani Altwaijry and Serge J. Belongie. Ultra-wide baseline aerial imagery matching in urban environments. In *British Machine Vision Conference, BMVC 2013, Bristol, UK, September 9-13, 2013*, 2013. [70](#), [73](#), [74](#), [75](#)
- [11] Anonymous. DeepSFM: Structure From Motion Via Deep Bundle Adjustment. In *Submission to ICLR*, 2020. [85](#)
- [12] Asha Anosheh, Torsten Sattler, Radu Timofte, Marc Pollefeys, and Luc Van Gool. Night-to-day image translation for retrieval-based localization. In *ICRA*, 2019. [125](#)
- [13] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN Architecture for Weakly Supervised Place Recognition. 2016. [83](#)
- [14] Relja Arandjelovic and Andrew Zisserman. Three things everyone should know to improve object retrieval. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2911–2918, 2012. [26](#), [29](#), [32](#), [36](#), [37](#), [59](#), [65](#), [93](#)
- [15] Hernan Badino, Daniel Huber, and Takeo Kanade. The CMU Visual Localization Data Set. <http://3dvis.ri.cmu.edu/data-sets/localization>, 2011. [87](#)

- [16] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. HPatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. [24](#), [26](#), [32](#), [36](#), [52](#), [55](#), [56](#), [57](#), [58](#), [78](#), [85](#), [87](#), [99](#), [102](#)
- [17] Vassileios Balntas, Shuda Li, and Victor Prisacariu. RelocNet: Continuous Metric Learning Relocalisation using Neural Nets. In *The European Conference on Computer Vision (ECCV)*, September 2018. [83](#)
- [18] Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *British Machine Vision Conference (BMVC)*, 2016. [29](#), [31](#), [32](#), [36](#), [49](#), [52](#), [59](#), [86](#), [93](#), [123](#)
- [19] Daniel Barath and Zuzana Kukelova. Homography from two orientation- and scale-covariant features. In *ICCV*, 2019. [48](#)
- [20] Daniel Barath and Jiří Matas. Graph-cut ransac. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. [47](#), [48](#), [86](#), [94](#)
- [21] Daniel Barath, Dmytro Mishkin, Ivan Eichhardt, Ilia Shipachev, and Jiri Matas. Efficient initial pose-graph generation for global sfm. In *CVPR*, 2020. [21](#), [125](#)
- [22] Daniel Barath, Michal Polic, Wolfgang Förstner, Torsten Sattler, Tomas Pajdla, and Zuzana Kukelova. Making affine correspondences work in camera geometry computation. *arXiv preprint arXiv:2007.10032*, 2020. [48](#)
- [23] Axel Barroso-Laguna, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Key.Net: Keypoint Detection by Handcrafted and Learned CNN Filters. In *ICCV*, 2019. [19](#), [46](#), [86](#), [93](#)
- [24] D. Baráth, I. Eichhardt, and L. Hajder. Optimal multi-view surface normal estimation using affine correspondences. *IEEE Transactions on Image Processing*, 28(7):3301–3311, 2019. [49](#), [50](#)
- [25] Adam Baumberg. Reliable feature matching across widely separated views. In *CVPR*, pages 1774–1781. IEEE Computer Society, 2000. [45](#), [56](#), [58](#), [93](#), [99](#), [100](#)
- [26] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *In ECCV*, pages 404–417, 2006. [28](#), [63](#), [86](#), [93](#)
- [27] Paul Beardsley, Phil Torr, and Andrew Zisserman. 3d model acquisition from extended image sequences. In *ECCV*, 1996. [16](#)
- [28] P. R. Beaudet. Rotationally invariant image operators. In *Proceedings of the 4th International Joint Conference on Pattern Recognition*, 1978. [16](#), [45](#), [46](#), [93](#), [99](#)
- [29] Jeffrey S. Beis and David G. Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *CVPR*, page 1000, USA, 1997. IEEE Computer Society. [16](#)
- [30] F. Bellavia and C. Colombo. Rethinking the sglh descriptor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):931–944, 2018. [46](#)
- [31] Fabio Bellavia and Carlo Colombo. Is there anything new to say about sift matching? *International Journal of Computer Vision*, pages 1–20, 2020. [46](#), [93](#)
- [32] A. Bhowmik, S. Gumhold, C. Rother, and E. Brachmann. Reinforced feature points: Optimizing feature detection and description for a high-level task. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. [124](#)
- [33] Jia-Wang Bian, Yu-Huan Wu, Ji Zhao, Yun Liu, Le Zhang, Ming-Ming Cheng, and Ian Reid. An Evaluation of Feature Matchers for Fundamental Matrix Estimation. 2019. [24](#), [87](#)
- [34] Eric Brachmann and Carsten Rother. Neural- Guided RANSAC: Learning Where to Sample Model Hypotheses. In *ICCV*, 2019. [85](#)

- [35] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 94
- [36] M. Brown, G. Hua, and S. Winder. Discriminative Learning of Local Image Descriptors. 2011. 83, 86, 102
- [37] M. Brown and D. Lowe. Invariant features from interest point groups. In *BMVC*, pages 23.1–23.10, 2002. 16
- [38] M. Brown and D. Lowe. Automatic Panoramic Image Stitching Using Invariant Features. *IJCV*, 74:59–73, 2007. 16, 26, 87
- [39] Matthew Brown and David G. Lowe. Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision (IJCV)*, 74(1):59–73, 2007. 31, 33, 36, 53
- [40] TB Brown, D Mane, A Roy, M Abadi, and J Gilmer. Adversarial patch. In *NeurIPS*, 2017. 18
- [41] Mai Bui, Christoph Baur, Nassir Navab, Slobodan Ilic, and Shadi Albarqouni. Adversarial Networks for Camera Pose Regression and Refinement. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2019. 83
- [42] Andrei Bursuc, Giorgos Tolias, and Herve Jegou. Kernel local descriptors with implicit rotation matching. In *ACM International Conference on Multimedia Retrieval*, 2015. 29
- [43] Michael Calonder, Vincent Lepetit, Mustafa Ozuysal, Tomasz Trzcinski, Christoph Strecha, and Pascal Fua. Brief: Computing a local binary descriptor very fast. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(7):1281–1298, July 2012. 29
- [44] Bingyi Cao, André Araujo, and Jack Sim. Unifying deep local and global features for image search. In *Computer Vision – ECCV 2020*, pages 726–743, 2020. 123
- [45] K. Chatfield, J. Philbin, and A. Zisserman. Efficient retrieval of deformable shape classes using local self-similarities. In *Workshop on Non-rigid Shape Analysis and Deformable Image Alignment, ICCV*, 2009. 26
- [46] Sunglok Choi, Taemin Kim, and Wonpil Yu. Performance evaluation of ransac family. In *BMVC*, 2009. 17
- [47] Christopher B. Choy, JunYoung Gwak, Silvio Savarese, and Manmohan Chandraker. Universal correspondence network. In *Advances in Neural Information Processing Systems*, pages 2414–2422, 2016. 29, 46, 51
- [48] O. Chum and A. Zisserman. An exemplar model for learning object classes. In *CVPR*, 2007. 17
- [49] Ondrej Chum and Jiri Matas. Matching with prosac – progressive sample consensus. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, pages 220–226, Washington, DC, USA, 2005. IEEE Computer Society. 17, 66, 86
- [50] Ondřej Chum, Jiří Matas, and Josef Kittler. Locally optimized ransac. In *Pattern Recognition*, 2003. 16, 66, 86, 94
- [51] Ondrej Chum, Tomas Werner, and Jiri Matas. Two-View Geometry Estimation Unaffected by a Dominant Plane. In *CVPR*, 2005. 17, 64, 66, 86, 94
- [52] Titus Cieslewski, Konstantinos G. Derpanis, and Davide Scaramuzza. Sips: Succinct interest points from unsupervised inlierness probability learning. In *3D Vision (3DV)*, 2019. 124
- [53] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *ICML*, 2016. 18

- [54] Kai Cordes, Bodo Rosenhahn, and Jörn Ostermann. Increasing the accuracy of feature evaluation benchmarks using differential evolution. In *IEEE Symposium Series on Computational Intelligence (SSCI) - IEEE Symposium on Differential Evolution (SDE)*, apr 2011. 70, 71, 72
- [55] Gabriela Csurka, Christopher R. Dance, and Martin Humenberger. From handcrafted to deep local features. *arXiv e-prints*, 2018. 16
- [56] Hainan Cui, Xiang Gao, Shuhan Shen, and Zhanyi Hu. Hsfm: Hybrid structure-from-motion. In *CVPR*, July 2017. 86
- [57] Anders Lindbjerg Dahl, Henrik Aanaes, and Kim Steenstrup Pedersen. Finding the best feature detector-descriptor combination. In *Proceedings of the 2011 International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission, 3DIMPVT '11*, pages 318–325, Washington, DC, USA, 2011. IEEE Computer Society. 75
- [58] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 17, 29, 73
- [59] Zheng Dang, Kwang Moo Yi, Yinlin Hu, Fei Wang, Pascal Fua, and Mathieu Salzmann. Eigendecomposition-Free Training of Deep Networks with Zero Eigenvalue-Based Losses. 2018. 93
- [60] Jana Noskova Daniel Barath, Jiri Matas. MAGSAC: marginalizing sample consensus. In *CVPR*, 2019. 47, 48, 84, 86, 94
- [61] D. Detone, T. Malisiewicz, and A. Rabinovich. Toward Geometric Deep SLAM. *arXiv preprint arXiv:1707.07410*, 2017. 83
- [62] D. Detone, T. Malisiewicz, and A. Rabinovich. Superpoint: Self-Supervised Interest Point Detection and Description. *CVPRW Deep Learning for Visual SLAM*, 2018. 19, 24, 46, 83, 86, 90, 91, 93, 104, 123, 124
- [63] Carl Doersch, Ankush Gupta, and Andrew Zisserman. Crosstransformers: spatially-aware few-shot transfer. *arXiv preprint arXiv:22007.11498*, 2020. 19
- [64] Jingming Dong, Nikolaos Karianakis, Damek Davis, Joshua Hernandez, Jonathan Balzer, and Stefano Soatto. Multi-view feature engineering and learning. In *CVPR*, June 2015. 116
- [65] Jingming Dong and Stefano Soatto. Domain-size pooling in local descriptors: DSP-SIFT. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5097–5106, 2015. 29, 101
- [66] Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin A. Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(9):1734–1747, 2016. 49
- [67] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, 2017. 125
- [68] Yves Dufournaud, Cordelia Schmid, and Radu Horaud. Matching Images with Different Resolutions. In *International Conference on Computer Vision & Pattern Recognition (CVPR '00)*, volume 1, pages 612–618, Hilton Head Island, United States, June 2000. IEEE Computer Society. 45
- [69] M. Dusmanu, I. Rocco, T. Pajdla, M. Pollefeys, J. Sivic, A. Torii, and T. Sattler. D2-Net: A Trainable CNN for Joint Detection and Description of Local Features. 2019. 19, 24, 84, 86, 92, 93, 104, 124
- [70] Patrick Ebel, Anastasiia Mishchuk, Kwang Moo Yi, Pascal Fua, and Eduard Trulls. Beyond Cartesian Representations for Local Descriptors. 2019. 19, 86, 92, 93, 101
- [71] Vassileios Balntas et.al. SILDa: A Multi-Task Dataset for Evaluating Visual Localization. <https://github.com/scape-research/silda>, 2018. 85, 87

- [72] Bin Fan, Fuchao Wu, and Zhanyi Hu. Aggregating gradient distributions into intensity orders: A novel local image descriptor. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, pages 2377–2384, 2011. 26, 29, 69
- [73] R. Fergus, P. Perona, and A. Zisserman. Object Class Recognition by Unsupervised Scale-Invariant Learning. In *CVPR*, pages 264–271, 2003. 17
- [74] Basura Fernando, Tatiana Tommasi, and Tinne Tuytelaars. Location recognition over large time lags. *Computer Vision and Image Understanding*, 139, 2015. 26, 27, 35, 57, 58
- [75] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, jun 1981. 16, 47, 83, 86, 94
- [76] W. Förstner, T. Dickscheid, and F. Schindler. Detecting interpretable and accurate scale-invariant keypoints. In *12th IEEE International Conference on Computer Vision (ICCV'09)*, Kyoto, Japan, 2009. 28
- [77] Wolfgang Förstner and Eberhard Gülch. A fast operator for detection and precise location of distinct points, corners and centres of circular features. In *Proc. ISPRS intercommission conference on fast processing of photogrammetric data*, pages 281–305, 1987. 16
- [78] C. Dance G. Csurka, J. Willamowski, L. Fan, and C. Bray. Visual Categorization with Bags of Keypoints. In *ECCV*, 2004. 17
- [79] Guillermo Gallego, Tobi Delbruck, Garrick Michael Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew Davison, Jorg Conradt, Kostas Daniilidis, and et al. Event-based vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 23
- [80] P. Gay, V. Bansal, C. Rubino, and A. D. Bue. Probabilistic Structure from Motion with Objects (PSfMO). In *ICCV*, 2017. 86
- [81] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. 2012. 85, 87
- [82] C. Godard, O. M. Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6602–6611, 2017. 19
- [83] Albert Gordo, Jon Almazan, Jerome Revaud, and Diane Larlus. End-to-end learning of deep visual representations for image retrieval. *IJCV*, 2017. 59, 60
- [84] Banglei Guan, Ji Zhao, Daniel Barath, and Friedrich Fraundorfer. Relative pose estimation for multi-camera systems from affine correspondences. *arXiv preprint arXiv:2007.10700*, 2020. 49
- [85] Xufeng Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In *CVPR*, 2015. 29, 31, 32, 49
- [86] M. J. Hannah. *Computer matching of areas in stereo images*. PhD thesis, 1974. 16
- [87] Jonathon S. Hare, Sina Samangoeei, and Paul H. Lewis. Efficient clustering and quantisation of sift features: Exploiting characteristics of the sift descriptor and interest region detectors under image inversion. In *ICMR 2011*, pages 2:1–2:8. ACM, 2011. 26
- [88] C.G. Harris and M.J. Stephens. A Combined Corner and Edge Detector. In *Fourth Alvey Vision Conference*, 1988. 16, 45, 46
- [89] R. I. Hartley. Projective reconstruction and invariants from multiple images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(10):1036–1041, Oct 1994. 83, 86
- [90] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004. 16, 23, 24, 25, 63, 75, 83, 94

- [91] R.I. Hartley. In Defense of the Eight-Point Algorithm. *PAMI*, 19(6):580–593, June 1997. 86
- [92] W. Hartmann, M. Havlena, and K. Schindler. Predicting matchability. In *CVPR*, pages 9–16, 2014. 49
- [93] Daniel C. Hauagge and Noah Snavely. Image matching using local symmetry features. In *Computer Vision and Pattern Recognition (CVPR)*, pages 206–213, 2012. 24, 26, 35, 57, 58, 70, 73, 75
- [94] K. He, Y. Lu, and S. Sclaroff. Local Descriptors Optimized for Average Precision. 2018. 83
- [95] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 39, 59
- [96] J. Heinly, J.L. Schoenberger, E. Dunn, and J-M. Frahm. Reconstructing the World in Six Days. 2015. 83, 86, 87
- [97] Geoffrey E Hinton, Alex Krizhevsky, and Sida D Wang. Transforming auto-encoders. In *ICANN*, 2011. 19
- [98] T. Hodaň, V. Vineet, R. Gal, E. Shalev, J. Hanzelka, T. Connell, P. Urbina, S. N. Sinha, and B. Guenter. Photorealistic image synthesis for object instance detection. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 66–70, 2019. 125
- [99] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In David Blei and Francis Bach, editors, *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 448–456. JMLR Workshop and Conference Proceedings, 2015. 31, 54
- [100] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, Teddy Furon, and Ondřej Chum. Efficient diffusion on region manifolds: Recovering small objects with compact cnn representations. In *CVPR*, 2017. 59, 60
- [101] Md Amirul Islam, Sen Jia, and Neil D. B. Bruce. How much position information do convolutional neural networks encode? In *ICLR*, 2020. 18
- [102] Daniel Barath Ivan Eichhardt. Relative pose from deep learned depth and a single affine correspondence. In *ECCV*, 2020. 49
- [103] N. Jacobs, N. Roman, and R. Pless. Consistent Temporal Variations in Many Outdoor Scenes. 2007. 24, 26, 85
- [104] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial Transformer Networks. *ArXiv e-prints*, June 2015. 53
- [105] Max Jaderberg, Karen Simonyan, and Andrew Zisserman. Spatial transformer networks. In *NeurIPS*, 2015. 18
- [106] Herve Jegou, Matthijs Douze, and Cordelia Schmid. On the burstiness of visual elements. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1169–1176, 2009. 37
- [107] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Improving bag-of-features for large scale image search. *International Journal of Computer Vision (IJCV)*, 87(3):316–336, 2010. 36, 59
- [108] Herve Jegou, Cordelia Schmid, Hedi Harzallah, and Jakob Verbeek. Accurate image search using the contextual dissimilarity measure. *Pattern Analysis and Machine Intelligence (PAMI)*, 32(1):2–11, 2010. 37
- [109] Yuhe Jin, Dmytro Mishkin, Anastasiia Mishchuk, Jiri Matas, Pascal Fua, Kwang Moo Yi, and Eduard Trulls. Image matching across wide baselines: From paper to practice. *International Journal of Computer Vision*, 2020. 16, 20, 21, 26, 42, 47, 78, 124
- [110] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*, 2017. 125

- [111] Christopher Kanan and Garrison W. Cottrell. Color-to-grayscale: Does the method matter in image recognition? *PLoS ONE*, 2012. [26](#)
- [112] Yan Ke and Rahul Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, CVPR'04, pages 506–513, Washington, DC, USA, 2004. IEEE Computer Society. [29](#)
- [113] Avi Kelman, Michal Sofka, and Charles V. Stewart. Keypoint descriptors for matching across multiple image modalities and non-linear intensity variations. In *CVPR*, 2007. [26](#), [70](#), [73](#), [75](#)
- [114] Avi Kelman, Michal Sofka, and Charles V Stewart. Keypoint descriptors for matching across multiple image modalities and non-linear intensity variations. In *CVPR 2007*, 2007. [57](#), [74](#)
- [115] Alex Kendall, Matthew Grimes, and Roberto Cipolla. PoseNet: A convolutional network for real-time 6-dof camera relocalization. In *ICCV*, 2015. [18](#), [83](#)
- [116] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. [52](#)
- [117] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 225–234, 2007. [16](#)
- [118] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Trans. Graph.*, 36(4), July 2017. [125](#)
- [119] J J Koenderink and A J van Doorn. Representation of local geometry in the visual system. *Biol. Cybern.*, 55(6):367–375, March 1987. [29](#)
- [120] J. Krishna Murthy, Ganesh Iyer, and Liam Paull. gradslam: Dense slam meets automatic differentiation. In *ICRA*, 2020. [19](#), [85](#)
- [121] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*. 2012. [17](#)
- [122] Vijay Kumar B. G., Gustavo Carneiro, and Ian Reid. Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5385–5394, 2016. [32](#)
- [123] Orest Kupyn, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiří Matas. Deblurgan: Blind motion deblurring using conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. [21](#)
- [124] S. Lazebnik, C. Schmid, and J. Ponce. A sparse texture representation using local affine regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1265–1278, 2005. [46](#)
- [125] S. Lazebnik, C. Schmid, and J. Ponce. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *CVPR*, 2006. [17](#)
- [126] Karel Lebeda, Jiří Matas, and Ondřej Chum. Fixing the locally optimized ransac. In *BMVC 2012*, 2012. [57](#), [70](#)
- [127] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *Artificial Intelligence and Statistics*, pages 562–570, 2015. [30](#)
- [128] K Lenc, V Gulshan, and A Vedaldi. Vlbenchmarks, 2012. [56](#), [87](#)
- [129] Karel Lenc, Jiri Matas, and Dmytro Mishkin. A few things one should know about feature extraction, description and matching. In *Proceedings of the Computer Vision Winter Workshop*, pages 67–74, 2014. [20](#), [21](#), [65](#), [72](#)

- [130] Karel Lenc and Andrea Vedaldi. Learning covariant feature detectors. In *ECCVW*, pages 100–117, 2016. [46](#), [49](#), [51](#)
- [131] Vincent Lepetit and Pascal Fua. Keypoint recognition using randomized trees. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(9), sep 2006. [62](#)
- [132] S. Leutenegger, M. Chli, and R. Y. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *ICCV*, pages 2548–2555, 2011. [93](#)
- [133] Jianguo Li, Mingjie Sun, and Changshui Zhang. Extreme values are accurate and robust in deep networks. 2020. [19](#)
- [134] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *CVPR*, 2018. [19](#), [87](#)
- [135] Tony Lindeberg. Detecting salient blob-like image structures and their scales with a scale-space primal sketch: A method for focus-of-attention. *Int. J. Comput. Vision*, 11(3):283–318, December 1993. [16](#), [45](#)
- [136] Tony Lindeberg. Feature detection with automatic scale selection. *Int. J. Comput. Vision*, 30(2):79–116, November 1998. [16](#), [45](#)
- [137] Tony Lindeberg. Scale-space theory in computer vision. 256, 2013. [16](#), [18](#)
- [138] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *ICLR*, 2019. [37](#)
- [139] Wei Liu, Yongtian Wang, Jing Chen, Junwei Guo, and Yang Lu. A completely affine invariant image-matching method based on perspective projection. *Mach. Vision Appl.*, 23(2):231–242, mar 2012. [63](#)
- [140] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004. [16](#), [26](#), [27](#), [29](#), [32](#), [45](#), [46](#), [52](#), [56](#), [57](#), [62](#), [65](#), [83](#), [84](#), [86](#), [90](#), [91](#), [93](#), [99](#), [107](#), [115](#), [123](#), [124](#)
- [141] D.G. Lowe. Object Recognition from Local Scale-Invariant Features. In *ICCV*, 1999. [16](#)
- [142] Zixin Luo, Tianwei Shen, Lei Zhou, Jiahui Zhang, Yao Yao, Shiwei Li, Tian Fang, and Long Quan. ContextDesc: Local Descriptor Augmentation with Cross-Modality Context. 2019. [86](#), [93](#)
- [143] Zixin Luo, Tianwei Shen, Lei Zhou, Siyu Zhu, Runze Zhang, Yao Yao, Tian Fang, and Long Quan. Geodesc: Learning local descriptors by integrating geometry constraints. In *ECCV*, 2018. [86](#), [93](#)
- [144] Simon Lynen, Bernhard Zeisl, Dror Aiger, Michael Bosse, Joel Hesch, Marc Pollefeys, Roland Siegwart, and Torsten Sattler. Large-scale, real-time visual-inertial localization revisited. 2019. [24](#), [83](#)
- [145] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km: The Oxford RobotCar dataset. 36(1):3–15, 2017. [87](#)
- [146] Andreas Madsen and Alexander Rosenberg Johansen. Neural arithmetic units. In *ICLR*, 2020. [18](#)
- [147] F. Maiwald. Generation of a benchmark dataset using historical photographs for an automated evaluation of different feature matching methods. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-2/W13:87–94, 2019. [62](#)
- [148] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extrema regions. In *BMVC*, pages 384–393, 2002. [26](#), [27](#), [34](#), [45](#), [62](#), [93](#), [99](#)
- [149] Iaroslav Melekhov, Gabriel J. Brostow, Juho Kannala, and Daniyar Turmukhambetov. Image stylization for robust features. *arXiv ePrint 2008.06959*, 2020. [24](#)

- [150] Iaroslav Melekhov, Juha Ylioinas, Juho Kannala, and Esa Rahtu. Relative camera pose estimation using convolutional neural networks. 2017. [18](#)
- [151] K. Mikolajczyk and C. Schmid. A Performance Evaluation of Local Descriptors. In *CVPR*, pages 257–263, June 2003. [16](#), [26](#), [85](#), [86](#), [99](#), [124](#)
- [152] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A Comparison of Affine Region Detectors. *IJCV*, 65(1/2):43–72, 2005. [16](#), [64](#), [70](#), [72](#), [124](#)
- [153] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision (IJCV)*, 60(1):63–86, 2004. [26](#), [28](#), [34](#), [45](#), [93](#), [99](#)
- [154] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision (IJCV)*, 60(1):63–86, 2004. [35](#), [58](#)
- [155] Krystian Mikolajczyk, Tinne Tuytelaars, Cordelia Schmid, Andrew Zisserman, Jiri Matas, Frederik Schaffalitzky, Timor Kadir, and Luc Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision (IJCV)*, 65(1):43–72, 2005. [24](#), [26](#), [36](#), [47](#), [56](#), [58](#), [70](#), [71](#)
- [156] Andrej Mikulik, Michal Perdoch, Ondřej Chum, and Jiří Matas. Learning vocabularies over a fine quantization. *International Journal of Computer Vision (IJCV)*, 103(1):163–175, 2013. [36](#), [59](#)
- [157] Andrej Mikulik, Filip Radenovic, Ondrej Chum, and Jiri Matas. Efficient image detail mining. In *ACCV*, 2014. [70](#), [72](#)
- [158] A. Mishchuk, D. Mishkin, F. Radenovic, and J. Matas. Working Hard to Know Your Neighbor’s Margins: Local Descriptor Learning Loss. In *NeurIPS*, 2017. [19](#), [20](#), [21](#), [33](#), [46](#), [49](#), [51](#), [52](#), [53](#), [54](#), [57](#), [59](#), [86](#), [90](#), [93](#), [101](#), [115](#), [123](#)
- [159] D. Mishkin and J. Matas. All you need is a good init. In *Proceedings of ICLR*, may 2016. [21](#)
- [160] D. Mishkin, M. Perdoch, and J. Matas. Place recognition with wxbs retrieval. jun 2015. [21](#)
- [161] D. Mishkin, F. Radenovic, and J. Matas. Repeatability is Not Enough: Learning Affine Regions via Discriminability. In *ECCV*, 2018. [20](#), [21](#), [46](#), [62](#), [93](#), [100](#), [101](#), [123](#)
- [162] Dmytro Mishkin, Jiri Matas, and Michal Perdoch. Mods: Fast and robust method for two-view matching. *Computer Vision and Image Understanding*, 141:81 – 93, 2015. [20](#), [35](#), [58](#), [61](#), [99](#), [115](#), [125](#)
- [163] Dmytro Mishkin, Jiri Matas, Michal Perdoch, and Karel Lenc. Wxbs: Wide baseline stereo generalizations. In *BMVC*, 2015. [20](#), [21](#), [34](#), [35](#), [57](#), [58](#)
- [164] Dmytro Mishkin, Michal Perdoch, and Jiri Matas. Two-view matching with view synthesis revisited. In *Image and Vision Computing New Zealand (IVCNZ), 2013 28th International Conference of*, pages 436–441, nov 2013. [20](#), [21](#), [26](#), [27](#), [65](#), [70](#), [75](#)
- [165] Dmytro Mishkin, Nikolay Sergievskiy, and Jiri Matas. Systematic evaluation of convolution neural network advances on the imagenet. *Computer Vision and Image Understanding*, 161:11 – 19, 2017. [21](#), [31](#), [54](#)
- [166] R. Mitra, N. Doiphode, U. Gautam, S. Narayan, S. Ahmed, S. Chandran, and A. Jain. A Large Dataset for Improving Patch Matching. *ArXiv e-prints*, January 2018. [33](#)
- [167] Lionel Moisan and Bérenger Stival. A probabilistic criterion to detect rigid point matches between two images and estimate the fundamental matrix. *Int. J. Comput. Vision*, 57(3):201–218, May 2004. [63](#)
- [168] Hans Peter Moravec. *Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover*. PhD thesis, 1980. [16](#)

- [169] Pierre Moreels and Pietro Perona. Evaluation of features detectors and descriptors based on 3d objects. *International Journal of Computer Vision*, 73(3):263–284, 2007. [70](#), [75](#), [76](#), [78](#), [80](#)
- [170] Jean-Michel Morel and Guoshen Yu. Asift: A new framework for fully affine invariant image comparison. *SIAM J. Img. Sci.*, 2(2), apr 2009. [24](#), [62](#), [63](#), [70](#), [72](#), [74](#), [125](#)
- [171] Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISAPP*, 2009. [27](#), [29](#), [36](#), [59](#), [70](#), [108](#)
- [172] Arun Mukundan, Giorgos Tolias, Andrei Bursuc, Hervé Jégou, and Ondřej Chum. Understanding and improving kernel local descriptors. *IJCV*, 2018. [93](#)
- [173] Arun Mukundan, Giorgos Tolias, and Ondrej Chum. Multiple-kernel local-patch descriptor. In *British Machine Vision Conference*, 2017. [32](#)
- [174] Arun Mukundan, Giorgos Tolias, and Ondrej Chum. Explicit Spatial Encoding for Deep Local Descriptors. 2019. [83](#)
- [175] Sander Münster, Ferdinand Maiwald, Christoph Lehmann, Taras Lazariv, Mathias Hofmann, and Florian Niebling. An automated pipeline for a browser-based, city-scale mobile 4d vr application based on historical images. In *Proceedings of the 2nd Workshop on Structuring and Understanding of Multimedia HeritAge Contents*, page 33–40, 2020. [61](#), [62](#)
- [176] R. Mur-Artal, J. Montiel, and J. Tardós. ORB-Slam: A Versatile and Accurate Monocular Slam System. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015. [16](#), [83](#)
- [177] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning (ICML)*, pages 807–814, 2010. [31](#), [54](#)
- [178] D. Nister. An Efficient Solution to the Five-Point Relative Pose Problem. June 2003. [86](#)
- [179] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Large-scale image retrieval with attentive deep local features. In *CVPR*, pages 3456–3465, 2017. [16](#), [60](#), [83](#), [86](#), [123](#), [124](#)
- [180] T. Ojala, M. Pietikainen, and D. Harwood. Performance evaluation of texture measures with classification based on Kullback discrimination of distributions. In *Pattern Recognition, 1994. Vol. 1 - Conference A: Computer Vision amp; Image Processing., Proceedings of the 12th IAPR International Conference on*, volume 1, October 1994. [29](#)
- [181] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. LF-Net: Learning Local Features from Images. 2018. [19](#), [86](#), [92](#), [93](#), [123](#)
- [182] Yanwei Pang, Wei Li, Yuan Yuan, and Jing Pan. Fully affine invariant surf for image matching. *Neurocomputing*, 85(0):6 – 10, 2012. [63](#)
- [183] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017. [31](#), [54](#)
- [184] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. [94](#)
- [185] M. Perd’och, O. Chum, and J. Matas. Efficient representation of local geometry for large scale object retrieval. In *CVPR*, 2009. [28](#), [34](#), [36](#), [46](#), [57](#), [58](#)
- [186] Michal Perdoch, Ondrej Chum, and Jiri Matas. Efficient representation of local geometry for large scale object retrieval. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9–16, 2009. [124](#)

- [187] Michal Perd'och, Jiri Matas, and Ondrej Chum. Epipolar geometry from two correspondences. In *ICPR*, 2006. 48
- [188] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object Retrieval with Large Vocabularies and Fast Spatial Matching. In *CVPR*, 2007. 17, 35, 36, 48, 49, 58, 59, 67, 124
- [189] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008. 35, 36, 37, 58, 59
- [190] Noé Pion, Martin Humenberger, Gabriela Csurka, Yohann Cabon, and Torsten Sattler. Benchmarking image retrieval for visual localization. In *3DV*, 2020. 16, 18
- [191] P. Pritchett and A. Zisserman. "matching and reconstruction from widely separated views". In *3D Structure from Multiple Images of Large-Scale Environments*, pages 78–92, 1998. 16
- [192] P. Pritchett and A. Zisserman. Wide baseline stereo matching. In *ICCV*, pages 754–760, 1998. 16, 83
- [193] J. Pritts, O. Chum, and J. Matas. Approximate models for fast and accurate epipolar geometry estimation. In *2013 28th International Conference on Image and Vision Computing New Zealand (IVCNZ 2013)*, 2013. 48
- [194] James Pritts, Zuzana Kukelova, Viktor Larsson, and Ondřej Chum. Rectification from radially-distorted scales. In C.V. Jawahar, Hongdong Li, Greg Mori, and Konrad Schindler, editors, *ACCV 2018*, pages 36–52. 49
- [195] James Pritts, Zuzana Kukelova, Viktor Larsson, and Ondrej Chum. Radially-distorted conjugate translations. In *CVPR*, 2018. 49
- [196] James Pritts, Zuzana Kukelova, Viktor Larsson, Yaroslava Lochman, and Ondrej Chum. Minimal solvers for rectifying from radially-distorted scales and change of scales. *Int. J. Comput. Vis.*, 128(4):950–968, 2020. 49
- [197] James Pritts, Zuzana Kukelova, Viktor Larsson, Yaroslava Lochman, and Ondřej Chum. Minimal solvers for rectifying from radially-distorted conjugate translations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 49, 50
- [198] M. Pultar, D. Mishkin, and J. Matas. Leveraging Outdoor Webcams for Local Descriptor Learning. In *Computer Vision Winter Workshop*, 2019. 21, 37, 42, 87, 102
- [199] Milan Pultar. Improving the hardnet descriptor. *arXiv ePrint 2007.09699*, 2020. 9, 21, 29, 37, 123
- [200] C.R. Qi, H. Su, K. Mo, and L.J. Guibas. Pointnet: Deep Learning on Point Sets for 3D Classification and Segmentation. 2017. 93
- [201] Filip Radenović, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum. Revisiting oxford and paris: Large-scale image retrieval benchmarking. In *CVPR*, 2018. 58, 60
- [202] Filip Radenovic, Johannes L Schonberger, Dinghuang Ji, Jan-Michael Frahm, Ondrej Chum, and Jiri Matas. From dusk till dawn: Modeling in the dark. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5488–5496, 2016. 24
- [203] Filip Radenovic, Giorgos Tolias, and Ondrej Chum. CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples. In *European Conference on Computer Vision (ECCV)*, pages 3–20, 2016. 19, 83
- [204] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning cnn image retrieval with no human annotation. *arXiv:1711.02512*, 2017. 19, 59, 60
- [205] R. Ranftl and V. Koltun. Deep Fundamental Matrix Estimation. 2018. 85, 86, 92, 93

- [206] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: An astounding baseline for recognition. In *CVPRW*, 2014. [17](#), [18](#)
- [207] Jérôme Revaud, Philippe Weinzaepfel, César Roberto de Souza, Noe Pion, Gabriela Csurka, Yohann Cabon, and Martin Humenberger. R2D2: Repeatable and Reliable Detector and Descriptor. 2019. [19](#), [24](#), [46](#), [86](#), [90](#), [93](#), [104](#), [123](#)
- [208] Edgar Riba, Dmytro Mishkin, Daniel Ponsa, Ethan Rublee, and Gary Bradski. Kornia: an open source differentiable computer vision library for pytorch. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, March 2020. [21](#), [93](#)
- [209] Mike Roberts and Nathan Paczan. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. arXiv 2020. [125](#)
- [210] M. Rodríguez, G. Facciolo, R. G. von Gioi, P. Musé, and J. Delon. Robust estimation of local affine maps and its applications to image matching. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020. [48](#)
- [211] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *Proceedings of the 9th European Conference on Computer Vision - Volume Part I, ECCV'06*, pages 430–443, Berlin, Heidelberg, 2006. Springer-Verlag. [45](#), [95](#)
- [212] E. Rublee, V. Rabaud, K. Konolidge, and G. Bradski. ORB: An Efficient Alternative to SIFT or SURF. In *ICCV*, 2011. [26](#), [29](#), [45](#), [46](#), [65](#), [86](#), [93](#), [99](#), [124](#)
- [213] M. Brown S. Winder. Photo tourism dataset. [38](#)
- [214] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *NeurIPS*, pages 3856–3866, 2017. [19](#)
- [215] Samuele Salti, Alessandro Lanza, and Luigi Di Stefano. Keypoints from symmetries by wave propagation. In *CVPR 2013*, pages 2898–2905, 2013. [28](#)
- [216] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Super-glue: Learning feature matching with graph neural networks. In *CVPR*, 2020. [19](#), [92](#), [123](#), [125](#)
- [217] T. Sattler, W. Maddern, C. Toft, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, F. Kahl, and T. Pajdla. Benchmarking 6DOF Outdoor Visual Localization in Changing Conditions. 2018. [83](#), [87](#)
- [218] T. Sattler, W. Maddern, A. Torii, J. Sivic, T. Pajdla, M. Pollefeys, and M. Okutomi. Benchmarking 6DOF Urban Visual Localization in Changing Conditions. *ArXiv e-prints*, July 2017. [24](#)
- [219] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Improving Image-Based Localization by Active Correspondence Search. In *ECCV*, 2012. [83](#), [85](#), [87](#), [99](#)
- [220] Torsten Sattler, Qunjie Zhou, Marc Pollefeys, and Laura Leal-Taixe. Understanding the limitations of cnn-based absolute camera pose regression. In *CVPR*, 2019. [18](#), [83](#)
- [221] N. Savinov, A. Seki, L. Ladicky, T. Sattler, and M. Pollefeys. Quad-Networks: Unsupervised Learning to Rank for Interest Point Detection. 2017. [46](#), [49](#), [51](#), [86](#)
- [222] Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *Proceedings of ICLR*, 2014. [31](#)
- [223] Bernt Schiele and James L. Crowley. Object recognition using multidimensional receptive field histograms. In *Proceedings of the 4th European Conference on Computer Vision-Volume I - Volume I, ECCV '96*, pages 610–619, London, UK, UK, 1996. Springer-Verlag. [29](#)
- [224] Cordelia Schmid and Roger Mohr. Matching by local invariants. 1995. [16](#)

- [225] Cordelia Schmid and Roger Mohr. Local grayvalue invariants for image retrieval. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(5):530–535, May 1997. 29
- [226] J.L. Schönberger and J.M. Frahm. Structure-From-Motion Revisited. In *CVPR*, 2016. 16, 17, 83, 86, 87, 94, 99
- [227] J.L. Schönberger, H. Hardmeier, T. Sattler, and M. Pollefeys. Comparative Evaluation of Hand-Crafted and Learned Local Features. In *CVPR*, 2017. 87
- [228] J.L. Schönberger, E. Zheng, M. Pollefeys, and J.M. Frahm. Pixelwise View Selection for Unstructured Multi-View Stereo. 2016. 87
- [229] Johannes L. Schonberger, Filip Radenovic, Ondrej Chum, and Jan-Michael Frahm. From single image query to detailed 3d reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 62
- [230] S. Se, D. G. Lowe, and J. Little. Mobile Robot Localization and Mapping with Uncertainty Using Scale-Invariant Visual Landmarks. *IJRR*, 22(8):735–758, 2002. 16
- [231] Eli Shechtman and Michal Irani. Matching local self-similarities across images and videos. In *CVPR 2007*, 2007. 26
- [232] Tianwei Shen, Zixin Luo, Lei Zhou, Runze Zhang, Siyu Zhu, Tian Fang, and Long Quan. Matchable image retrieval by learning from surface reconstruction. In *ACCV*, 2018. 42
- [233] Yunxiao Shi, Jing Zhu, Yi Fang, Kuochin Lien, and Junli Gu. Self-Supervised Learning of Depth and Ego-motion with Differentiable Bundle Adjustment. 2019. 85
- [234] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *ICCV*, 2015. 29, 86
- [235] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Descriptor learning using convex optimisation. In *European Conference on Computer Vision (ECCV)*, pages 243–256, 2012. 29
- [236] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale visual recognition. In *Proceedings of ICLR*, May 2015. 38
- [237] J. Sivic and A. Zisserman. Video Google: A Text Retrieval Approach to Object Matching in Videos. In *ICCV*, 2003. 17
- [238] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *International Conference on Computer Vision (ICCV)*, pages 1470–1477, 2003. 36, 59
- [239] Stephen M. Smith and J. Michael Brady. Susan—a new approach to low level image processing. *International Journal of Computer Vision*, 23(1):45–78, May 1997. 45
- [240] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: Exploring photo collections in 3d. *ToG*, 25(3):835–846, 2006. 16
- [241] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research (JMLR)*, 15(1):1929–1958, 2014. 31, 54
- [242] C. V. Stewart, Chia-Ling Tsai, and B. Roysam. The dual-bootstrap iterative closest point algorithm with application to retinal image registration. *IEEE Transactions on Medical Imaging*, 22(11):1379–1394, 2003. 16
- [243] C. Strecha, W.V. Hansen, L. Van Gool, P. Fua, and U. Thoennessen. On Benchmarking Camera Calibration and Multi-View Stereo for High Resolution Imagery. 2008. 85, 87
- [244] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A Benchmark for the Evaluation of RGB-D SLAM Systems. 2012. 95

- [245] A. Stylianou, A. Abrams, and R. Pless. Characterizing feature matching performance over long time periods. In *WACV*, pages 892–898, 2015. 27
- [246] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019. 19
- [247] Niko Suenderhauf and Arren Glover. The vprice challenge 2015: Visual place recognition in changing environments, 2015. 26
- [248] Weiwei Sun, Wei Jiang, Eduard Trulls, Andrea Tagliasacchi, and Kwang Moo Yi. ACNe: Attentive Context Normalization for Robust Permutation-Equivariant Learning. 2020. 19, 85, 86, 93, 104
- [249] Michael J. Swain and Dana H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, Nov 1991. 29
- [250] Hajime Taira, Masatoshi Okutomi, Torsten Sattler, Mircea Cimpoi, Marc Pollefeys, Josef Sivic, Tomas Pajdla, and Akihiko Torii. InLoc: Indoor Visual Localization with Dense Matching and View Synthesis. 2019. 99
- [251] Chengzhou Tang and Ping Tan. Ba-Net: Dense Bundle Adjustment Network. 2019. 85
- [252] Maxim Tatarchenko, Stephan R. Richter, Rene Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3d reconstruction networks learn? In *CVPR*, 2019. 18
- [253] Keisuke Tateno, Federico Tombari, Iro Laina, and Nassir Navab. Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *CVPR*, July 2017. 85
- [254] B. Thomee, D.A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L. Li. YFCC100M: the New Data in Multimedia Research. In *Communications of the ACM*, 2016. 87
- [255] Yuandong Tian, Xinlei Chen, and Surya Ganguli. Understanding self-supervised learning dynamics without contrastive pairs. *arXiv e-prints 2102.06810*, 2021. 123
- [256] Yurun Tian, Xin Yu, Bin Fan, Fuchao Wu, Huub Heijnen, and Vassileios Balntas. Sosnet: Second order similarity regularization for local descriptor learning. In *CVPR*, 2019. 83, 86, 93
- [257] Joseph Tighe and Svetlana Lazebnik. Superparsing: Scalable nonparametric image parsing with superpixels. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *ECCV*, 2010. 17
- [258] Carl Toft, Daniyar Turmukhambetov, Torsten Sattler, Fredrik Kahl, and Gabriel J. Brostow. Single-image depth prediction makes feature matching easier. In *European Conference on Computer Vision (ECCV)*, 2020. 125
- [259] E. Tola, V. Lepetit, and P. Fua. DAISY: An Efficient Dense Descriptor Applied to Wide-Baseline Stereo. *PAMI 2010*, 32(5):815–830, 2010. 26
- [260] Giorgos Tolias, Yannis Avrithis, and Hervé Jégou. Image Search with Selective Match Kernels: Aggregation Across Single and Multiple Images. *IJCV*, 116(3):247–261, Feb 2016. 83
- [261] Giorgos Tolias and Herve Jegou. Visual query expansion with or without geometry: refining local descriptors by feature aggregation. *Pattern Recognition*, 47(10):3466–3476, 2014. 36, 37, 59, 60
- [262] Giorgos Tolias, Tomas Jenicek, and Ondřej Chum. Learning and aggregating deep local descriptors for instance-level recognition. In *ECCV 2020*, pages 460–477, 2020. 124
- [263] A. Torii, R. Arandjelović, J. Sivic, M. Okutomi, and T. Pajdla. 24/7 place recognition by view synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. 24

- [264] P.H.S. Torr and A. Zisserman. Feature Based Methods for Structure and Motion Estimation. In *Workshop on Vision Algorithms*, 1999. [16](#)
- [265] P.H.S. Torr and A. Zisserman. MLESAC: A New Robust Estimator with Application to Estimating Image Geometry. *CVIU*, 78:138–156, 2000. [86](#)
- [266] Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Herve Jegou. Fixing the train-test resolution discrepancy. In *NeurIPS*, volume 32, pages 8252–8262. Curran Associates, Inc., 2019. [123](#)
- [267] Andrew Trask, Felix Hill, Scott E Reed, Jack Rae, Chris Dyer, and Phil Blunsom. Neural arithmetic logic units. In *NeurIPS*, 2018. [18](#)
- [268] B. Triggs, P. Mclauchlan, R. Hartley, and A. Fitzgibbon. Bundle Adjustment – A Modern Synthesis. In *Vision Algorithms: Theory and Practice*, pages 298–372, 2000. [83](#)
- [269] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *J. Cognitive Neuroscience*, 3(1):71–86, January 1991. [29](#)
- [270] Tinne Tuytelaars and Krystian Mikolajczyk. Local invariant feature detectors: A survey. *Found. Trends. Comput. Graph. Vis.*, 3(3):177–280, July 2008. [16](#), [45](#)
- [271] Michał J. Tyszkiewicz, Pascal Fua, and Eduard Trulls. Disk: Learning local features with policy gradient. In *NeurIPS*, 2020. [123](#)
- [272] C. Varytimidis, K. Rapantzikos, and Y. Avrithis. Wash: Weighted α -shapes for local feature detection. In *ECCV 2012*, 2012. [28](#), [45](#)
- [273] Andrea Vedaldi and Brian Fulkerson. Vlfeat: An open and portable library of computer vision algorithms. In *Proceedings of the 18th ACM International Conference on Multimedia*, MM '10, pages 1469–1472, 2010. [93](#)
- [274] Yannick Verdie, Kwang Yi, Pascal Fua, and Vincent Lepetit. Tilde: a temporally invariant learned detector. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5279–5288, 2015. [46](#), [49](#), [51](#), [85](#), [86](#), [87](#)
- [275] S. Vijayanarasimhan, S. Ricco, C. Schmid, R. Sukthankar, and K. Fragkiadaki. Sfm-Net: Learning of Structure and Motion from Video. 2017. [85](#)
- [276] Felix von Hundelshausen. D-nets: Beyond patch-based image descriptors. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '12, pages 2941–2948, Washington, DC, USA, 2012. IEEE Computer Society. [73](#)
- [277] Vasillios Vonikakis, Dimitrios Chrysostomou, Rigas Kouskouridas, and Antonios Gasteratos. A biologically inspired scale-space for illumination invariant feature detection. *Measurement Science and Technology*, 2013. [24](#), [28](#)
- [278] Qianqian Wang, Xiaowei Zhou, Bharath Hariharan, and Noah Snavely. Learning feature descriptors using camera pose supervision. In *ECCV*, 2020. [19](#)
- [279] X. Wei, Y. Zhang, Y. Gong, and N. Zheng. Kernelized Subspace Pooling for Deep Local Descriptors. 2018. [83](#)
- [280] Kyle Wilson and Noah Snavely. Robust global translations with 1dsfm. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014. [125](#)
- [281] Changchang Wu. Towards Linear-Time Incremental Structure from Motion. In *3DV*, 2013. [86](#), [99](#)
- [282] Gehua Yang, Charles V Stewart, Michał Sofka, and Chia-Ling Tsai. Registration of challenging image pairs: Initialization, estimation, and decision. *Pattern Analysis and Machine Intelligence (PAMI)*, 29(11):1973–1989, 2007. [35](#), [57](#), [58](#), [72](#), [74](#)
- [283] Y. Yao, Y. Jafarian, and H. S. Park. Monet: Multiview semi-supervised keypoint detection via epipolar divergence. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 753–762, 2019. [19](#)

- [284] K. M. Yi, E. Trulls, Y. Ono, V. Lepetit, M. Salzmann, and P. Fua. Learning to Find Good Correspondences. 2018. [19](#), [85](#), [86](#), [87](#), [89](#), [92](#), [93](#), [94](#), [104](#)
- [285] K. M. Yi, Y. Verdie, P. Fua, and V. Lepetit. Learning to Assign Orientations to Feature Points. In *CVPR*, 2016. [45](#), [46](#), [49](#), [51](#), [52](#), [54](#)
- [286] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. LIFT: Learned invariant feature transform. In *European Conference on Computer Vision (ECCV)*, pages 467–483, 2016. [24](#), [46](#), [51](#), [86](#)
- [287] Andy B Yoo, Morris A Jette, and Mark Grondona. Slurm: Simple linux utility for resource management. In *Workshop on Job Scheduling Strategies for Parallel Processing*, pages 44–60. Springer, 2003. [95](#)
- [288] Fuchao Wu Yurun Tian, Bin Fan. L2-net: Deep learning of discriminative patch descriptor in euclidean space. In *CVPR*, 2017. [30](#), [31](#), [32](#), [34](#), [36](#), [49](#), [86](#), [93](#)
- [289] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *CVPR*, 2015. [29](#), [32](#), [49](#), [101](#)
- [290] Jiahui Zhang, Dawei Sun, Zixin Luo, Anbang Yao, Lei Zhou, Tianwei Shen, Yurong Chen, Long Quan, and Hongen Liao. Learning Two-View Correspondences and Geometry Using Order-Aware Network. 2019. [85](#), [86](#), [89](#), [93](#)
- [291] Jiahui Zhang, Dawei Sun, Zixin Luo, Anbang Yao, Lei Zhou, Tianwei Shen, Yurong Chen, Long Quan, and Hongen Liao. Learning two-view correspondences and geometry using order-aware network. In *ICCV*, 2019. [92](#)
- [292] Richard Zhang. Making convolutional networks shift-invariant again. In *ICML*, 2019. [18](#)
- [293] Xu Zhang, Felix Yu, Svebor Karaman, and Shih-Fu Chang. Learning discriminative and transformation covariant local feature detectors. In *CVPR*, 2017. [46](#), [49](#), [51](#), [86](#)
- [294] Xu Zhang, Felix X. Yu, Sanjiv Kumar, and Shih-Fu Chang. Learning spread-out local feature descriptors. In *ICCV*, 2017. [49](#)
- [295] Chen Zhao, Zhiguo Cao, Chi Li, Xin Li, and Jiaqi Yang. NM-Net: Mining Reliable Neighbors for Robust Feature Correspondences. 2019. [86](#), [93](#)
- [296] Bin Fan Zhenhua Wang and Fuchao Wu. Local intensity order pattern for feature description. In *IEEE International Conference on Computer Vision (ICCV)*, pages 603–610, Nov. 2011. [26](#), [29](#), [69](#)
- [297] Brady Zhou, Philipp Krähenbühl, and Vladlen Koltun. Does computer vision matter for action? *Science Robotics*, 4(30), 2019. [19](#)
- [298] Qunjie Zhou, Torsten Sattler, Marc Pollefeys, and Laura Leal-Taixe. To learn or not to learn: Visual localization from essential matrices. In *ICRA*, 2020. [18](#), [83](#)
- [299] Siyu Zhu, Runze Zhang, Lei Zhou, Tianwei Shen, Tian Fang, Ping Tan, and Long Quan. Very Large-Scale Global SfM by Distributed Motion Averaging. June 2018. [83](#), [86](#)
- [300] C. Lawrence Zitnick and Krishnan Ramnath. Edge foci interest points. In *International Conference on Computer Vision (ICCV)*, pages 359–366, 2011. [26](#), [28](#), [34](#), [35](#), [45](#), [57](#), [58](#), [85](#), [87](#)
- [301] Barbara Zitova and Jan Flusser. Image registration methods: a survey. *Image and Vision Computing*, 2003. [24](#)
- [302] Karel Zuiderveld. Graphics gems iv. chapter Contrast Limited Adaptive Histogram Equalization, pages 474–485. Academic Press Professional, Inc., San Diego, CA, USA, 1994. [116](#)