Robust, refined and selective matching for accurate camera pose estimation

Zhe Liu

LABORATOIRE D'INFORMATIQUE
GASPARD MONGE

Sous la co-tutelle de :
CNRS
ÉCOLE DES PONTS PARISTECH
ESIEE PARIS
UPEM • UNIVERSITÉ PARIS-EST MARNE-LA-VALLÉE

UNIVERSITÉ
PARIS-EST
MARNE-LA-VALLÉE

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

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PARIS EST
Domaine : Traitement du Signal et des Images

présentée par **Zhe LIU**
pour obtenir le grade de
DOCTEUR DE L'UNIVERSITÉ PARIS EST

Robust, refined and selective matching for accurate camera pose estimation

---

Soutenue publiquement le 13 avril 2015 devant le jury composé de :

Fredrik KAHL	Lund University	Examinateur
Renaud KERIVEN	Acute3D	Examinateur
Renaud MARLET	École des Ponts ParisTech	Directeur de thèse
Lionel MOISAN	Université Paris Descartes	Rapporteur
Pascal MONASSE	École des Ponts ParisTech	Co-directeur de thèse
Tomas PAJDLA	Czech Technical University	Rapporteur

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

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PARIS EST
Domaine : Traitement du Signal et des Images

présentée par **Zhe LIU**
pour obtenir le grade de
DOCTEUR DE L'UNIVERSITÉ PARIS EST

Robust, refined and selective matching for accurate camera pose estimation

---

École des Ponts ParisTech
LIGM-IMAGINE
6, Av Blaise Pascal - Cité Descartes
Champs-sur-Marne
77455 Marne-la-Vallée cedex 2
France

Université Paris-Est Marne-la-Vallée
École Doctorale Paris-Est MSTIC
Département Études Doctorales
6, Av Blaise Pascal - Cité Descartes
Champs-sur-Marne
77454 Marne-la-Vallée cedex 2
France

Acknowledgments

Thesis defense committee: I express my warm thanks to Fredrik Kahl, Renaud Keriven, Lionel Moisan and Tomas Pajdla for accepting the invitation to be part of my PhD thesis defense committee. In particular I want to thank Lionel Moisan and Tomas Pajdla who, as official reviewers, had a significant volume of work.

Supervisor: I'm using this opportunity to express my gratitude to my supervisors Renaud Marlet and Pascal Monasse for their patient support throughout my PhD in academic area and in life convenience. I have never stopped learning new things from them during discussions, which were great experiences. Their knowledge and help have guided me to discover the world of research, and inspired me toward the right direction. I now understand why the tracks of unsuccessful experiments are as important as the successful ones.

Colleague: I'm thankful for the very comfortable surroundings between colleagues in the laboratory. I enjoyed a lot the candies left in the coffee room and the delicious cakes! Your varied knowledge and cooperative work has been precious for me: Guillaume Obozinski, Nikos Komodakis, Sergey Zagoruyko, Raghudeep Gadde and Francisco Suzano in machine learning, Yohann Salaun for the work in line segment matching, Amine Bourki concerning super-pixels and vanishing points, Mateusz Kozinski in facade grammar, Martin De la Gorce and Alexandre Boulc'h in point cloud semantisation, Laura Fernandez in disparity maps, Wenbin Zhou in cooking etc. . . as well as moments with Chaohui Wang, Arnak Dalalyan, Gabriele Facciolo, Loïc Landrieu, Bertrand Neveu, Olivier Tournaire, Marina Vinyes, Maria Vakalopoulou. A special thank goes to Pierre Moulon, for his various help such as putting my code online, discussions, technical support in my work and happy hours on some Sunday afternoon. I also thank Brigitte Mondou, our excellent secretary, who gave me a lot of administrative support in visa, trip organization as well as any other documents.

Family: I want to give my thanks to our parents, who have supported me with all their hearts and taken care of our two babies. I reserve my greatest thanks for my wife, Huixin Yin, who has the determination of following me anywhere in the world regardless the language obstacles, and brought two angels to the world during my thesis. Things would not be possible without your support!

Family in France: I also want to express my gratitude to Emeric Bouin and his family, whom I have met since the begining of my study in France. You received me as a family member and gave me various support during my stay. I enjoyed the vacations we spent together and had the chances to integrate into French life. I wish all the best thing for you and hope to see you later in China!

Résumé

Grâce aux progrès récents en photogrammétrie, il est désormais possible de reconstruire automatiquement un modèle d'une scène 3D à partir de photographies ou d'une vidéo. La reconstruction est réalisée en plusieurs étapes. Tout d'abord, on détecte des traits saillants (features) dans chaque image, souvent des points mais plus généralement des régions. Puis on cherche à les mettre en correspondance entre images. On utilise ensuite les traits communs à deux images pour déterminer la pose (positions et orientations) relative des images. Puis les poses sont mises dans un même repère global et la position des traits saillants dans l'espace est reconstruite (structure from motion). Enfin, un modèle 3D dense de la scène peut être estimé.

La détection de traits saillants, leur appariement, ainsi que l'estimation de la position des caméras, jouent des rôles primordiaux dans la chaîne de reconstruction 3D. Des imprécisions ou des erreurs dans ces étapes ont un impact majeur sur la précision et la robustesse de la reconstruction de la scène entière. Dans cette thèse, nous nous intéressons à l'amélioration des méthodes pour établir la correspondance entre régions caractéristiques et pour les sélectionner lors de l'estimation des poses de caméras, afin de rendre les résultats de reconstruction plus robustes et plus précis.

Nous introduisons tout d'abord une contrainte photométrique pour une paire de correspondances (VLD) au sein d'une même image, qui est plus fiable que les contraintes purement géométriques. Puis, nous proposons une méthode semi-locale (K-VLD) pour la mise en correspondance, basée sur cette contrainte photométrique. Nous démontrons que notre méthode est très robuste pour des scènes rigides, mais aussi non-rigides ou répétitives, et qu'elle permet d'améliorer la robustesse et la précision de méthodes d'estimation de poses, notamment basées sur RANSAC.

Puis, pour améliorer l'estimation de la position des caméras, nous analysons la précision des reconstructions et des estimations de pose en fonction du nombre et de la qualité des correspondances. Nous en dérivons une formule expérimentale caractérisant la relation "qualité contre quantité". Sur cette base, nous proposons une méthode pour sélectionner un sous-ensemble des correspondances de meilleure qualité de façon à obtenir une très haute précision en estimation de poses.

Nous cherchons aussi à raffiner la précision de localisation des points en correspondance. Pour cela, nous développons une extension de la méthode de mise en correspondance aux moindres carrés (LSM) en introduisant un échantillonnage irrégulier et une exploration des échelles d'images. Nous montrons que le raffinement et la sélection de correspondances agissent indépendamment pour améliorer la reconstruction. Combinées, les deux méthodes produisent des résultats encore meilleurs.

Mots-clefs

vision par ordinateur ; stéréovision ; estimation robuste ; contrainte photométrique ; ligne virtuelle ; descripteur de ligne virtuelle ; méthode semi-locale de mise en correspondance ; sélection de correspondances ; raffinement de correspondances.

Abstract

With the recent progress in photogrammetry, it is now possible to automatically reconstruct a model of a 3D scene from pictures or videos. The model is reconstructed in several stages. First, salient features (often points, but more generally regions) are detected in each image. Second, features that are common in images pairs are matched. Third, matched features are used to estimate the relative pose (position and orientation) of images. The global poses are then computed as well as the 3D location of these features (structure from motion). Finally, a dense 3D model can be estimated.

The detection of salient features, their matching as well as the estimation of camera poses play a crucial role in the reconstruction process. Inaccuracies or errors in these stages have a major impact on the accuracy and robustness of reconstruction for the entire scene. In this thesis, we propose better methods for feature matching and feature selection, which improve the robustness and accuracy of existing methods for camera position estimation.

We first introduce a photometric pairwise constraint for feature matches (VLD), which is more reliable than geometric constraints. Then we propose a semi-local matching approach (K-VLD) using this photometric match constraint. We show that our method is very robust, not only for rigid scenes but also for non-rigid and repetitive scenes, which can improve the robustness and accuracy of pose estimation methods, such as based on RANSAC.

To improve the accuracy in camera position estimation, we study the accuracy of reconstruction and pose estimation in function of the number and quality of matches. We experimentally derive a "quantity vs. quality" relation. Using this relation, we propose a method to select a subset of good matches to produce highly accurate pose estimations.

We also aim at refining match position. For this, we propose an improvement of least square matching (LSM) using an irregular sampling grid and image scale exploration. We show that match refinement and match selection independently improve the reconstruction results, and when combined together, the results are further improved.

Keywords

computer vision; structure from motion; stereovision; robust estimation; photometric constraint; virtual line descriptor; semi-local matching; match selection; match refinement.

Sommaire

1 Preamble **15**
1.1 Methods for 3D reconstruction. 16
1.1.1 Laser scanner 16
1.1.2 Photometric stereo 16
1.1.3 Structured light scanner 17
1.1.4 Structure from Motion 17
1.2 Applications of 3D reconstruction 18

2 Introduction **21**
2.1 Thesis contribution 22
2.1.1 Publications 22
2.1.2 Software contributions 22
2.1.3 Teaching and supervision 23
2.2 Manuscript organization 24

3 Overview of Structure from Motion **25**
3.1 Notations 26
3.2 Introduction 27
3.3 Projective geometry 27
3.4 Pinhole camera model 28
3.5 Two-view camera geometry 30
3.6 Feature detection and description 33
3.6.1 Feature detection 34
3.6.2 Feature description 35
3.7 Feature matching 37
3.7.1 Local descriptor matching 37
3.7.2 Global matching strategy 38
3.7.3 Model fitting methods—the RANSAC family 39
3.7.4 Graph matching methods 42
3.8 Two-view camera calibration 43
3.8.1 Model refinement 43
3.8.2 Essential matrix decomposition 44
3.8.3 Bundle adjustment 45
3.9 N-view camera calibration 46
3.9.1 Incremental methods 46
3.9.2 Global methods 46

4 Feature correspondence **47**
4.1 Introduction . . . 49
4.1.1 Feature matching by RANSAC . . . 49
4.1.2 Graph matching methods . . . 49
4.1.3 Region growing methods . . . 50
4.2 Our contributions in matching . . . 50
4.2.1 Robust 2nd-order photometric criterion . . . 50
4.2.2 Light semi-local matching strategy . . . 51
4.3 Virtual line descriptor (VLD) . . . 51
4.3.1 Geometric consistency . . . 52
4.3.2 Line covering . . . 52
4.3.3 Inter-point gradient histogram . . . 53
4.3.4 Inter-point orientation . . . 53
4.3.5 Distance between two VLDs . . . 55
4.3.6 VLD-consistency . . . 55
4.3.7 High contrast suppression . . . 55
4.4 K-VLD: a K-connected VLD-based matching method . . . 56
4.4.1 Neighborhoods . . . 57
4.4.2 Problem statement . . . 58
4.4.3 Algorithm . . . 59
4.4.4 Optimizations and heuristics . . . 59
4.5 Evaluation . . . 60
4.5.1 Changing imaging conditions . . . 60
4.5.2 Strong occlusions . . . 60
4.5.3 Ambiguity and RANSAC prefiltering . . . 64
4.5.4 Comparison of ASIFT and K-VLD . . . 64
4.6 Parameters . . . 66
4.7 K-VLD's contribution for N-view SfM . . . 68
4.8 Running time . . . 69
4.9 Limitations of VLD and K-VLD methods . . . 70
4.9.1 Limitation w.r.t. detection inaccuracies . . . 70
4.9.2 Limitation w.r.t. repetitive patterns . . . 70
4.10 Conclusion . . . 71

5 Various applications of the K-VLD method **75**
5.1 Introduction . . . 75
5.2 Image Color Blending . . . 76
5.2.1 Introduction . . . 76
5.2.2 Related work . . . 76
5.2.3 K-VLD dense common region detection . . . 76
5.2.4 Application to the color coherence . . . 77
5.2.5 Conclusion . . . 78
5.3 Line segment matching . . . 80
5.3.1 Introduction . . . 80
5.3.2 Related work . . . 80
5.3.3 Our contribution . . . 81
5.3.4 Line segment detection and description . . . 81
5.3.5 Initial candidate matches . . . 81
5.3.6 Pairwise geometric constraint . . . 82
5.3.7 Semi-local photometric constraint . . . 83
5.3.8 K-VLD method for line segments . . . 84

5.3.9 Experiments . . . 84
5.3.10 Computation time . . . 89
5.3.11 Conclusion . . . 89
5.4 Deformable object matching . . . 91
5.4.1 Introduction . . . 91
5.4.2 Experiments . . . 91
5.5 Urban localization from Google street views . . . 97
5.5.1 Introduction . . . 97
5.5.2 K-VLD contribution in air-ground image matching . . . 97
5.5.3 Experiment . . . 99

6 Match refinement for highly accurate two-view SfM 101
6.1 Introduction . . . 102
6.2 Least Squares Matching (LSM) . . . 102
6.2.1 Local geometric transformation . . . 103
6.2.2 Local photometric transformation . . . 103
6.2.3 Local transformation extimation . . . 104
6.3 Sampling density and coverage zone . . . 104
6.4 Least Square Focused Matching . . . 105
6.4.1 Image intensity variation . . . 106
6.4.2 Focused grid . . . 106
6.4.3 Scale exploration . . . 108
6.5 Parameters . . . 109
6.6 Impact of the kind of feature detector . . . 109
6.7 Experiments . . . 110
6.7.1 Covering factor evaluation . . . 110
6.7.2 LSM vs. LSFM . . . 110
6.8 Extension: Surface normal estimation . . . 111
6.8.1 Introduction . . . 111
6.8.2 Homography Recovery . . . 112
6.8.3 Normal illustration . . . 112
6.9 Conclusion . . . 112

7 Match selection for highly accurate two-view SfM 115
7.1 Introduction . . . 117
7.2 Statistical behavior of SfM errors . . . 117
7.2.1 Theoretical results . . . 117
7.2.2 Empirical Results . . . 118
7.2.3 Realistic, semi-synthetic dataset . . . 119
7.2.4 Analysis . . . 119
7.3 Match Selection to Improve Accuracy . . . 123
7.3.1 Cleaning up input matches . . . 123
7.3.2 Comparing subsets of matches . . . 123
7.3.3 Exploring subsets of matches . . . 124
7.4 Ranking matches . . . 125
7.4.1 SIFT ranking function . . . 125
7.5 Algorithm . . . 126
7.5.1 Match selection alone (without match refinement) . . . 126
7.5.2 Match selection with match refinement . . . 127
7.5.3 Comparison to related methods . . . 128
7.6 Experiments . . . 128

7.6.1 Match selection and refinement 129
7.7 Visual illustration of 3D reconstruction accuracy 130
7.8 Number of matches kept by match selection 134
7.9 Various possible bias in alternative algorithms 134
7.9.1 Cleaning up matches with RANSAC before selection is biased . . 134
7.9.2 Distance to the epipolar line is biased for ranking matches 135
7.10 Conclusion 136
7.11 Annex 137
7.11.1 Regression correlation coefficient 137
7.11.2 Maximum crushing expression derivation 137

8 Conclusion and perspectives **139**
8.1 Conclusion 139
8.2 Perspectives 140

Chapter 1

Preamble

The first digital camera was built in 1975 by Steven Sasson. Since then, this technology has been expanding, spreading everywhere from satellites to watches. The birth of digital cameras transformed photos to a flow of digital numbers that can be manipulated via computers. Today's cameras are able to take photos or videos with much higher quality, raising up many applications and researches in computer vision. As photos capture the real world, a natural question is whether we can reconstruct the 3D world with captured photos. Several technologies have been developed to achieve the 3D reconstruction. The most popular one is *stereo-vision*. With a collection of photos taken from different view points, the idea of stereo-vision is to recover the spatial information of cameras as well as observed objects. Successful approaches such as *Structure from Motion* give the possibility to reconstruct a 3D world using a simple camera. These technologies have been applied in various domains: medical imaging, robotic vision, movie post-production, video game industry, planet exploration, etc. Many applications have specific requirements. Some put the stress on reconstruction accuracy, some need methods that can be applied under difficult conditions, others require real-time processing speed, etc, which leads to many active research area in stereo-vision.

This thesis aims at identifying and improving weak steps in 3D reconstruction pipelines, in terms of accuracy and robustness. Our work mainly focuses on the critical early stages of Structure from Motion for 3D reconstruction.

1.1 Methods for 3D reconstruction.

There exists many ways to reconstruct the 3D world, such as using a laser scanner, photometric stereo, structural pattern and structure from motion. We briefly go through these methods.

1.1.1 Laser scanner

When reconstructing a 3D scene, a laser scanner emits laser rays and receives their reflection in order to measure object positions. The most widely used scanners follow the LIDAR technology. It has many applications and its wavelengths is variable from about 10 micrometers to the UV to suit the material of the target. These scanners produce very accurate point clouds of the scanned scene. However, the price of these scanners is high and requires specific skills to be properly manipulated; moreover, the scanning times can be up to several minutes, depending on the required density of the point cloud. Thus, this method is mostly used for specific professional tasks.

Figure 1.1: Images from http://fr.wikipedia.org/wiki/Lidar and http://www.uav-lidar.com.

1.1.2 Photometric stereo

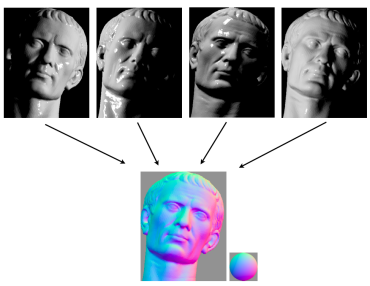


Figure 1.2: image from Wu et al. [85]

In an interior scene with a controlled light source and a camera at a fixed position, it is possible to estimate surface normals of an object. By taking several photos with different lighting conditions (with changes of light position), the surface normals at various points on the object are estimated by analyzing the reflected lights. The global surface structure is reconstructed by integrating the local surface normals. This method has a high requirement on the control of the light sources. Besides, any uncalibrated extra light source may degrade the result. This technique thus has a limited field of application.

1.1.3 Structured light scanner

There exists another kind of controlled light system for 3D reconstruction called "structured light scanner". It starts by projecting a light pattern containing specific structure such as lines, points or any other patterns to the scene. Then a camera observes the deformation of the projected pattern to reconstruct the depth map of the scene. The Kinect, developed and commercialized by Microsoft, uses this structured light scanner system, and is capable of reconstructing the depth map in real-time, using an infrared light pattern. As the camera captures specific patterns, it has a certain robustness against external light. However, the range of mapping depends on the reception of the pattern, and a too strong external light still erases the patterns. By consequence, structured light applies mostly to indoor scenes for the 3D reconstruction of small areas.

Figure 1.3: Kinect and its structured light patterns.

1.1.4 Structure from Motion

The 3D scene can also be reconstructed by digital photographs. By taking several images from different positions around an object, we can recover both the camera positions and the 3D scene (Figure 1.4). This method is called *structure from motion* (SfM) as the scene is reconstructed from moving camera positions. A more detailed description is available in Chapter 3. A variation to SfM is the *Simultaneous localization and mapping* (SLAM) method, which reconstructs 3D scene and maps the camera position at real time (e.g., on a video flow). Compared to the other methods, SfM has several advantages, making it the most explored approach in stereo vision for reconstruction:

- Low accessible price: Compared to LIDAR system, a camera for SfM is much cheaper.
- Mobility: Only a camera is needed for SfM photo acquisition. A camera can be of very small size and mounted on a UAV.
- Simple manipulation: No special skill is needed.
- Adapted to various conditions: SfM is suitable for both indoor and outdoor environments, and no special light condition is required.
- Large datasets available: With the fast growing availability of photo datasets on the internet such as Flickr, Facebook and Google images. A large quantity of photos are already present. People can reconstruct a popular scene even without taking photos by themselves.
- In practice, methods with projected light are limited to interior scene. In contrast, SfM doesn't need any projected light nor specific settings. Besides, it is less affected by the lighting condition and may recover the real color of the objects.

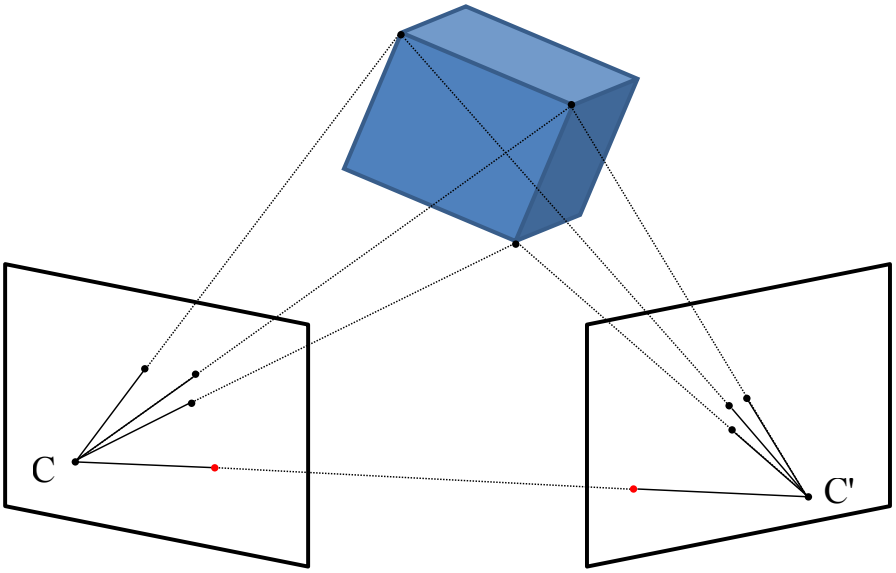


Figure 1.4: Structure from motion approaches reconstruct the 3D object from photos taken from different view points.

1.2 Applications of 3D reconstruction

The 3D reconstruction technologies have various applications in many areas. Some are very close to our everyday life.

Products such as Google Maps allow people to have an aerial view of the Earth, and its street-view function is able to provide a virtual visit by calibrating and adapting in real time the landscape images to the view point. Though the technology does not really reconstruct the 3D scene, but provides only discrete available view points, the result of virtual visits shows a very promising future. Projects that virtually reconstruct entire cities are also available from companies such as Acute3D[1].

The 3D reconstruction technology is also closely linked to robotics. The visual system in robots can help to provide location information based either on reconstructed scenes or on an external database [51]. The spread of unmanned vehicles with mounted cameras expands and requires better reconstruction technologies in terms of speed, accuracy and scale. The Google Car is able to drive and park alone, without human intervention. The micro aerial vehicles (MAV) as Black Hornet nano air vehicles carrying a camera are a fast growing area. Home cleaning robots with visual systems also show their advantages in trajectory planning.

Figure 1.5: Left: Google auto-drive car. Middle: Black Hornet nano air vehicle. Right: Hauzen VC-RE70V SLAM based vacuum robot.

[1] http://www.acute3d.com

In the movie and game industries, modeling realistic scene via stereo vision can reduce the costs. Actually, the 3D reconstruction technology still faces some limitations and is only used as an initial base on which manual scene refinements are required. But partial/semi-automatic reconstruction techniques are already commercialized (such as Kinect).

Chapter 2

Introduction

3D reconstruction by stereo-vision has been actively explored for years. The method "Structure from motion" is particularly efficient, and has been widely studied. The idea of SfM is as follows. First, images from different view-points of the object are captured. Second, specific features such as remarkable points or lines are extracted from images. Third, we establish correspondences between the features from different images pointing to the same space location. Fourth, in an either incremental or global manner, both approximate camera positions and actual feature location in 3D are estimated. Finally, in a multi-view case, an optimization taking into account all images is performed.

It is possible to reconstruct the scene from two images (2-view SfM) or more (N-view SfM). The 2-view SfM is a fundamental step in any case. Two steps play a very important role in SfM results:

- The feature correspondence has very important impact on the reconstruction quality, as it participates both in the camera position estimation and scene reconstruction.

- In 2-view and N-view SfM, the position estimation of cameras also have an important impact on the reconstruction result.

Many techniques have been proposed to produce high quality correspondences and camera positions. The standard pipeline is as follows: the program produces descriptors for detected features, and matches the features according to descriptor similarity. The initial matches are then filtered by RANSAC-like methods to separate inliers and outliers. Finally, the camera position is estimated by an optimization algorithm using as input all inliers.

This approach works correctly in general, but still faces problems:

- feature localization accuracy,

- robustness of feature matching,

- model optimization accuracy given inliers.

In this thesis, we try to identify problems occurring while establishing the point correspondences and estimating the camera positions. We propose solutions to overcome the present limitations, which have been tested under the 2-view SfM case and could be extended to the N-view SfM case in the future. Besides, our work in feature matching is not limited to SfM.

2.1 Thesis contribution

This thesis focuses on exploring new methods to increase the 2-view structure from motion performance in terms of robustness and accuracy.

The main contributions of the thesis are the following:

- A pairwise photometric constraint between feature correspondences: Most constraints between matches are based only on a geometric relation. We show that photometric information between matches can be used as a robust pairwise signature. We encode the photometric information between features using a *virtual line descriptor* (VLD), and use the similarity of VLDs to evaluate the photometric consistency between matches. The boost of using VLDs in pairwise match constraints is proved by our experiments.
- A semi-local matching method using VLD constraint: We introduce a semi-local approach in feature correspondence filtering. Using VLDs, our method outperforms the state-of-the-art algorithms in terms of accuracy, robustness and scalability. It is robust even for non-rigid scene.
- The "quantity vs. quality" trade-off in reconstruction accuracy: A study of reconstruction accuracy in the 2-view SfM case is carried out with a theoretical motivation and extended based on experimental observations, while varying the number of matches and the inaccuracy.
- A new match-selection approach for camera position estimation: The "quantity vs. quality" study shows the possibility of using fewer matches with higher accuracy to have a more accurate camera pose estimation. Different from traditional approaches that use all inliers to estimate the camera pose, our match-selection approach tests a series of match subsets of different sizes and their corresponding estimation candidates, and chooses a subset which better optimizes the "quantity vs. quality" trade-off.
- Extension of least square matching (LSM) for match localization refinement: We extend the LSM methods for match refinement with an image scale exploration and a focused grid.

2.1.1 Publications

- *Match Selection and Refinement for Highly Accurate Two-View Structure from Motion*. Z. Liu , P. Monasse and R. Marlet. European Conference on Computer Vision (ECCV), 2014, Oral.
- *Virtual Line Descriptor and Semi-Local Graph Matching Method for Reliable Feature Correspondence*. Z. Liu and R. Marlet. British Machine Vision Conference (BMVC), 2012, Poster.

2.1.2 Software contributions

K-VLD

We built an open source `C++` library implementing our semi-local matching method using photometric pairwise constraint. Matching tests for rigid scenes and deformable objects are provided (cf. Figure 2.1). Link: `https://github.com/Zhe-LIU-Imagine/KVLD`

Figure 2.1: Illustration of matching result by our online code of K-VLD method.

MRMS_online

We developed a `C++` library implementing the match refinement and match selection methods. The refinement method adjusts the feature locations for existing matches and selects the matches for an optimal "quality vs. quantity" trade-off. The two methods can be applied independently or combined to achieve more accurate 2-view structure from motion results (Figure 2.2). `https://github.com/Zhe-LIU-Imagine/MRMS_online`

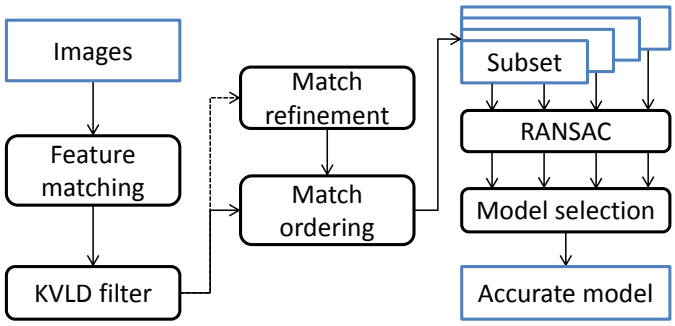


Figure 2.2: Pipeline of the combined method of match refinement and match selection.

Groti 2.0

We are the co-author of a visual assistant for studying feature detection, description, matching and calibration. It has various functions and provides information to help the analysis of every step of 2-view structure from motion.

2.1.3 Teaching and supervision

Teaching

I taught an introduction to C++ programming course for first year students at *École des Ponts ParisTech* (64 hours) for 3 years.

Supervised internship - Groti 2.0 bases

I supervised the internships in the IMAGINE group of two freshmen from *École des Ponts ParisTech*. The work was to setup the basic structure of the `Groti` project, using

the `Qt` platform.

2.2 Manuscript organization

The manuscript is organized in the following way. In Chapter 3, we start by describing the basics of structure from motion, including the feature matching process. Then we explain in Chapter 4 the semi-local matching approach with photometric pairwise constraint, followed in Chapter 5 by its various applications including a robust matching method for segments. In Chapter 6, we present our match refinement work. Finally, we introduce in Chapter 7 the match selection strategy for a better "quality vs. quantity" balance and a more accurate camera pose estimation .

Chapter 3

Overview of Structure from Motion

Before we go into the details of our work, this chapter gives a synoptic view of each step of the Structure from Motion approach (SfM).

Contents

3.1 Notations **26**
3.2 Introduction **27**
3.3 Projective geometry **27**
3.4 Pinhole camera model **28**
3.5 Two-view camera geometry **30**
3.6 Feature detection and description **33**
3.6.1 Feature detection 34
3.6.2 Feature description 35
3.7 Feature matching **37**
3.7.1 Local descriptor matching 37
3.7.2 Global matching strategy 38
3.7.3 Model fitting methods—the RANSAC family 39
3.7.4 Graph matching methods 42
3.8 Two-view camera calibration **43**
3.8.1 Model refinement 43
3.8.2 Essential matrix decomposition 44
3.8.3 Bundle adjustment 45
3.9 N-view camera calibration **46**
3.9.1 Incremental methods 46
3.9.2 Global methods 46

3.1 Notations

$\mathbb{R}^n$	Euclidean space
$\mathbb{P}^n$	Projective space
$\mathbf{X}$	3D point observed in world coordinate frame, $\mathbf{X} = \begin{pmatrix} X & Y & Z \end{pmatrix}^T$
$\mathbf{X}_c$	3D point observed in camera coordinate frame, $\mathbf{X}_c = \begin{pmatrix} X_c & Y_c & Z_c \end{pmatrix}^T$
$\tilde{\mathbf{X}}$	3D point in homogeneous form $\tilde{\mathbf{X}} = \begin{pmatrix} X & Y & Z & 1 \end{pmatrix}^T$
$\tilde{\mathbf{x}}_c$	Projection of $\mathbf{X}_c$ to the focal plan of camera $\mathbf{C}$ in homogeneous form, $\tilde{\mathbf{x}}_c = \begin{pmatrix} \frac{X_c f}{Z_c} & \frac{Y_c f}{Z_c} & f \end{pmatrix}^T \sim \mathbf{X}_c$
$\tilde{\mathbf{x}}$	2D point in homogeneous form in pixel dimensions, $\tilde{\mathbf{x}} = \begin{pmatrix} u & v & 1 \end{pmatrix}^T$
$\mathbf{x}$	2D point in pixel dimensions in the image, $\mathbf{x} = \begin{pmatrix} u & v \end{pmatrix}^T$
R	Rotation matrix
$\mathbf{t}$	Translation vector
$\mathbf{C}$	Camera center in world coordinate frame
$\mathbf{K}$	Calibration matrix, intrinsic parameters of a camera
$\mathbf{P}$	Camera central projection matrix
π_f	focal plane of a camera

3.2 Introduction

In order to reconstruct a 3D scene, an SfM approach usually goes through the following steps. Interest points (features) are detected, and a specific signature (descriptor) is assigned to every feature. A matching strategy is then applied to establish feature correspondences. Given the feature matches between two images, we recover the epipolar geometry by estimating a fundamental matrix F relating them. If the internal calibration parameters are known (the calibration matrix K), this also provides an estimation of the camera motion (rotation R, translation t) and the 3D positions of matched points (cf. Figure 3.1).

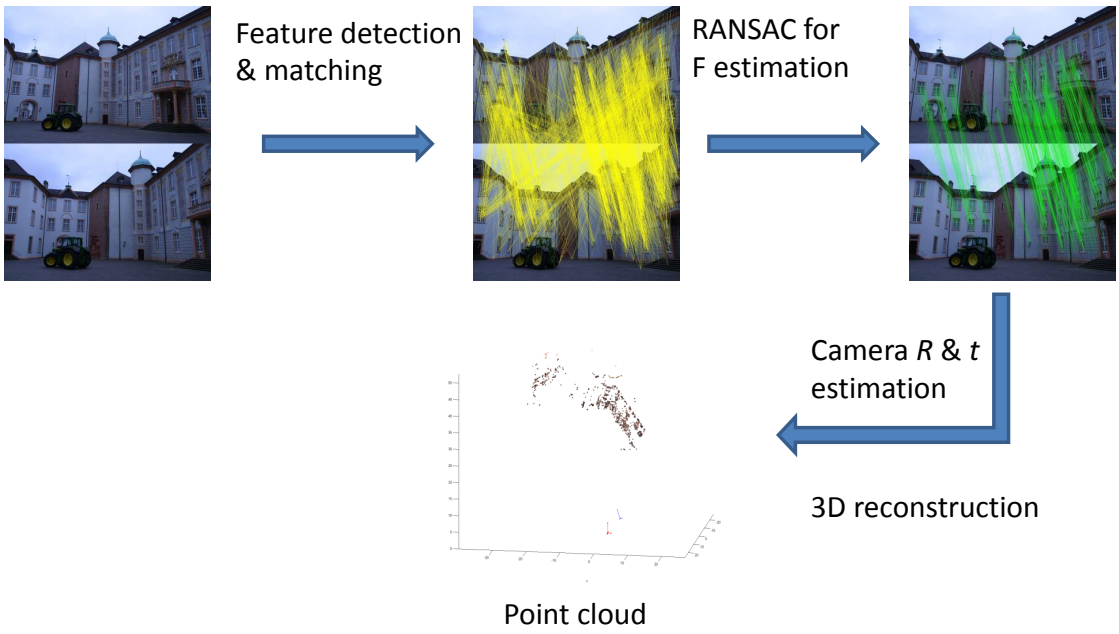


Figure 3.1: A classical two-view SfM pipeline

Organization

In this chapter, we recall some elements of projective geometry and the pinhole camera model, with an emphasis on two-view camera geometry. We then go through the feature detection, description and matching methods. Finally, we explain how the camera poses are estimated (external calibration phase).

3.3 Projective geometry

In all the following sections, we work in the Euclidean space and/or the projective space. More detailed notions about projective geometry can be found in [34]. A projective space has the following properties:

1. A point $\mathbf{X} = (X_1, X_2, \dots X_n) \in \mathbb{R}^n$ of dimension n in Euclidean space is represented by a vector of dimension $n+1$ in projective space as $(X_1, X_2, \dots X_n, 1) \in \mathbb{P}^n$, which is also called the homogeneous form of the point $\mathbf{X}$. It is also written $\tilde{\mathbf{X}} = (\mathbf{X}, 1) \in \mathbb{P}^n$ for short.

2. For any $c \in \mathbb{R}\backslash\{0\}$, and point $\tilde{\mathbf{X}} = (X_1, X_2, \dots X_n, w) \in \mathbb{P}^n$, the point $c\tilde{\mathbf{X}} = (cX_1, cX_2, \dots cX_n, cw)$ is considered as equivalent to $\tilde{\mathbf{X}}$, noted as $c\tilde{\mathbf{X}} \sim \tilde{\mathbf{X}}$.

3. Points at infinity are noted as $(X_1, X_2, \dots X_n, 0)$. Note that $(0, 0, \dots, 0) \notin \mathbb{P}^n$.

We go through two concrete examples to illustrate how projective geometry simplifies algebraic formulations.

Projection of 3D points to a plane

In the coordinate frame of the camera, where the center $\mathbf{C}$ of the camera is the coordinate origin, we suppose a 3D point $\mathbf{X}_c = (X_c, Y_c, Z_c)$ is projected to the image plane $z = f$ to point $\tilde{\mathbf{x}}_c = (\frac{X_c f}{Z_c}, \frac{Y_c f}{Z_c}, f)$ in 2D space through a ray passing by the center of the camera and $\mathbf{X}_c$. All points on this ray $\{\mathbf{X}' = c\mathbf{X}_c | c \in \mathbb{R}\backslash\{0\}\}$ project to the same $\tilde{\mathbf{x}}_c$ on the image plane. This becomes obvious with the 2D projective space $\mathbb{P}^2$ as all points in the ray $\{(cX_c, cY_c, cZ_c), c \in \mathbb{R}\backslash\{0\}\} \in \mathbb{P}^2$ are similar to $\tilde{\mathbf{x}}_c = (\frac{X_c f}{Z_c}, \frac{Y_c f}{Z_c}, f)$. That is to say, considering $\mathbb{P}^2 = \mathbb{R}^3\backslash\{0\}$, we have $\tilde{\mathbf{x}}_c \sim \mathbf{X}_c$.

Reference change in 3D space

In a Euclidean space, if the camera is not centered at the origin, a point observed at $\mathbf{X}_c = (X_c, Y_c, Z_c)$ in the camera coordinate, is expressed as $\mathbf{X} = (X, Y, Z)$ in the absolute coordinate frame. Then we have $\mathbf{X}_c = R(\mathbf{X} - \mathbf{C})$, where R is the rotation of the world coordinate from the camera coordinate. We define the camera translation as:

$$\mathbf{t} = -R\mathbf{C}. \tag{3.1}$$

The relation between $\mathbf{X}_c$ and $\mathbf{X}$ can be expressed in terms of a rotation and a translation, i.e., with R the rotation of world coordinate axes observed in the camera coordinates, and $\mathbf{t}$ the translation of the origin of the world coordinate observed in the camera coordinate, we write $\mathbf{X}_c = R\mathbf{X} + \mathbf{t}$. Under homogeneous forms, this expression becomes:

$$\begin{bmatrix} \mathbf{X}_c \\ 1 \end{bmatrix} = \begin{bmatrix} R & \mathbf{t} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix} \tag{3.2}$$

The rotation matrix R and the translation vector $\mathbf{t}$ are also called the extrinsic parameters of a camera.

3.4 Pinhole camera model

A digital camera is equipped with a grid of optical sensors that can capture light intensity. When we take a photo, light enters the camera and goes through a series of lenses before it finally hits an optical sensor during a short exposure time. If the captured image is very near to a projection of the 3D scene to a 2D plan through a pinhole, we obtain a simple projection model that is called pinhole camera model. This model is also equivalent to the projection of 3D scenes through a very fine convex lens. The principal property of the pinhole camera model involved here is that all light rays passing through the center of the camera do not change direction, thus all points are projected on the intersections of the image plane and the rays passing through the camera center from the 3D points. By convention, we define the z direction as the direction where the camera is pointed, see Figure 3.2.

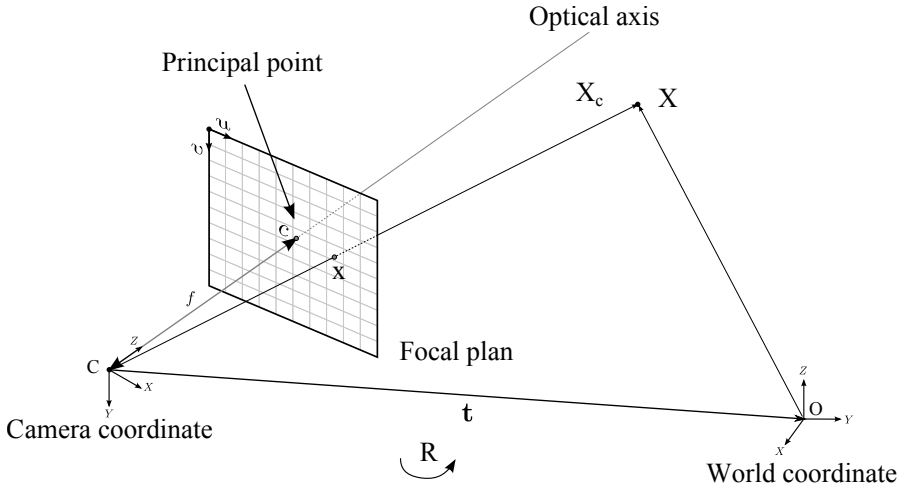


Figure 3.2: Pinhole camera illustration. A 3D point $\mathbf{X}$ observed in world coordinate has coordinate $\mathbf{X}_c$ in camera coordinate; it is projected to $\tilde{\mathbf{x}}$ on the focal plane by a perspective projection of the camera center. The rotation matrix and the translation vector between the camera orientation and position $\mathbf{C}$ and the world coordinate of origin $\mathbf{O}$ are called the camera extrinsic parameters. The focal plane is usually situated behind the camera center $\mathbf{C}$ at $z = -f$; however, it is equivalent and simpler to consider a focal plane in front of the center.

Image coordinate and Camera coordinate

We want to establish a relation between the projection of a 3D point on the focal plane and its pixel position captured in the image. Supposing a 3D point observed in camera coordinate $\mathbf{X}_c = (X_c, Y_c, Z_c)$ is projected on a 2D point $\mathbf{x}_c = (\frac{X_c f}{Z_c}, \frac{Y_c f}{Z_c})$ of the focal plane, with an associated homogeneous form $\tilde{\mathbf{x}}_c = (\frac{X_c f}{Z_c}, \frac{Y_c f}{Z_c}, f)$, and this point is captured in an image at position (u, v). We associate an homogeneous form for this position $\tilde{\mathbf{x}} = (u, v, 1)$ in image. The point $(0, 0, f)$, origin of the plane $z = f$ is not necessarily the origin of the image, we note its position in the image as (x_0, y_0) in terms of pixel dimensions and call it the principal point, see Figure 3.3. For a unit distance in the world coordinate frame, we note the number of pixels in x and y directions as m_x and m_y. The relation between $\tilde{\mathbf{x}}_c$ and $\tilde{\mathbf{x}}$ is thus expressed by (3.3):

$$\tilde{\mathbf{x}} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \mathbf{K}\tilde{\mathbf{x}}_c, \text{ where } \mathbf{K} = \begin{bmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$
$$\text{with } \alpha_x = m_x f \text{ and } \alpha_y = m_y f$$

The matrix $\mathbf{K}$ is called the camera calibration matrix, m_x and m_y are usually close but not necessarily equal. s is the shearing parameter. The matrix $\mathbf{K}$ is also called the intrinsic parameters of a camera. Here the common defect known as image distortion is not taken into account.

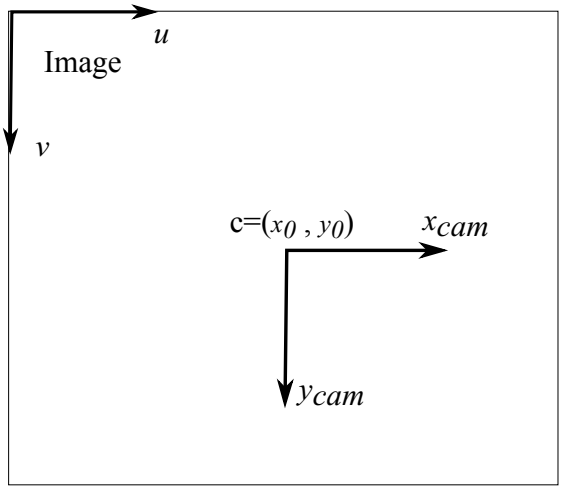


Figure 3.3: Image coordinate and camera coordinate, c is the principal point. By convention, the origin of the image is on the top left corner with v direction pointing down.

Intrinsic and extrinsic parameters of a pinhole camera

Finally, we can put the world coordinates, the camera coordinates and the image coordinates together in an equation and write:

$$\tilde{\mathbf{x}} \sim \mathbf{K}\tilde{\mathbf{x}}_c \sim \mathbf{K}\mathbf{X}_c = \mathbf{K}\begin{bmatrix} R & t \end{bmatrix}\begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix}. \tag{3.4}$$

The matrix $\mathbf{P} = \mathbf{K}\begin{bmatrix} R & t \end{bmatrix}$ is called the camera projection matrix, with $\mathbf{K}$ the *internal* parameters describing camera intrinsic properties, and $\begin{bmatrix} R & t \end{bmatrix}$ the *external* parameters describing camera orientation and position.

3.5 Two-view camera geometry

This section presents the relationship between observations from different cameras of a common 3D point $\mathbf{X}$. We discuss only the two-view case, as it is similar for the general N-view case. In this section we suppose there is a 3D point $\mathbf{X}$ in world coordinates, and there are two cameras located at position $\mathbf{C}$ and $\mathbf{C}'$ with rotation R and R', and intrinsic calibration matrix $\mathbf{K}$ and $\mathbf{K}'$. For convenience reason, we set $R = \mathbb{1}$ and $\mathbf{C} = 0$. According to the previous section, we have:

$$\tilde{\mathbf{x}} \sim \mathbf{K}\tilde{\mathbf{x}}_c \text{ and } \tilde{\mathbf{x}}_c \sim \begin{bmatrix} R & t \end{bmatrix}\begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix},$$
$$\tilde{\mathbf{x}}' \sim \mathbf{K}'\tilde{\mathbf{x}}'_c \text{ and } \tilde{\mathbf{x}}'_c \sim \begin{bmatrix} R' & t' \end{bmatrix}\begin{bmatrix} \mathbf{X} \\ 1 \end{bmatrix}.$$

Epipolar geometry

The epipolar geometry describes basically the relationship linking $\tilde{\mathbf{x}}_c$ and $\tilde{\mathbf{x}}'_c$ between two cameras. We know that $\mathbf{C}$, $\mathbf{C}'$ and $\mathbf{X}$ define a plane noted as $\pi_p = (\mathbf{C}\mathbf{X}\mathbf{C}')$. $\tilde{\mathbf{x}}_c$ and $\tilde{\mathbf{x}}'_c$ lay on π_p as $\overrightarrow{\mathbf{C}\mathbf{X}}$ and $\overrightarrow{\mathbf{C}'\mathbf{X}}$ pass through them, which means $\tilde{\mathbf{x}}_c$ (respectively $\tilde{\mathbf{x}}'_c$) must be on the intersection of image plane π_f (respectively π'_f) and π_p. By consequence, knowing the camera positions and $\tilde{\mathbf{x}}_c$ without knowing $\mathbf{X}$, $\tilde{\mathbf{x}}'_c$ must lay on the line l'_e

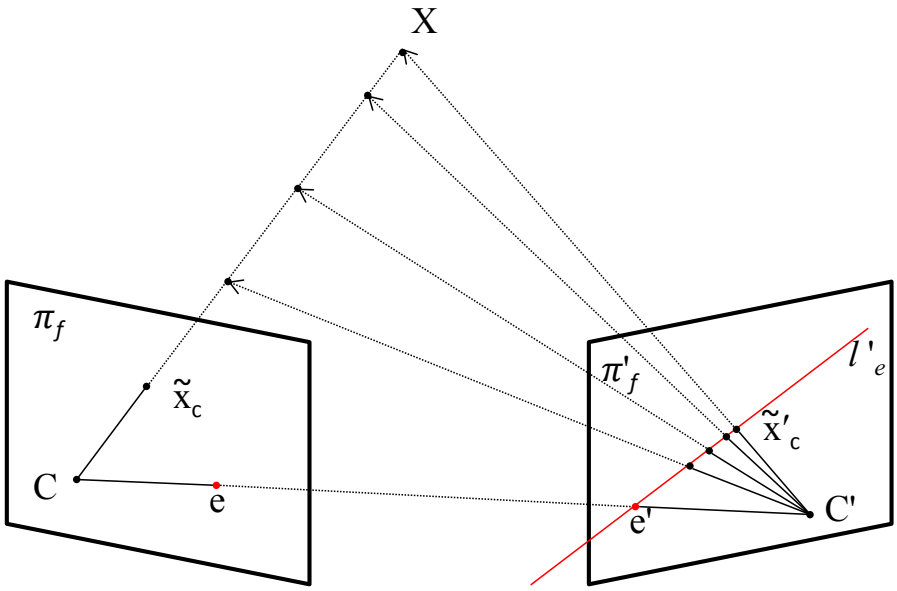


Figure 3.4: Illustration of epipolar geometry: $\mathbf{C}$ and $\mathbf{C}'$ are camera centers. For a 3D point $\mathbf{X}$, we observe point $\tilde{\mathbf{x}}_c$ on the focal plane of the left camera, and $\tilde{\mathbf{x}}'_c$ on the right camera. The epipolar geometry expresses the following constraint: knowing $\tilde{\mathbf{x}}_c$, point $\tilde{\mathbf{x}}'_c$ lays on the intersection line l'_e of plane $(\mathbf{CXC}')$ and the focal plane π'_f. l'_e is the epipolar line for $\tilde{\mathbf{x}}'_c$. Different positions on l'_e correspond to different 3D points on the ray $\overrightarrow{\mathbf{C}\tilde{\mathbf{x}}_c}$. $\mathbf{e}'$ is the perspective projection of $\mathbf{C}$ on π'_f , all epipolar lines l'_e pass through $\mathbf{e}'$, and vise-versa for $\mathbf{e}$.

defined by $\mathbf{C}$, $\tilde{\mathbf{x}}_c$ and $\mathbf{C}'$ as described on Figure 3.4 called the *epipolar line*. However, there is no way to determine the exact location of $\mathbf{X}$ or $\tilde{\mathbf{x}}'_c$, as different positions of $\tilde{\mathbf{x}}'_c$ on the epipolar line correspond to 3D points at different distances to the camera center $\mathbf{C}$ on the ray $\overrightarrow{\mathbf{C}\tilde{\mathbf{x}}_c}$.

Essential matrix

More formally, we can write this constraint as:

$$(\mathbf{X}-\mathbf{C}')^T[(\mathbf{C}'-\mathbf{C})]_\times(\mathbf{X}-\mathbf{C})=0 \tag{3.5}$$

where $[v]_\times$ denotes the matrix of the cross-product with vector v on the left.

We have supposed that $R=\mathbb{1}$ and $\mathbf{C}=0$, thus $\mathbf{X}_c=\mathbf{X}$. Besides, we apply (3.1), then $\mathbf{C}'-\mathbf{C}=-R'^{-1}\mathbf{t}$. Using the right part of (3.4) and $R'^{-1}=R'^T$, we get $\mathbf{X}'_c=R'(\mathbf{X}-\mathbf{C}')$, i.e., $\mathbf{X}'^T_c R'=(\mathbf{X}-\mathbf{C}')^T$. Finally, we can simplify (3.5) to:

$$\mathbf{X}'^T_c R'[R'^T\mathbf{t}']_\times\mathbf{X}_c=0 \;\Leftrightarrow\; \tilde{\mathbf{x}}'^T_c R'[R'^T\mathbf{t}']_\times\tilde{\mathbf{x}}_c=0$$

or:

$$\begin{aligned}&\tilde{\mathbf{x}}'^T_c\mathbf{E}\tilde{\mathbf{x}}_c=0\\&\text{with }\mathbf{E}=R'[R'^T\mathbf{t}']_\times\end{aligned} \tag{3.6}$$

The term $R'[R'^T\mathbf{t}']_\times$ is defined as the *essential matrix* $\mathbf{E}$ of the second camera to the first one according to [34]. It is not unique for a given pair of cameras, since $c\mathbf{E}$ with $c\in\mathbb{R}$ also encodes the same constraints as (3.6). The essential matrix has 5 degrees of freedom, namely 3 angles for the rotation and 2 for the translation direction.

Fundamental matrix

Equation (3.6) can be further transformed to the following:

$$\tilde{\mathbf{x}}'^T \mathbf{K}'^{-T} R' [R'^T \mathbf{t}']_\times \mathbf{K}^{-1} \tilde{\mathbf{x}} = 0,$$

or

$$\begin{aligned} \tilde{\mathbf{x}}'^T \mathbf{F} \tilde{\mathbf{x}} &= 0 \\ \mathbf{F} = \mathbf{K}'^{-T} \mathbf{E} \mathbf{K}^{-1} &= \mathbf{K}'^{-T} R' [R'^T \mathbf{t}']_\times \mathbf{K}^{-1}. \end{aligned} \tag{3.7}$$

Here $\mathbf{F}$ is called the *fundamental matrix* of the second camera to the first one. The main reason of using the fundamental matrix rather than the essential matrix is that it establishes a relationship between different observations in image positions, which is more straightforward. For instance, we observe $\tilde{\mathbf{x}}$ in the image, but to calculate $\tilde{\mathbf{x}}_c$, we also need to know $\mathbf{K}$ which is not always available. The difficulty of dealing with F comes from its 7 degrees of freedom, since $\det F = 0$ and the scale of F is insignificant.

We have formulated the epipolar constraint using either the fundamental or the essential matrix. As explained before, it is not possible to determine the exact location of $\tilde{\mathbf{x}}'$ or $\tilde{\mathbf{x}}'_c$ on the epipolar line without more information about $\mathbf{X}$.

Homography transformation

One pertinent extra information could be that $\mathbf{X}$ lays on a plane $\pi_h = \{n, d\}$ of equation $n^T \mathbf{X} + d = 0$, which is a common case for planar scenes, see Figure 3.5. We suppose π_h doesn't pass through $\mathbf{C}$. Since $R = \mathbb{1}$ and $\mathbf{t} = 0$, we have $\tilde{\mathbf{x}}_c \sim \mathbf{X}_c = \mathbf{X}$, supposing $\lambda \tilde{\mathbf{x}}_c = \mathbf{X}$ we have $\lambda n^T \tilde{\mathbf{x}}_c + d = 0$, which leads to $\lambda = -d/(n^T \tilde{\mathbf{x}}_c)$. So we have $\mathbf{X} = -d\tilde{\mathbf{x}}_c/(n^T \tilde{\mathbf{x}}_c)$. Now, according to Equation (3.4) for the second camera, the extra constraint can be expressed as following:

$$\tilde{\mathbf{x}}'_c \sim \mathbf{X}'_c = R' \lambda \tilde{\mathbf{x}}_c + \mathbf{t}',$$

which leads to

$$\tilde{\mathbf{x}}'_c \sim R' \tilde{\mathbf{x}}_c + \mathbf{t}/\lambda = (R' - \mathbf{t} n^T / d) \tilde{\mathbf{x}}_c.$$

We rewrite the equation using $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{x}}'$, and finally get:

$$\tilde{\mathbf{x}}' \sim \mathbf{K}' (R' - \mathbf{t} n^T / d) \mathbf{K}^{-1} \tilde{\mathbf{x}}. \tag{3.8}$$

Here, $\mathbf{H} = \mathbf{K}' (R' - \mathbf{t} n^T / d) \mathbf{K}^{-1}$ is called a *homography matrix*, and the transformation of a planar scene between two viewpoints is defined as a *homography transformation*. The homography matrix has 8 degrees of freedom, for scaling is insignificant.

Conclusion

We have established relationships for observations of a same 3D point from two different view points in the general case and in the particular case of a homography transformation case. Knowing the essential matrix, or the fundamental/homograph matrix with intrinsic parameters, we can recover camera positions, which is not available in general case. To solve this inverse problem, we need 3D points captured from different images, or more exactly, several observations $(\mathbf{x}, \mathbf{x}')$ of the same 3D points. This is the *point correspondence* problem. The next section explains techniques to establish correspondences, especially for points.

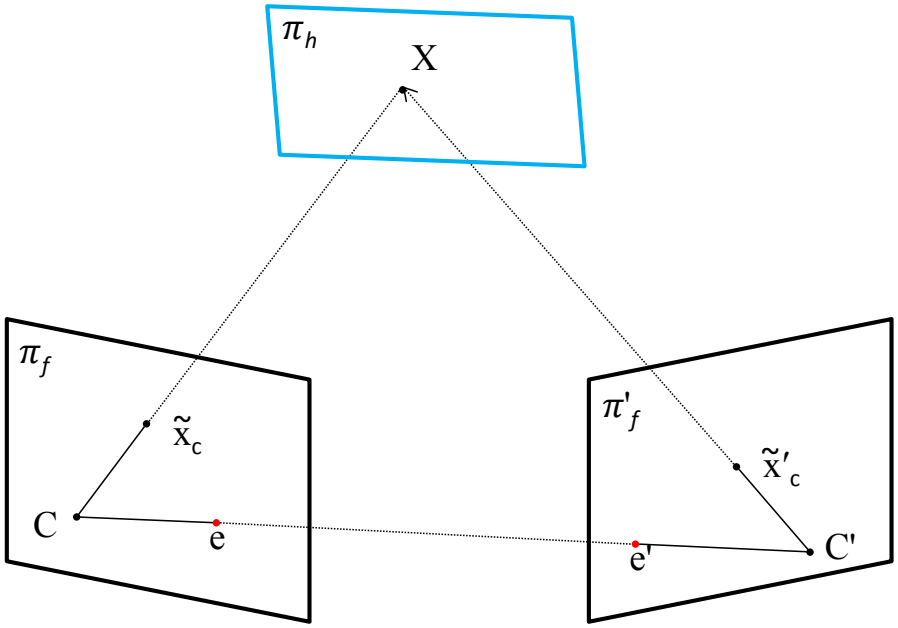


Figure 3.5: Illustration of homography transformation.

3.6 Feature detection and description

As explained before, we will need to retrieve observations of a same point in different images. This raises several issues. First, given an observation in one image, the question is how to efficiently and robustly find in the other images the corresponding observation. Second, how to localize as accurately as possible different observations of a same object? Even with the best techniques, a digital image is produced by a grid of optical sensors and thus only a discrete sampling of the real world's projection, which causes imprecision of location.

To overcome these difficulties, practitioners rely on feature matching techniques, which form a fundamental research area in computer vision. In images, features are specific structures that are more or less stable from different view points. They can be points, edges or any other object satisfying certain stability properties.

To solve the inverse problem of camera position recovery, people first detect features in images, then describe each feature with either geometric or photometric information over an appropriate zone around the detected feature, and finally try to establish the correct correspondences between features. This pipeline involves three important topics as feature detection, feature description, and feature matching. In this section and in the following one, we will give an overview of these topics.

- Feature detection: Extracting specific structures from images.
- Feature description: Assigning a more or less view-point invariant signature to each feature, so that observations of the same object in the other images have a similar signature, and different features have different signatures.
- Feature matching: Assigning correspondences to features from different images using various methods, based on feature descriptor comparison, and possibly, geometric and/or photometric consistency.

3.6.1 Feature detection

It is beyond hope to try to find correspondences for every point in an image directly, as some textureless positions in the images are hard to be correctly located in the other images. In stereo vision, people look for specific features in the images, and work only with these detected features. There are many criteria to evaluate the performance of features detection.

- **Location accuracy**: Features should be **distinctive** with respect to their neighborhood to be accurately located.
- **Robustness/repeatability**: Features should be **robust** under moderate view point changes. Precisely, since photos could be taken from any position (rotation and translation in space) and distance, features should still be detected under translation, orientation and scale changes. Some more advanced features detectors are designed to be invariant under an affine transformation.
- **Scale and orientation**: Features do not necessarily have a specific size nor orientation. However, the feature descriptor needs to choose an area to produce a signature, and knowing an orientation largely reduces the description space. In order to always describe the feature based on a similar area in different images, feature detection is usually coupled with a detection scale and orientation.
- **Density and distribution**: A good feature detector should detect a sufficient number of features. Ideally, detected features should also be evenly distributed in the whole image as opposed to being concentrated in a small area. Detectors should also avoid the overlap of similar features (with similar coordinate, orientation and scale) to avoid the ambiguity in feature description.

Some aspects restrain others, such as increasing accuracy and robustness may reduce feature numbers. There are several categories of features known in computer vision, such as corners, segments and regions. We will focus on features describing corners and regions as they are quite similar in presentation and application.

Figure 3.6: Feature detection: specific structures of images are detected, along with a scale (circle radius length) and an orientation (displayed radius).

Corners and regions

Corners were used relatively early as features. The first corner features that have been largely used were Harris corners [33] in 1988, which are invariant to orientation and do

not take scale into account. The work of Lindeberg [46] analyzed the rescaling effect on images and characterized the properties of scale-invariant features. Then, the Harris-Laplace detector [56] extended the Harris detector with image scale exploration. The image scale exploration has also been applied to Hessian-affine region detectors [55] and led to the famous SIFT detector, defined by D.G. Lowe [50] in 1999. SIFT uses an image pyramid to present image scales and the differences of Gaussian-convolved images to efficiently compute scale invariant blob-like features. It has been further refined by Brown and Lowe [12] to achieve sub-pixel precision. The work of Morel and Yu [59] in 2009 introduces various synthetic images with different view points to increase robustness against perspective transformation. The KAZE feature by Alcantarilla et al. [4] in 2012 introduced non-linear image scale space to increase the robustness.

Matas et al. [52] have developed the MSER feature detector, which detects contrasted regions in the images. Later work by Forssen and Lowe [28] also incorporated the scale exploration.

Most of the above features have the problem of a computational burden. Thus people have also looked at the speed in detection. SURF [74] uses integral images to efficiently (in memory and speed) interpret images of different scales. FAST [69] applies a learning approach to detect corner features.

Comparison of feature categories

Usually, corner detectors are more accurate in localization than region detectors; however, corners often lay on the border of objects and are less reliable in scale, making their surroundings unstable under view-point change. Thus it adds extra difficulties to create reliable signature for corners. Blob-like feature detectors discover stable regions under view-point change with a reliable scale, and for the reason explained before, it is easier to describe these features. In our experiments, MSER detects much fewer features than blob-like features.

3.6.2 Feature description

Once features are detected, we want to locate observations from different images of a same object and link them together. For this, detected features are assigned with a signature called a descriptor. A descriptor efficiently characterizes the information of an area around the feature, it is usually a vector of numbers. A good descriptor should produces similar signatures for different observations of a same point $\mathbf{X}$ and different-enough signatures for different points (cf. Figure 3.7). Besides, a descriptor should deal with several challenging situations for realistic images, and be robust to these situations:

- Invariant to light conditions: as photos are taken at different times, the light conditions may be different, including shading.
- Invariant to translation, rotation and scale: as photos are taken from different positions, features from different images of a same object are translated, rotated and re-scaled. (More complex transformations may occur.)

Besides, descriptors must deal with a dilemma between distinctiveness and repeatability. Signatures need to be distinctive enough to differentiate from each others and should be repeated from different images under different challenging situation. Distinctive descriptors tend to be less repeatable and vise-versa.

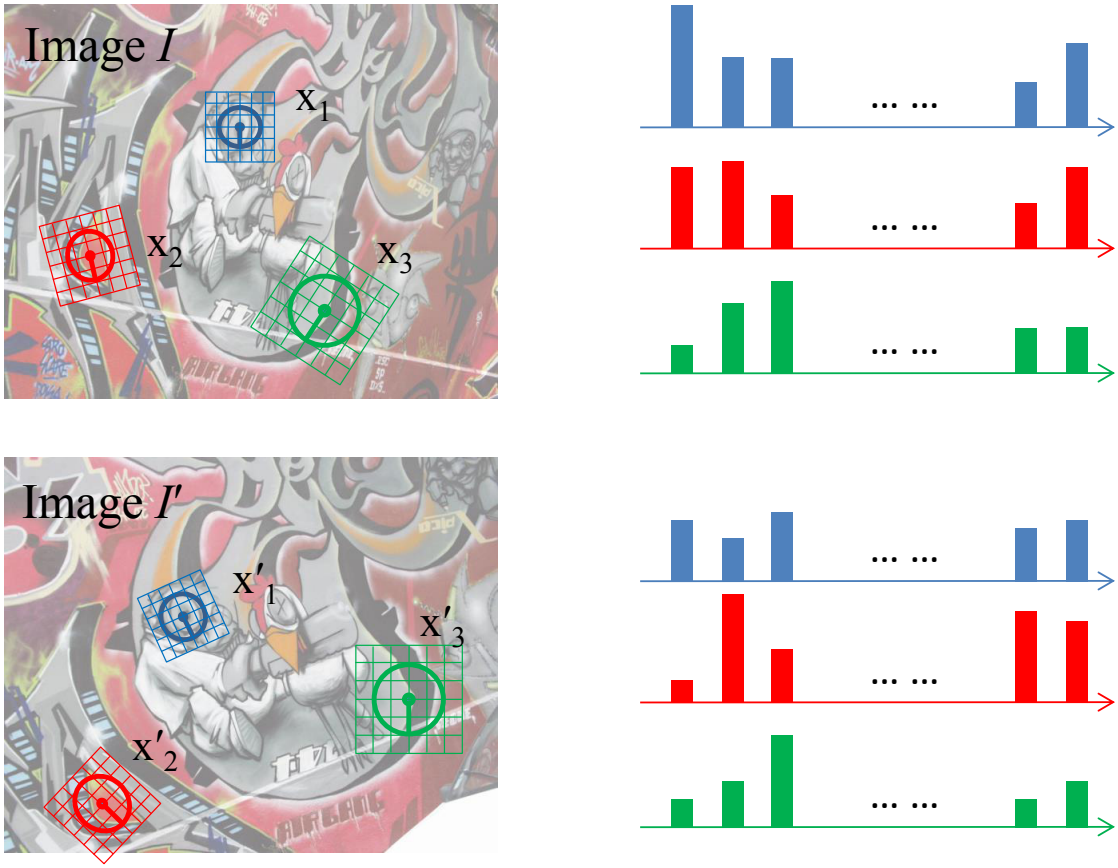


Figure 3.7: Feature description: a signature characterizing a feature neighborhood is computed for every detected feature.

The SIFT descriptor by Lowe [50] in 1999 has been widely used for its performance facing these criteria. The SIFT descriptor, a 128 float-size vector, consists of 16 histograms of gradients discretized into 8 bins. It encodes a form of spatial information around the feature. The gradients are produced at the detected image scale. The use of gradients is more robust than the direct use of image intensity against light changes. With the help of feature orientation, the 16 histograms characterize the area around the feature and the use of histograms allows small shift of gradients due to perspective transformation. This descriptor has been applied to various detectors such as MSER, Harris-Laplace and Hessian-affine features... The work by Mikolajczyk and Schmid [53] shows that the SIFT descriptor has good performance under various situations.

The SIFT descriptor is a vector of length 128 for every feature, while there could be thousands of features in one image. This limits the applications for a large quantity of images. To reduce the memory consumed by SIFT, various approaches have been proposed. In matching learning, discrete SIFT features are used in the *bag of words* model; some also reduce descriptor size. Alternatively, Calonder et al. [14] calculate a binary descriptor based on intensity comparison between samples, and use a learning method to train a more performing descriptor with much smaller memory requirements.

A feature descriptor characterizes the local information around features by a signature, which is used to establish feature correspondences. Due to limited local information and challenging changing situations, pairing features with most similar descriptors is not sufficient.

3.7 Feature matching

Features correspondences (cf. Figure 3.8), also called matches, are difficult to establish but are very important for SfM. Here we focus on the two-view case. We suppose we have images I and I' with extracted features $\{\mathbf{x}_i, i \in \{1,\dots,n\}\}$ and $\{\mathbf{x}'_{i'}, i' \in \{1,\dots,n'\}\}$. We note descriptors for features as $\{desc(\mathbf{x}_i)\}$ and $\{desc(\mathbf{x}'_{i'})\}$. It is much more efficient to match only features instead of matching every pixel, but there still exists $n \times n'$ possibilities. In order to produce reliable correspondences, several strategies at local and global scales can be applied.

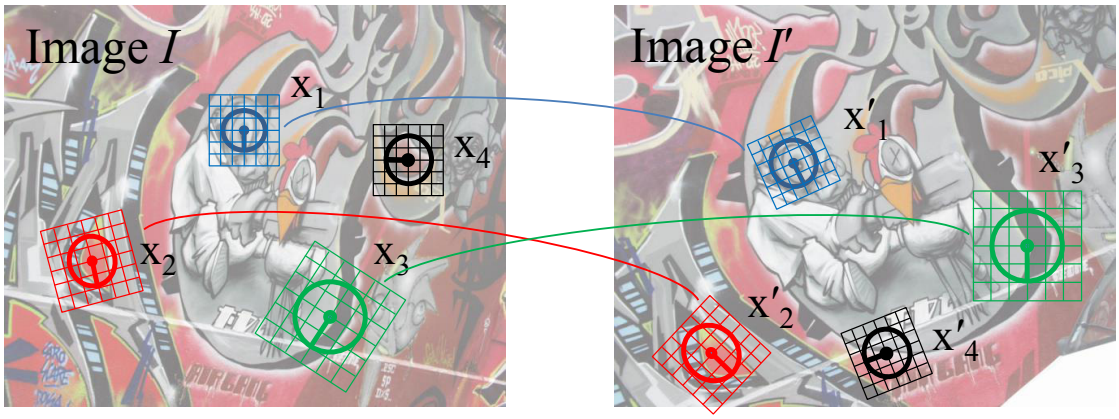


Figure 3.8: Feature matching.

3.7.1 Local descriptor matching

Locally, we can pair features according to the similarity of signatures. In order to produce correct matches, various strategies can be applied for different purposes.

First nearest neighbor (FNN)

The most basic approach is to match with the first nearest neighbor (FNN) in the signature space. In other words, for each feature $\mathbf{x}_i$, we look for $\mathbf{x}'_{i'}$ satisfying

$$\mathbf{x}'_{i'} = \underset{\mathbf{x}'_{j'}}{\arg\min}\, d\left(desc(\mathbf{x}_i), desc(\mathbf{x}'_{j'})\right)$$

where $desc(\mathbf{x})$ is the signature of x and $d(s,s')$ is the distance between signatures s and s'.

Note that this approach is not symmetric. Thus if $\mathbf{x}'_{i'}$ is the FNN for $\mathbf{x}_i$, conversely $\mathbf{x}_i$ may not be the FNN of $\mathbf{x}'_{i'}$.

Symmetric matching

The next approach adds the symmetric property to FNN. A pair of features $(\mathbf{x}_i, \mathbf{x}'_{i'})$ construct a match if and only if they are FNN of each other.

$$\mathbf{x}'_{i'} = \underset{\mathbf{x}'_{j'}}{\arg\min}\, d\left(desc(\mathbf{x}_i), desc(\mathbf{x}'_{j'})\right)$$

$$\mathbf{x}_i = \underset{\mathbf{x}_j}{\arg\min}\, d\left(desc(\mathbf{x}_j), desc(\mathbf{x}'_{i'})\right)$$

Lowe ratio

In natural scenes, and even more so in urban scenes, it is very common to have repetitive features, which creates ambiguity as descriptors of different features can be very similar. To partly get around ambiguous choices, Lowe [50] introduced a threshold ratio. Given $\mathbf{x}_i$, let $\mathbf{x}'_{i'}$ be its FNN and $\mathbf{x}'_{j'}$ be its second nearest neighbor. The Lowe ratio is defined as:

$$ratio(\mathbf{x}_i) = \frac{d\left(desc(\mathbf{x}_i), desc(\mathbf{x}'_{j'})\right)}{d\left(desc(\mathbf{x}_i), desc(\mathbf{x}'_{i'})\right)}.$$

To reduce the sensitivity to repetitive patterns, only features with a Lowe ratio smaller than a threshold (typically 60%, or up to 80% to have more matches) are considered as matches.

Upper bound criterion

An upper bound value for descriptor differences can also be used: only matches with dissimilarity smaller than this upper bound value will be considered.

Fast approximate nearest neighbors

The FNN approach requires a comparisons with every possible candidate, which means $n \times n'$ descriptor comparisons, which is computationally burdensome. Muja and Lowe [64] apply a kD-tree structure to efficiently identify approximate nearest neighbors. The partition on binary tree decreases the number of comparisons at the cost of matching accuracy.

3.7.2 Global matching strategy

Matches obtained from local methods often contain numerous false correspondences (mismatches), due to the limited use of local information only. In fact, too strict criteria would lead to too few selected matches (still with mismatches), and too loose criteria includes a lot of mismatches, see Figure 3.9. In challenging conditions, a local matching strategy itself cannot produce satisfying matches, a global matching strategy is thus needed. Still it is possible to apply a loose local matching strategy in order to accept more correct correspondences, and leave the task of remove mismatches to a global matching strategy.

Being aware of the limitation of local matching strategies, researchers have looked for global constraints that may help improving the correspondences. Typically, the question addressed is: given an initial group of matches produced by local matching, how to remove as many false matches to outliers as possible while keeping as many correct ones in inliers as possible? Two main approaches have been developed to solve this problem. If the scene has a rigid structure, then the epipolar constraint can be applied. Thus one solution is to try to fit 3D camera position hypotheses to the scene with epipolar constraint (or homography constraint in planar situation).

If there is no obvious rigid structure, people try to consider the constraints between matches, which leads to a graph matching problem. Though graph matching methods are more complex and have a reduced usage compared to model fitting in the case of SfM, these two approaches can compensate their defects and have better results when combined.

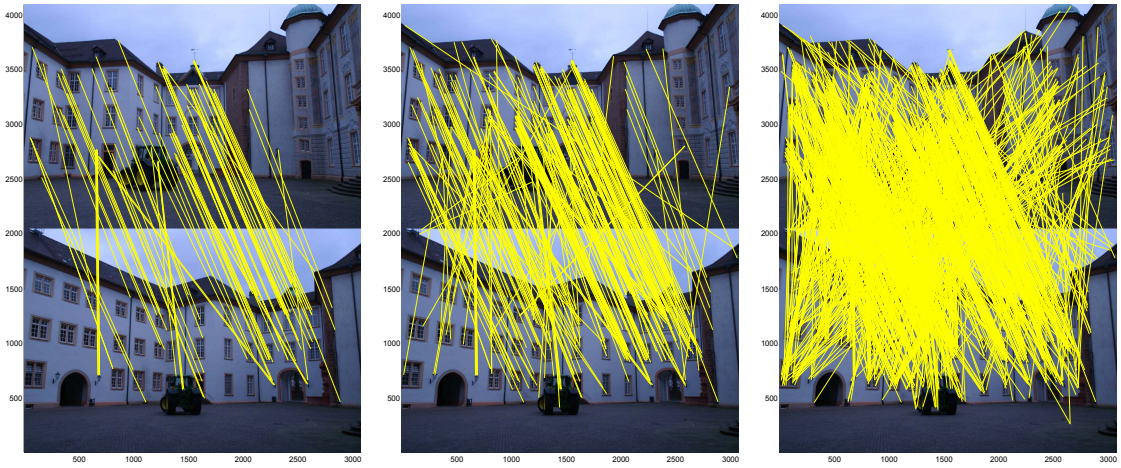


Figure 3.9: Result of local SIFT matching: The symmetric first nearest neighbor (FNN) strategy is applied in each case. Left: Lowe ratio=0.6, middle: Lowe ratio=0.8, right: Lowe ratio=0.98. Due to local matching and descriptor performance, applying a too strict threshold ratio removes many correct matches, applying a too loose strategy leads to many mismatches. In any case, it is difficult to get rid of all mismatches while keeping a sufficient number of correct matches.

3.7.3 Model fitting methods—the RANSAC family

If there is a rigid structure between both images, a possibility to remove false matches consists in trying to fit a camera position hypothesis to the scene; if the hypothesis is correct, then correct matches should satisfy the constraint. However, matches selected by local matching are contaminated by mismatches; in order to provide a good hypothesis, we will need uncontaminated match samples.

One of the first widely used algorithms to solve this chicken-and-egg dilemma is the RANdom SAmple Consensus algorithm, abbreviated *RANSAC*, by Fischler and Bolles [27]. Given a group of contaminated data samples, where correct samples satisfy a geometric model constraint, and false samples are randomly distributed in space, the RANSAC method randomly retrieves a small set of samples from the data, and estimates a hypothesis of the model. If all samples in this set are correct, the hypothesis should be close enough to the solution and thus many other correct samples should also support this hypothesis (with an error tolerance δ). By iteratively renewing sets of samples a large number of times, RANSAC probabilistically finishes by getting a close-enough hypothesis, as well as the samples supporting this hypothesis. Example of line fitting with contaminated data are shown in Figure 3.10.

The error tolerance threshold and the number of iterations are two important parameters for RANSAC methods. The error tolerance threshold should depend on the variance of inliers imprecision; a too small value leads to fewer inliers and produces an unstable/biased result; a too big value accepts many mismatches thus degrades the final results. The probability P (level of confidence) to return a right hypothesis depends on the number of iterations K, the number n of samples to generate an hypothesis, and the ratio ρ of inliers in the contaminated data. Since RANSAC is a random process, the more iterations we perform, the better chance it has of returning a good result. The relation between P, K, n and ρ can be expressed as:

$$1 - P = (1 - \rho^n)^K.$$

In this simple setting, the required number of iterations w.r.t. to an expected confi-

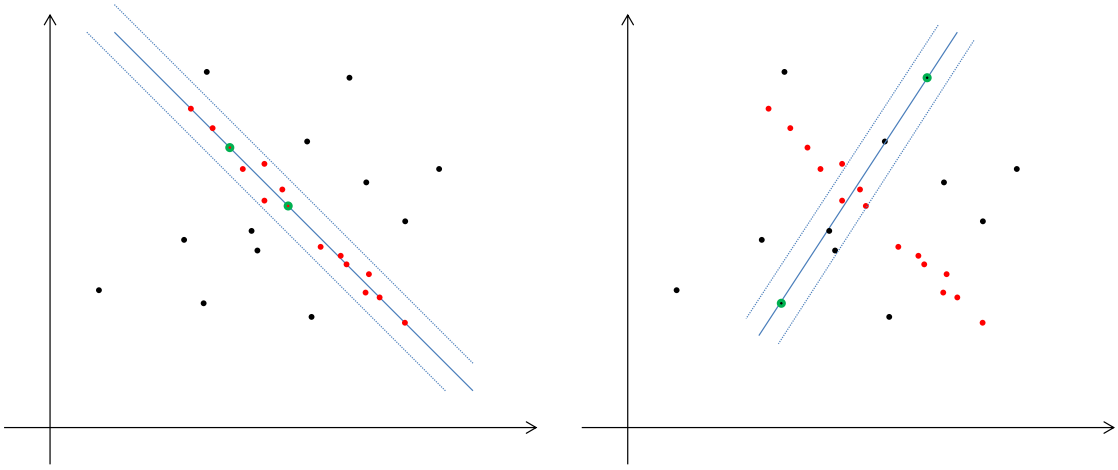


Figure 3.10: RANSAC method for line fitting: Inliers are in red, outliers are in black. Two hypotheses have been generated based on points in green. Left: the hypothesis based on two inliers is close enough to the right solution, and thus has many supporting points in the tolerance range. Right: the hypothesis generated by contaminated data has fewer supporting point in the tolerance range.

dence P is thus:

$$K = \frac{\log(1-P)}{\log(1-\rho^n)} \tag{3.9}$$

P is usually a number close to 1 (not equal) and n is fixed by the nature of the geometric model required to generate an hypothesis. Only δ and ρ require extra knowledge about the data samples, although some RANSAC variants try to estimate them. The basic RANSAC algorithm is depicted in Figure 3.11.

Application to fundamental matrix estimation

We leave the model hypothesis refinement step to Section 3.8. RANSAC has been widely used in stereo vision for Structure from Motion. Data samples are input matches $m_i = (\mathbf{x}_i, \mathbf{x}'_i)$, the hypothesis can be a fundamental matrix, an essential matrix or a homography matrix. (Here feature $\mathbf{x}'_{i'}$ matching features $\mathbf{x}_i$ are re-indexed as $\mathbf{x}'_i$ to simplify notations.) We give details here for the fundamental matrix estimation as it is the most common case in SfM. First, a hypothesis of the fundamental matrix is represented in vector form as:

$$H = \begin{bmatrix} f_1 \\ f_2 \\ . \\ . \\ . \\ f_9 \end{bmatrix}, \text{ where } \mathbf{F} = \begin{bmatrix} f_1 & f_2 & f_3\zeta \\ f_4 & f_5 & f_6\zeta \\ f_7\zeta & f_8\zeta & f_9\zeta^2 \end{bmatrix}, \tag{3.10}$$

with ζ a conditioning parameter. (Note that H represents here a hypothesis, not a homography.) If the match m_i satisfies the epopolar constraint defined by $\mathbf{F}$, then we have

$$\tilde{\mathbf{x}}_i'^T \mathbf{F} \tilde{\mathbf{x}}_i = 0.$$

with $\tilde{\mathbf{x}}_i = \begin{pmatrix} u_i \\ v_i \\ 1 \end{pmatrix}$ and $\tilde{\mathbf{x}}'_i = \begin{pmatrix} u'_i \\ v'_i \\ 1 \end{pmatrix}$; we can expand this to the scalar equation

$$f_1 u'_i u_i + f_2 u'_i v_i + f_3 u'_i \zeta + f_4 v'_i u_i + f_5 v'_i v_i + f_6 v'_i \zeta + f_7 u_i \zeta + f_8 v_i \zeta + f_9 \zeta^2 = 0. \tag{3.11}$$

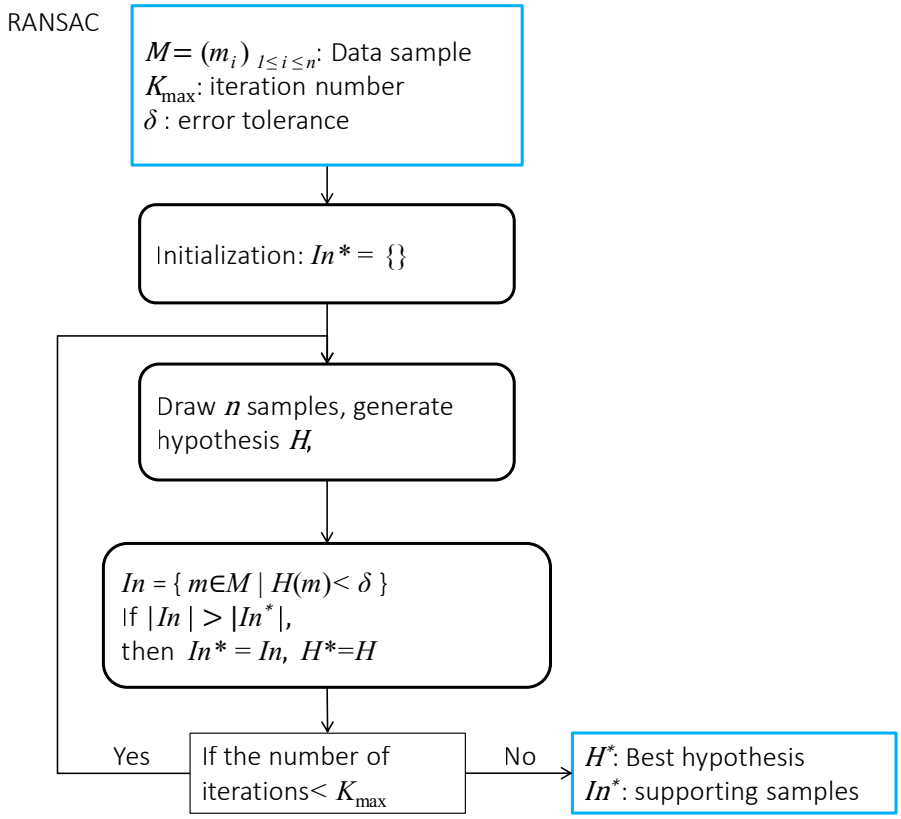


Figure 3.11: Basic RANSAC algorithm. Note that H represents here a hypothesis, not a homography. $H(m)$ represent the fitting error of the match m with H.

We note the vector form of match m_i as:

$$w_i^T = \begin{bmatrix} u'_i u_i & u'_i v_i & u'_i \zeta & v'_i u_i & v'_i v_i & v'_i \zeta & u_i \zeta & v_i \zeta & \zeta^2 \end{bmatrix} \tag{3.12}$$

As the fundamental matrix is non-null and invariant to re-scaling, we constrain the equation by $H^T H = 1$.

Either 7 or 8 random matches are used to generate a hypothesis H according to different methods for solving the equation. For example, for the 7-point algorithm:

$$\begin{bmatrix} w_{i1}^T \\ w_{i2}^T \\ \vdots \\ w_{i7}^T \end{bmatrix} H = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \text{where } (i1, i2 \ldots i7) \text{ are random indices.} \tag{3.13}$$

(This leaves one degree of freedom in H, which can be solved using the contraint $\det F = 0$.) If a match m_i supports the hypothesis H, then we have:

$$w_i^T H \le \delta. \tag{3.14}$$

Variants

There exists a very rich family of RANSAC variants aiming at different purposes. We just list a few examples here. More details can be found in [19, 67].

- Fast convergence: Without additional information, the retrieval of small n data is uniform among all input data, thus every generated hypothesis is independent

from the others. Several methods have been proposed to accelerate the generation of a correct hypothesis. Moisan and Stival [57] propose to retrieve samples from supporting points of the last valid hypothesis, which quickly leads to a correct result. Chum and Matas propose the Prosac method [52], which ranks data samples according to a confidence function, and generates hypotheses with most confident samples as a priority. The confidence function determines the gain in convergence. Parameters as the descriptor differences can be used as the confidence function.

- Robustness: There are many variants to increase the robustness of RANSAC methods. MLESAC, by Torr et Zisserman [77], introduces a probabilistic measure for hypothesis quality supposing the mismatch distribution is uniform and the correct matches are Gaussian distributed. ORSA, by Moisan and Stival [57], applies an "a contrario" approach to the measure of hypothesis quality, supposing that feature distribution in images is uniform. This method automatically selects ρ during iterations.

- Accuracy: LO-RANSAC by Chum et al. [21] proposes to refine the generated hypothesis during iterations, which is also the case for ORSA method.

Limitation of RANSAC methods

There are two major limitations for RANSAC-like methods. The first one is the number of iterations. Since the RANSAC iteration number depends on the ratio of inliers ρ, the required iteration number K increases dramatically when ρ drops below 50%. The second limitation is related to the epipolar geometry. As mentioned before, knowing the camera position and a point $\mathbf{x}$ in image I, any point on the corresponding epipolar line in image I' can satisfy the constraint. Thus, if H^* is a correct hypothesis and $\mathbf{x}'_{i'}$ is a false match of $\mathbf{x}_i$ on the epipolar line, $m = (\mathbf{x}_i, \mathbf{x}'_{i'})$ still satisfies the condition $H^*(m_i) < \delta$. RANSAC-like methods are unable to remove this type of outliers. In Chapter 4, we will address this problem with our K-VLD method.

3.7.4 Graph matching methods

Graph matching methods are other tools to generate feature correspondences, which globally optimize a match consistency. They can be used even with non rigid scenes. The idea is to construct a graph for each image where vertices are features and edges are pairwise relations between features. Graph matching methods try to establish a vertex correspondence between two graphs, satisfying matching constraints or optimizing a global score. The feature descriptor similarity is expressed via vertex similarity, and higher order constraints are represented via hyper-edges. More details can be found in the review [22]. For graph matching methods, the nature of constraints between features has a direct impact on the result. However, this issue has little been addressed.

Pairwise constraints

Second-order graph matching methods, such as [18, 44], use point distances. Alexander et al. [9] combine the distance with the orientation information. Albarelli et al. [3] use both feature orientation and scale to predict the projection of neighbor features, which is a more elaborate and fruitful constraint.

Higher order constraints

Some methods look for higher order constraints (involving more than two vertices) to gain accuracy and robustness to noise. A usual approach is to express the triangle similarity as a 3^{rd}-order constraint [43, 68, 16]. A 4^{th}-order constraint can be used to model a local affine transformation [24]. Even higher order constraints can express projective-invariant potentials [24].

Limitation of graph matching methods

As far as we know, most practical constraints are purely based on geometric relations, which are brittle under perspective transformation. It supposes that the structure in the two images remains rigid. We believe that a combination with photometric information can boost the performance of any graph matching methods, to the point that a simple 2^{nd}-order matching method could yield very good results.

For high-order graph matching, the running time and memory consumption are a major issue, especially for large datasets (images with hundreds or thousands of features): the complexity is at least $O(N^d)$ where N is the number of points and d the order.

Besides, graph matching methods are not well suited to remove numerous mismatches; the inlier rate is assumed to be relatively large. For instance, Lee et al. [43] only describe experiments with at least 50% of inliers (and at most 60 points), and Duchenne et al. [24] show a severe drop of performance when the inlier rate falls below 30% (with less than 100 points). In Chapter 4, we will discuss our solution to these limitations.

For these reasons, SfM uses essentially a model fitting approach to remove mismatches, and graph matching methods are used more as an optional intermediate step.

3.8 Two-view camera calibration

3.8.1 Model refinement

RANSAC-like methods remove mismatches and propose a rough model hypothesis, which can be used to recover camera positions. A more accurate model can then be computed based on all the inliers. We keep using here the previous example of fundamental matrix. Torr and Murray [78] have presented various existing solutions to refine the fundamental matrix with accuracy in mind. Kanatani and Sugaya [36] look for an optimal solution with a more complex form. We only present the naive method and the iterative re-weighted least square method (IRLS), which presents a close to optimal performance with a simple implementation. We suppose a preceding RANSAC method produces an inlier set $\{m_i | i \in \{1,2\ldots,n\}\}$.

Naive method: minimization of an algebraic error

The naive method minimizes the sum of the square algebraic error for inliers

$$H_{opt} = \underset{H,\|H\|=1}{\arg\min} \sum_{i=1}^{n} (w_i^T H)^2,$$

or in matrix form

$$H_{opt} = \underset{H,\|H\|=1}{\arg\min} H^T (W^T W) H \tag{3.15}$$
$$\text{with } W = \begin{bmatrix} w_1^T \\ \vdots \\ w_n^T \end{bmatrix}.$$

Equation (3.15) is a classic optimization problem that admits as solution an eigenvector of $W^T W$ (9×9 matrix) associated to its smallest eigenvalue.

Iterative re-weighted least square method: minimization of the geometric error

The naive method is fast, but produces a sub-optimal estimate of $\mathbf{F}$. It is better to minimize the sum of square distances to epipolar lines (the geometric error), see Figure 3.12. To do that, we first calculate the epipolar line equations for both images:

$$l_i = \mathbf{F}^T \tilde{\mathbf{x}}'_i = \begin{bmatrix} a_i \\ b_i \\ c_i \end{bmatrix} \text{ and } l'_i = \mathbf{F}\tilde{\mathbf{x}}_i = \begin{bmatrix} a'_i \\ b'_i \\ c'_i \end{bmatrix}. \tag{3.16}$$

Given that the formula of the distance from a point (x_0, y_0) to a line $ax + by + c = 0$ is

$$\frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}},$$

we compute the square distance from the point $\tilde{\mathbf{x}}_i$ to line l_i by

$$d(l_i, \tilde{\mathbf{x}}_i)^2 = \frac{(\tilde{\mathbf{x}}_i^T l_i)^2}{a_i^2 + b_i^2} = \frac{(\tilde{\mathbf{x}}_i^T \mathbf{F}^T \tilde{\mathbf{x}}'_i)^2}{a_i^2 + b_i^2} = \frac{H^T (w_i w_i^T) H}{a_i^2 + b_i^2}.$$

Thus for m_i, the sum of the square epipolar distances in the two images is

$$H^T (\sum_i c_i^2 w_i w_i^T) H, \text{ with } c_i = \sqrt{\frac{1}{(a_i^2 + b_i^2)} + \frac{1}{(a_i'^2 + b_i'^2)}}.$$

We note

$$W_e = \begin{bmatrix} c_1 w_1^T \\ \vdots \\ c_n w_n^T \end{bmatrix}. \tag{3.17}$$

Then H_{opt} is an eigenvector of $W_e^T W_e$ associated to its smallest eigenvalue. However, the coefficients c_i depend on H. We thus iterate several times the optimization process; at every iteration, c_i are updated with the latest H_{opt}. Other similar definition for c_i have been proposed. For more details, we refer to Torr and Murray's work [78].

3.8.2 Essential matrix decomposition

Once the fundamental matrix is refined, it is used to compute the essential matrix. The SVD decomposition of an essential matrix gives:

$$\mathbf{E} = \mathbf{U}\Sigma\mathbf{V}^T \text{ with } \Sigma = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{3.18}$$

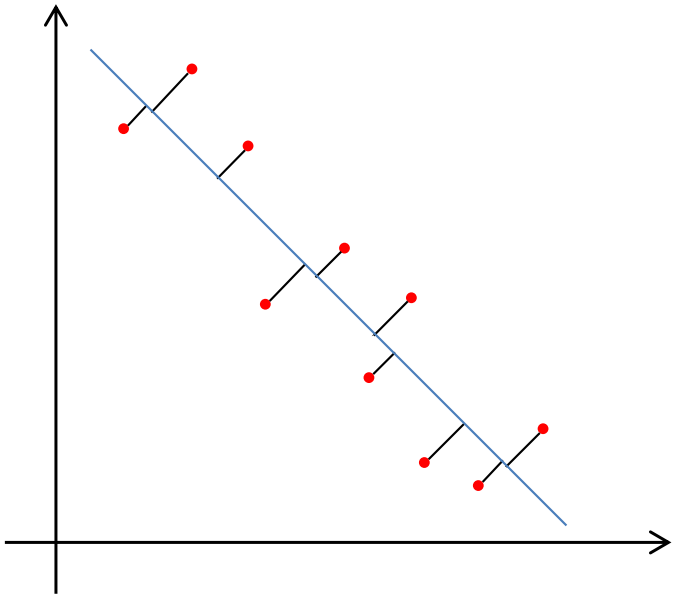


Figure 3.12: At each iteration, the IRLS method tries to approximate the sum of the square geometric error with the estimated model by re-weighting input samples before generating the next model.

The camera's rotation and translation can be expressed as follows:

$$\begin{aligned} [\mathbf{t}]_\times &= \pm \mathbf{V}\mathbf{G}\Sigma\mathbf{V}^T \\ R &= \pm \mathbf{U}\mathbf{G}^{-1}\mathbf{V}^T. \\ \text{with } \mathbf{G} &= \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{3.19}$$

The sign of the rotation matrix is determined by $\det R = 1$ and only one solution of $\mathbf{t}$ will project features in front of both cameras.

3.8.3 Bundle adjustment

An alternative to determine the camera positions and reconstruct the 3D points is the bundle adjustment, first introduced by Triggs et al. [79]. It can be applied to the two-view SfM problem as well as in the N-view case. The idea is to consider as variables the camera projection matrices $\{\mathbf{P}_j | j \in \{1 \dots n_c\}\}$ and 3D point positions $\{\mathbf{X}_i | i \in \{1 \dots n_p\}\}$, and iteratively refine these variables via a Levenberg-Marquardt process. We note $\mathbf{x}_i^j$ the observation of feature $\mathbf{X}_i$ by the j^{th} camera, $\bar{\mathbf{x}}_i^j$ the projection of $\mathbf{X}_i$ to the j^{th} image so that $\bar{\mathbf{x}}_i^j \sim \mathbf{P}_j \mathbf{X}_i$, and $\delta_{i,j}$ is defined as:

$$\delta_{i,j} = \begin{cases} 1 & \text{if } \mathbf{X}_i \text{ is seen by the } j^{th} \text{ camera;} \\ 0 & \text{otherwise.} \end{cases} \tag{3.20}$$

The minimized function in bundle adjustment is the sum of squared reprojection errors:

$$\underset{\{\mathbf{P}_j\},\{\mathbf{X}_i\}}{\arg\min} \sum_{i=1}^{n_p} \sum_{j=1}^{n_c} \delta_{i,j} (\mathbf{x}_i^j - \bar{\mathbf{x}}_i^j)^2 \text{ with } \bar{\mathbf{x}}_i^j \sim \mathbf{P}_j \mathbf{X}_i. \tag{3.21}$$

This algorithm requires an initialization that is close enough to the optimum, otherwise it could return a sub-optimal result.

3.9 N-view camera calibration

The calibration of the camera positions for multiple images usually goes through the following steps: first, a two-view camera calibration process is performed for every possible image pair. If the two-view calibration succeeds, it tries to establish the pair-wise spatial relationship with the feature correspondences. Second, the matches that potentially point to a same 3D point are grouped together to form feature tracks and initial global positions are computed for each image according to some algorithm. Last, a pose refinement process is performed. For this, their exist two approaches: incremental or global refinement.

3.9.1 Incremental methods

The incremental approach tries to add images one after another, refining the position of already-calibrated cameras as new images are added. Bundler [72] is a well-known method and system following this schema. This approach may suffer from drift errors.

3.9.2 Global methods

The global approach tries to deal with all images at the same time to avoid drift errors and better deal with image mismatches. The global method proposed by Moulon et al. [62] shows its advantage in accuracy. It is however more demanding in terms of computing power and memory. It is thus better suited to a smaller number of images.

As our work does not focus on N-view SfM, a more complete study of N-view camera calibration is out of the scope of this thesis.

Chapter 4

Feature correspondence

In the early stage of Structure From Motion, finding reliable correspondences between sets of features in two images has an important impact on the quality of output reconstruction. Despite extensive work in this domain, difficult but common cases like repetitive patterns or poor texture are still persistent problems for feature correspondence. In this chapter, we present our contribution in improving the robustness of feature matching, and its application in various domains, other than structure from motion.

Contents

4.1 Introduction 49
4.1.1 Feature matching by RANSAC 49
4.1.2 Graph matching methods 49
4.1.3 Region growing methods 50
4.2 Our contributions in matching 50
4.2.1 Robust 2nd-order photometric criterion 50
4.2.2 Light semi-local matching strategy 51
4.3 Virtual line descriptor (VLD) 51
4.3.1 Geometric consistency 52
4.3.2 Line covering 52
4.3.3 Inter-point gradient histogram 53
4.3.4 Inter-point orientation 53
4.3.5 Distance between two VLDs 55
4.3.6 VLD-consistency 55
4.3.7 High contrast suppression 55
4.4 K-VLD: a K-connected VLD-based matching method 56
4.4.1 Neighborhoods 57
4.4.2 Problem statement 58
4.4.3 Algorithm 59
4.4.4 Optimizations and heuristics 59
4.5 Evaluation 60
4.5.1 Changing imaging conditions 60
4.5.2 Strong occlusions 60
4.5.3 Ambiguity and RANSAC prefiltering 64
4.5.4 Comparison of ASIFT and K-VLD 64
4.6 Parameters 66

4.7 K-VLD's contribution for N-view SfM 68
4.8 Running time 69
4.9 Limitations of VLD and K-VLD methods 70
4.9.1 Limitation w.r.t. detection inaccuracies 70
4.9.2 Limitation w.r.t. repetitive patterns 70
4.10 Conclusion 71

4.1 Introduction

Finding reliable correspondences between sets of feature points in two images is a key step in a number of computer vision problems, e.g., camera calibration and object recognition. To achieve this task, feature detectors such as SIFT [49], SURF [8], Harris-affine [54] or MSER [52] identify interest points or areas in images robustly. By design, the detected points or areas are salient enough to be likely also salient in other views of the same scene, under different imaging conditions (viewpoint, lighting, orientation, scale, etc.).

Besides, these points or areas can be individually described based on their scale, if any, as well as on an abstraction of their photometric neighborhood, e.g., based on the distribution of local gradients. Such feature descriptors include SIFT [49], SURF [8] and MSER shape descriptor [28]. Like detectors, these descriptors are designed to be robust, to some extent, to variations such as noise or change of viewpoint, orientation or illumination.

Matching detected features in two images based on the similarity of their descriptor often provides good correspondences . However, it also includes false matches. Eliminating those false matches while preserving true correspondences remains challenging for images with ambiguities or strong transformations. Ambiguity usually arises from repetitive patterns (e.g., facade windows) or lack of texture. In this case, the descriptors are not discriminative enough to safely differentiate feature points. There actually is a balance to find as repeatable descriptors tend to be less distinctive, and vice versa. As for strong transformations, they can sometimes be avoided by carefully controlling imaging conditions. Yet some sharp transformations cannot be avoided, e.g., due to strong occlusions, when a foreground object obstructs very different background areas. To get both a high number of correct matches and a low mismatch ratio, just comparing individual feature descriptors is not enough. Global methods are required, such as RANSAC or graph matching.

4.1.1 Feature matching by RANSAC

For rigid transformations, RANSAC-like methods [27] can accurately separate inliers from outliers. They randomly sample subsets of correspondences to build a putative model of the transformation (fundamental/homography matrix) and count the number of matches that are compatible with the model. The largest consensus set defines what is to be considered as inliers, other matches being regarded as outliers.

This works well if the inlier rate ρ is high, not if it is low. The reason is that the number of required sampling iterations is on the order of $1/\rho^n$, where n is the number of correspondences to draw to define a model, cf. Section 3.7.3. (In general, for the fundamental matrix, $n = 7$ or 8 [34].) Better drawing strategies such as MLESAC [77] or PROSAC [20] can greatly reduce the number of models to sample, but they are nonetheless not well suited for inlier rates lower than 50%. Only a few methods such as ORSA [57] can treat an inlier rate of 10%. Yet in any case, all RANSAC-like methods inherently suffer from a limitation when estimating the fundamental matrix: they cannot eliminate mismatches corresponding to points that have matches near their epipolar line but far from the correct location, which may degrade precision.

4.1.2 Graph matching methods

Graph matching is also a tool to determine feature point correspondences, with a global consistency criterion. It applies not only to rigid scenes but also to deformable objects.

The idea is to construct a graph where vertices are feature points and edges are pairwise relations. Higher-order constraints, involving more than two vertices, can be modeled as hyperedges. Graph matching methods try to establish a vertex correspondence between two graphs, satisfying matching constraints or optimizing a global score. Some can also handle inexact matching, allowing different structures to some extent [22].

For 2nd-order graph matching, many methods use the relative distance between points as constraint [18, 44], possibly in combination with angles [9]. Feature orientation and scale are used too, e.g., to define an affine transformation predicting the projection of neighboring points [3]. Some robust pairwise descriptors combine individual feature descriptors too [30].

A better matching accuracy or robustness to noise can be achieved with higher order graph matching [43, 68, 16]. A common 3rd-order constraint expresses triangle similarity [43]. 4th-order constraints typically include consistency w.r.t. a local affine transformation [88, 24]. Graph matchers supporting edges of even higher-order can for instance also express projective-invariant potentials [24]. However, despite recent advances in higher-order graph matching, the running time and memory consumption remain an issue, especially for large datasets (images with hundreds or thousands of features): the complexity is at least $O(N^d)$ where N is the number of points and d the order of the constraints. Besides, although some methods explicitly include a treatment of outliers, e.g., using absorbing nodes [43], the inlier rate is still assumed to be relatively large. For instance, Lee et al. [43] only describe experiments with at least 50% inliers (and at most 60 points), and Duchenne et al. [24] show a severe drop of performance when the inlier rate falls below 30% (with less than 100 points).

Last but not least, without a strong consistency criterion, graph matching methods follow either the "match until conflict" strategy, i.e., they search the maximum number of no-conflicting matches, or "match ordering" strategy, i.e., consistent matches have a higher ranking without a clear and explicit separation of inlier/outlier. Both strategies may work for small number of matches but are prone to overlook outliers at larger scales.

4.1.3 Region growing methods

Match propagation is an approach to deal with ambiguous feature correspondences [42, 17, 26, 65]. It propagates matches from seeds to their neighbors according to the local transformation consistency. The performance of these algorithms depends on the distinctiveness of pairwise or higher-order constraints. Ok et al. [65] mention that the geometric pairwise constraints based on a local feature transformation is experimentally very noisy; this is why they use 4-th order constraints.

4.2 Our contributions in matching

4.2.1 Robust 2nd-order photometric criterion

Feature descriptors provide 1st-order photometric information to estimate correspondence likelihood and identify potential matches. All other information used for matching is generally restricted to geometric information, i.e., relative point location in the image. This is the case for RANSAC methods and for most existing graph matchers. Although some authors mention possible extensions of graph matching potentials to photometric information [24], such uses are scarce and tend to translate into quasi-dense matching [26]. For instance, experiments in [88, 44, 3, 23, 43, 68, 16] are limited to

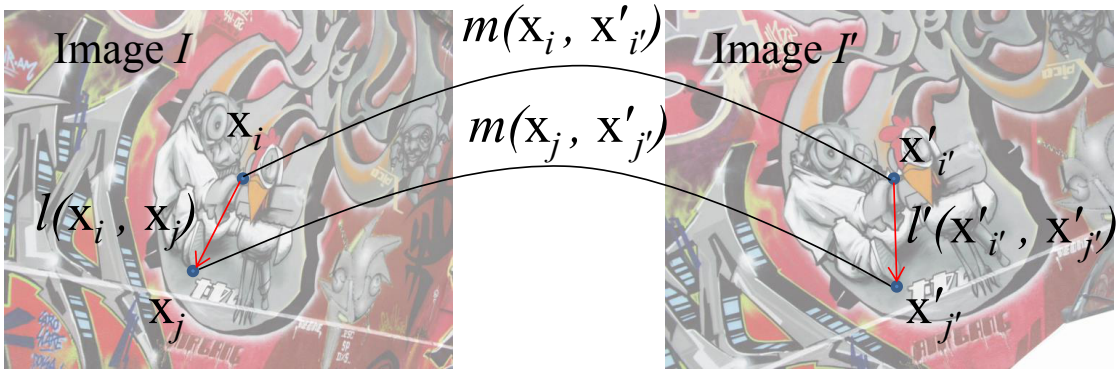


Figure 4.1: General idea of 2nd-order photometric criterion: information along oriented segments $(\mathbf{x}_i, \mathbf{x}_j)$ and $(\mathbf{x}'_{i'}, \mathbf{x}'_{j'})$ is unlikely to be similar unless both matches $(\mathbf{x}_i, \mathbf{x}'_{i'})$ and $(\mathbf{x}_j, \mathbf{x}'_{j'})$ are correct.

geometric relations as triangle similarity, arc length, descriptor's scale and orientation. Photometric information between features is ignored.

We propose here a novel, simple and efficient, 2nd-order photometric criterion. It is based on the fact that for points $\mathbf{x}_i, \mathbf{x}_j$ in image I and $\mathbf{x}'_{i'}, \mathbf{x}'_{j'}$ in image I', it is unlikely to find similar photometric information around lines $(\mathbf{x}_i, \mathbf{x}_j)$ and $(\mathbf{x}'_{i'}, \mathbf{x}'_{j'})$ unless both $(\mathbf{x}_i, \mathbf{x}'_{i'})$ and $(\mathbf{x}_j, \mathbf{x}'_{j'})$ are correct matches, see Figure 4.1. Photometric similarity between lines is stronger when the 2D lines actually lie on a 3D plane which is not occluded, but some similarity still remains under moderate surface curvature and occlusion.

To express this property, we define a virtual line descriptor (VLD) that captures photometric information between two points. The distance between two such descriptors measures the dissimilarity between the corresponding two virtual lines. It can be used in the 2nd-order term of graph matchers to improve their accuracy.

4.2.2 Light semi-local matching strategy

Thanks to the robustness of our VLD used as 2nd-order photometric criterion, we also define a semi-local matching strategy based on VLD which is a light version of graph matching using photometric information. It can be used as a preprocessor to RANSAC methods to improve the quality of match selection by considerably increasing the inlier rate before RANSAC. As it can eliminate false matches near epipolar lines, it greatly improves precision. As the inlier rate is improved, the needed number of iterations in RANSAC can be considerably reduced.

4.3 Virtual line descriptor (VLD)

As far as we know, when it comes to studying the consistency of a pair of matches, existing pairwise constraints are mostly based on geometry only. However the pure geometric constraint lacks distinctiveness. A pair of features $(\mathbf{x}_i, \mathbf{x}_j)$ in I is unstable under perspective transformation in image I' and could have several occurrences $(\mathbf{x}'_{i'}, \mathbf{x}'_{j'})$ having a high consistency score. The use of pairwise photometric information along the path $(\mathbf{x}_i, \mathbf{x}_j)$ avoids spurious matching and thus is more robust.

The general idea of our descriptor for a virtual line between points $\mathbf{x}_i$ and $\mathbf{x}_j$ is to describe its photometric information by a number of SIFT-like gradient histograms. To ensure the robustness of this description, we consider a regular covering, with some

overlap, of an image strip between $\mathbf{x}_i$ and $\mathbf{x}_j$, and use a SIFT-like descriptor to characterize each element of the covering. The global line descriptor is the concatenation of the descriptors of each covering element. It inherits SIFT descriptor's robustness to noise and changes of scale, orientation and illumination.

4.3.1 Geometric consistency

Before describing a line, we actually first check a geometric constraint, extending that of Albarelli et al. [3].

Given matches $m_{i,i'} = (\mathbf{x}_i, \mathbf{x}'_{i'})$ and $m_{j,j'} = (\mathbf{x}_j, \mathbf{x}'_{j'})$, and assuming that the local transformation around $\mathbf{x}'_{i'}$ is close to a similarity, we define the point $\mathbf{p}'_{j'}$ in image I' as the expected position of $\mathbf{x}'_{j'}$ (cf. Figure 4.2):

$$\mathbf{p}'_{j'} = \mathbf{x}'_{i'} + \frac{s(\mathbf{x}'_{i'})}{s(\mathbf{x}_i)} R(a(\mathbf{x}'_{i'}) - a(\mathbf{x}_i)) \overrightarrow{\mathbf{x}_i \mathbf{x}_j}, \tag{4.1}$$

where $s(\mathbf{x})$ is the scale of feature point $\mathbf{x}$, $a(\mathbf{x})$ is the angle of the main orientation at $\mathbf{x}$, and $R(\alpha)$ is the rotation of angle α. Permuting I and I' defines a point $\mathbf{p}_j$ in image I as the expected position of $\mathbf{x}_j$. The transformation error of $(\mathbf{x}_j, \mathbf{x}'_{j'})$ by $(\mathbf{x}_i, \mathbf{x}'_{i'})$ is measured in I based on distances $d_{i,j} = dist(\mathbf{x}_i, \mathbf{x}_j)$, $t_{i,j} = dist(\mathbf{x}_i, \mathbf{p}_j)$, $e_{i,j} = dist(\mathbf{x}_j, \mathbf{p}_j)$, and likewise in I'.

The normalized and symmetrized score of geometric consistency (smaller means more consistent) for matches $m_{i,i'}$ and $m_{j,j'}$ is defined as:

$$\chi(m_{i,i'}, m_{j,j'}) = \min(\eta_{i,i',j,j'}, \eta_{j,j',i,i'}) \tag{4.2}$$

$$\text{where} \quad \eta_{i,i',j,j'} = \frac{e'_{i',j'}}{\min(d'_{i',j'}, t'_{i',j'}).} = \eta_{i',i,j',j} = \frac{e_{i,j}}{\min(d_{i,j}, t_{i,j})}$$

Matches $m_{i,i'}$ and $m_{j,j'}$ are considered as *consistent w.r.t. geometry* iff $\chi(m_{i,i'}, m_{j,j'}) < \chi_{\max}$. In all our experiments, we use a threshold value $\chi_{\max} = 0.5$. This fast prefiltering step eliminates many false matches before photometric comparison while preserving most good matches.

Comparison with other geometric constraints

Our pairwise geometric constraint improves over other existing ones. [88] uses a function of the difference of distance between $(\mathbf{x}_i, \mathbf{x}_j)$ and $(\mathbf{x}'_{i'}, \mathbf{x}'_{j'})$ such as $e^{-|dist(\mathbf{x}_i,\mathbf{x}_j) - dist(\mathbf{x}'_{i'},\mathbf{x}'_{j'})|}$, [44] uses $4.5 - \frac{(dist(\mathbf{x}_i,\mathbf{x}_j) - dist(\mathbf{x}'_{i'},\mathbf{x}'_{j'}))^2}{2\sigma^2}$ for some σ. Albarelli et al. [3] propose in their work a better constraint as $e^{-\lambda \max(e'_{i,j}, e'_{j,i})}$ (c.f. Figure 4.2 for $e'_{i,j}$), but it is neither symmetric nor invariant to scale. Our geometric constraint $\chi(m_{i,i'}, m_{j,j'})$ is a better one, with invariance to scale, and symmetry.

4.3.2 Line covering

For two points $\mathbf{x}_i$ and $\mathbf{x}_j$ in image I at distance d one from another, we consider U interpoint disks (D_u) of radius $r = \frac{d}{U+1}$ centered on points $\mathbf{x}_i + \frac{u}{U+1}\overrightarrow{\mathbf{x}_i \mathbf{x}_j}$ for $u \in \{1, \ldots, U\}$ (see Figure 4.3). Each disk is then described at image scale $s = \max(r/r_{\min}, 1)$ where $r_{\min}$ is a minimum description radius. In our experiments, we use $U = 10$ and $r_{\min} = 5$ pixels, which provides a good balance between discrimination and repeatability.

In practice, scales can be discretized and precomputed, to avoid rescaling the image for each new pair of points. As for SIFT [49], we construct a pyramid of scaled

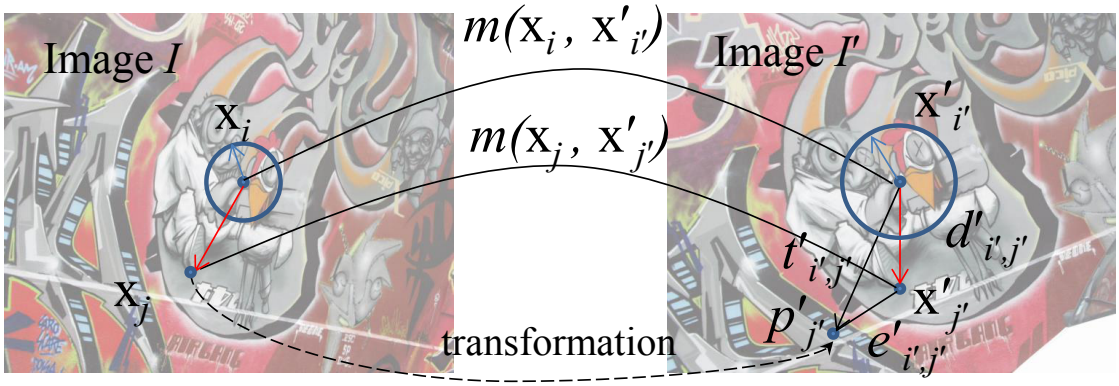


Figure 4.2: Distances used for the computation of the transformation error $\eta_{i,i',j,j'}$. It is understood as the following: according to the transformation from $\mathbf{x}_i$ to $\mathbf{x}'_{i'}$, the error ratio from the projected point $p'_{j'}$ to $\mathbf{x}'_{j'}$.

images. In our experiments, a geometric progression with ratio $\sqrt{2}$ proved enough for repeatability. For any disk radius r in the original image I, we thus use scale $s^* = 2^{q/2}$ for q natural integer such that $2^{q/2} \leq s < 2^{(q+1)/2}$, i.e., $q = \lfloor 2\frac{\log s}{\log 2} \rfloor$. In the scaled image, the disk radius is $r^* = r/s^*$ (see Figure 4.4).

4.3.3 Inter-point gradient histogram

The descriptor for disk D_u is a single SIFT-like local gradient histogram [49]. (SIFT actually defines a grid of 4×4 such histograms.) We use V bins $(h_{u,v})_{v\in\{1,\dots,V\}}$ to represent the distribution: each pixel in the disk votes in the orientation bin corresponding to its gradient (relatively to the line direction), weighted by the gradient magnitude and by a Gaussian-weighted circular window with $\sigma = \frac{3}{2}r^*$ like SIFT. In our experiments, like SIFT, we use $V = 8$. The line histograms are then normalized so that $\sum_{u=1}^{U}\sum_{v=1}^{V} h_{u,v} = 1$. (Contrary to SIFT, we use the L^1-norm rather than the L^2-norm for better discrimination.)

4.3.4 Inter-point orientation

Inter-point orientation is computed as SIFT too, with some adaptation. We construct an orientation histogram for D_u using W bins $(O_{u,w})_{w\in\{0,\dots,W-1\}}$. As slightly more variations can be expected on the line between two points than on the feature point themselves, we recommend $W > V$ (as in SIFT). In all our experiments we use $W = 24$ (whereas SIFT uses 36 bins), which intuitively improves robustnes and empirically seems to preserve enough discrimination. In addition, we treat opposite directions together and actually consider the derived histogram $(\widehat{O}_{u,w})_{w\in\{0,\dots,W-1\}}$ defined as $\widehat{O}_{u,w} = O_{u,w} - O_{u,(w+W/2) \bmod W}$ (i.e, $\widehat{O}_{u,w} = -\widehat{O}_{u,(w+W/2) \bmod W}$). This also happens to preserve enough discrimination while enhancing robustness. The main orientation w_u^* is finally defined as the bin of the derived histogram with the highest value.

$$w_u^* = \underset{w\in\{0,\dots,W-1\}}{\operatorname{argmax}} \widehat{O}_{u,w} \tag{4.3}$$

Note that $\max_{w\in\{0,\dots,W-1\}} \widehat{O}_{u,w} = \max_{w\in\{0,\dots,W/2-1\}} |\widehat{O}_{u,w}| \geq 0$. The reason why we use $\max_{w\in\{0,\dots,W-1\}} \widehat{O}_{u,w}$ instead of $\max_{w\in\{0,\dots,W-1\}} O_{u,w}$ is that, we believe the first one is more robust, as it has the biggest gap compared to its opposite side, which is more difficult to be replaced.

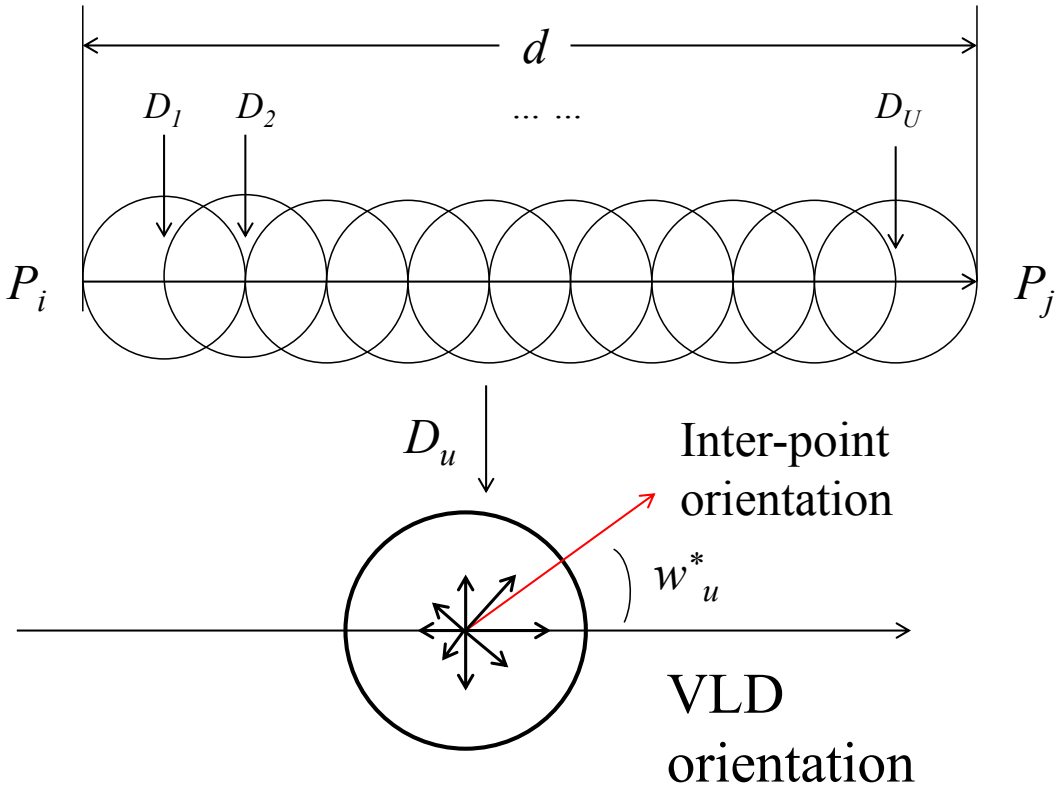


Figure 4.3: Top: disk covering of line $(\mathbf{x}_i, \mathbf{x}_j)$. Bottom: 8-bin histogram of gradient orientation for disk D_u, and main orientation w_u^*.

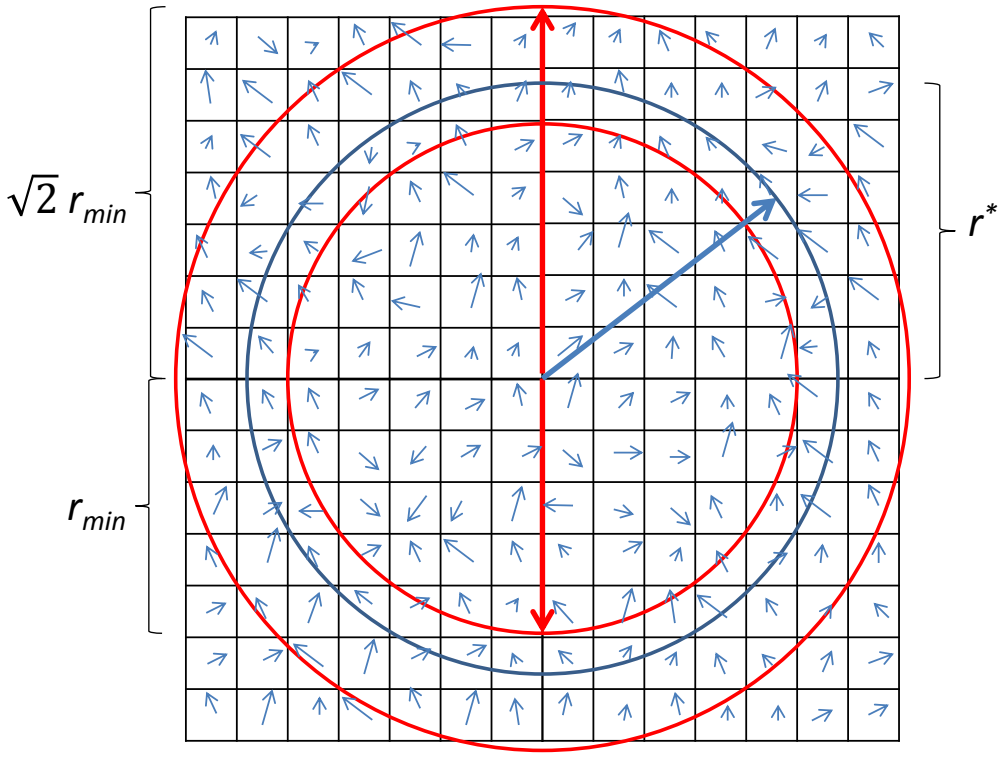


Figure 4.4: For $r \geq r_{\min}$, the VLD is computed on the q^{th} image scale on r^*-radius disks, where $r_{\min} \leq r^* = r/2^{q/2} < r_{\min}\sqrt{2}$.

We also define normalizing factors γ_u to weigh each orientation over the whole line:

$$w_u^* = \underset{w\in\{0,\dots,W-1\}}{\operatorname{argmax}} \widehat{O}_{u,w} \qquad \gamma_u = \frac{\widehat{O}_{u,w_u^*}}{\sum_{u=1}^{U} \widehat{O}_{u,w_u^*}} \geq 0 \tag{4.4}$$

Each disk D_u is represented by the histogram $(h_{u,v})_{v\in\{1,\dots,V\}}$, the orientation w_u^* and the normalizing factor γ_u. The size of the overall line descriptor is thus $U\,(V+2)$.

4.3.5 Distance between two VLDs

Given lines $l_{i,j} = (\mathbf{x}_i, \mathbf{x}_j)$ in I and $l'_{i',j'} = (\mathbf{x}'_{i'}, \mathbf{x}'_{j'})$ in I', we now define the distance between their descriptors. For each inter-point disk D_u in I and corresponding inter-point disk D'_u in I', we compute both the difference of the gradient histograms and the difference of the main orientations. Orientations w_u^* and $w_u'^*$ are compared modulo $W/2$ (most dissimilar orientation), normalized to 1, and weighted by the average of the orientation normalizing factors γ_u and γ'_u, resulting in a value in $[0,1]$. The differences of the gradient histograms and the main orientations are then linearly combined with a weighting factor $\beta \in [0,1]$:

$$\tau(l,l') = \beta \sum_{u=1}^{U}\sum_{v=1}^{V} |h_{u,v} - h'_{u,v}| + (1-\beta)\sum_{u=1}^{U}\left(\frac{\gamma_u+\gamma'_u}{2}.\frac{\min(|w_u^*-w_u'^*|, W-|w_u^*-w_u'^*|)}{W/2}\right). \tag{4.5}$$

Experimentally (see Section 4.6), we use $\beta = 0.36$, The value of β is coupled with the value of $\tau_{\max}$ (section below) to best separate consistent/inconsistent VLDs (cf. Figure 4.15).

4.3.6 VLD-consistency

The VLD-distance between matches $m_{i,i'}{=}(\mathbf{x}_i, \mathbf{x}'_{i'})$ and $m_{j,j'}{=}(\mathbf{x}_j, \mathbf{x}'_{j'})$ is the VLD-distance between the corresponding lines: $\tau(m_{i,i'}, m_{j,j'}) = \tau(l_{i,j}, l_{i',j'}) \in [0,1]$ where $l_{i,j} = (\mathbf{x}_i, \mathbf{x}_j)$ and $l'_{i',j'} = (\mathbf{x}'_{i'}, \mathbf{x}'_{j'})$. The lower τ, the more similar the virtual lines.

It can be used in the pairwise score of a graph matcher (cf. Section 3.7.4), e.g., with a contribution of the form $\exp(-\lambda\tau^2)$. Experimentally (see Section 4.6), we use $\lambda = 100$.

When a binary choice (consistent or not) is required, matches $m_{i,i'}$ and $m_{j,j'}$ are said **VLD-consistent** iff their virtual lines satisfy $\tau(m_{i,i'}, m_{j,j'}) \leq \tau_{\max}$. Experimentally (see Section 4.6), we use $\tau_{\max} = 0.35$. The matches are said **gVLD-consistent** iff they are both geometry- and VLD-consistent.

4.3.7 High contrast suppression

VLDs are discriminative when they contain a variety of gradient directions. If the gradient pattern is mostly the same on all VLD disks, then the virtual line is likely not to be discriminative because its descriptor will not vary if we shift (translate) the line along its direction. This typically occurs when the virtual line follows a highly contrasted image edge. Such an example is illustrated in Figure 4.5 where the putative matches $(\mathbf{x}'_i)$ $i \in \{1,\dots,3\}$ in I' of points $(\mathbf{x}_i)$ $i \in \{1,\dots,3\}$ in I are shifted up while staying gVLD-consistent as a group. We want to detect such a situation to prevent corresponding virtual lines from being taken into account in matching decisions.

For this reason, we define a line contrast indicator:

$$\kappa = \frac{s^*}{U\,d}\sum_{u=1}^{U} \widehat{O}_{u,w_u^*} \tag{4.6}$$

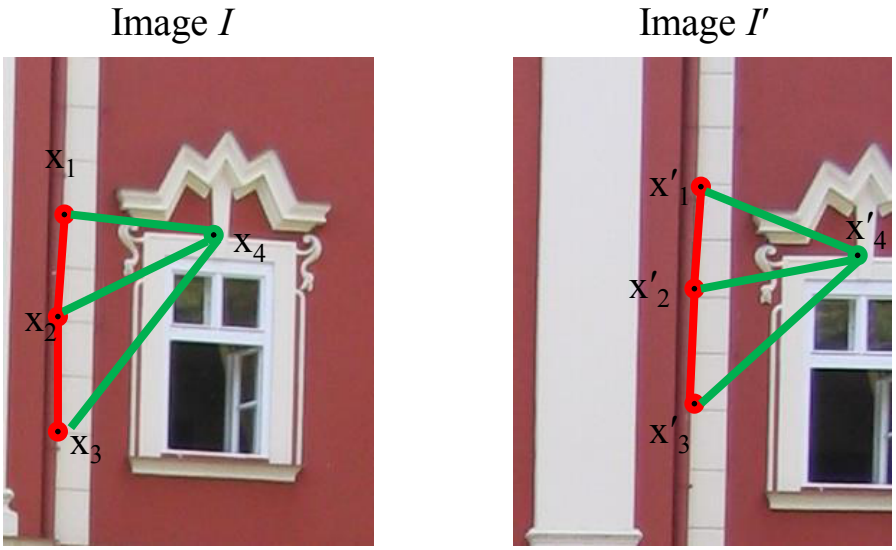


Figure 4.5: Example to illustrate the necessity of high contrast suppression: Four matches $(\mathbf{x}_i, \mathbf{x}'_i)$, $i \in \{1,2,3,4\}$ are detected. Among them, $(\mathbf{x}_i, \mathbf{x}'_i)$, $i \in \{1,2,3\}$ are along an image edge and correspond to points $(\mathbf{x}'_i)$ that are globally shifted up, only $(\mathbf{x}_4, \mathbf{x}'_4)$ is correct. However, VLDs between $(\mathbf{x}_i, \mathbf{x}'_i)$, $i \in 1\ldots3$ in red are gVLD-consistent between themselves, still we do not want to use them for match validation. Note however that VLDs outside the edges with $(\mathbf{x}_4, \mathbf{x}'_4)$ in green are still discriminative. Thus, we penalize VLDs along edges, using a form of high contrast suppression.

In equation (4.6), s^* is the re-scaling factor of the current image scale from which the VLD is computed. The gradient of a point in the image is computed based on its neighboring pixels; its intensity is inversely proportional to the observed scale, i.e., for a same region, the gradients appear sharper if the region appears smaller and vice-versa. For this reason, equation (4.6) contains a normalizing factor s^*/d.

VLDs with contrast κ above given threshold $\kappa_{\max}$ are considered unreliable and discarded. Experimentally (see Section 4.6), assuming image intensity in the range $0,\ldots,255$, we use $\kappa_{\max} = 30$.

4.4 K-VLD: a K-connected VLD-based matching method

VLD can be directly used as a pairwise constraint in 2nd or higher-order graph matching methods. Yet, existing graph matching methods do not scale well to large numbers of matches and, as shown in the experiment section, they may perform poorly when the foreground creates background occlusions. Besides, some of them are not well suited for large outlier elimination.

In this section, we introduce K-VLD, a novel matching method that overcomes these limitations. It is semi-local in the sense that the score of a match depends on its consistency with neighboring matches. The consistency is both geometric and photometric, using our VLD criterion.

The basic idea is that, given a potential match $(\mathbf{x}_i, \mathbf{x}'_{i'})$, if there are in the neighborhood of $\mathbf{x}_i$ and $\mathbf{x}'_{i'}$ at least K other matches $(\mathbf{x}_{j_k}, \mathbf{x}'_{j'_k})_{k\in\{1,\ldots,K\}}$ that are gVLD-consistent with $(\mathbf{x}_i, \mathbf{x}'_{i'})$, then $(\mathbf{x}_i, \mathbf{x}'_{i'})$ is likely to be a correct match, see Figure 4.6. The method can be seen as a simplified 2nd-order graph matcher specialized for image features. It provides a binary assessment for each match (correct or not) as well as a consistency score for further filtering.

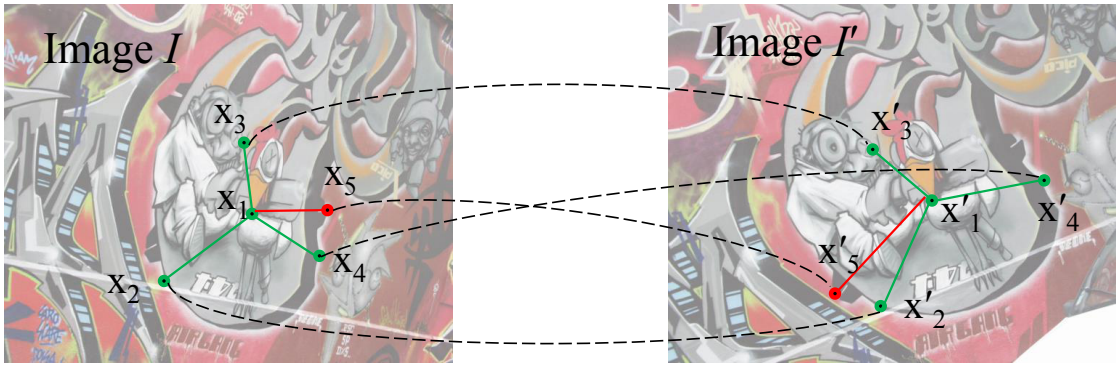


Figure 4.6: K-VLD's idea: Five matches as $m_{i,i} = (\mathbf{x}_i, \mathbf{x}'_i)$, $i \in [1 \dots 5]$ are detected, $(\mathbf{x}_5, \mathbf{x}'_5)$ is a mismatch. We suppose they are all neighbors. $(\mathbf{x}_1, \mathbf{x}'_1)$ are likely to be gVLD-consistent with its neighbors if the neighbors are correct matches, i.e., having several gVLD-consistent neighbors $m_{i,i}, i \in [2 \dots 4]$ makes $m_{1,1}$ very credible as a correct match.

4.4.1 Neighborhoods

Having more than K gVLD-consistent matches makes the considered match very likely to be a correct match. It is inefficient and unnecessary to check gVLD-consistency with all other matches. In fact, the more distant the matches (i.e., the longer the virtual lines), the more likely the virtual lines are to differ. It is thus enough in practice to check K-connected gVLD-consistency only within a neighborhood of the points, to avoid quadratic complexity. We actually adapt the size of the neighborhoods to the density ρ of feature points. Neighborhoods are defined as disks centered on $\mathbf{x}_i$ and $\mathbf{x}'_{i'}$, with respective radius B and B'. Given a set $\mathcal{M}$ of potential matches between I and I', with minimum inlier rate $\rho_{\min}$, and assuming a more or less uniform distribution, then the average number of correct matches K_B in a B-neighborhood is:

$$K_B = \frac{\pi B^2}{\text{area}(I)} \rho_{\min} |\mathcal{M}|. \tag{4.7}$$

As B should be chosen such that $K_B \geq K$, we get a definition for the minimum radius B_K of the neighborhood from (4.7). Moreover, for stability reason, we exclude neighboring points $\mathbf{x}_j$ that are too close to $\mathbf{x}_i$, within $B_{\min}$ pixels. Wrapping up, we say that a match $(\mathbf{x}_j, \mathbf{x}'_{j'})$ is a neighbor of $(\mathbf{x}_i, \mathbf{x}'_{i'})$ iff $\mathbf{x}_j$ is in the $(B_{\min}, B_K)$-annulus centered on $\mathbf{x}_i$ in I, or $\mathbf{x}'_{j'}$ is in the $(B_{\min}, B'_K)$-annulus centered on $\mathbf{x}'_{i'}$ in I'. This provides a relation between the minimum radius B and the minimum number of agreeing neighbors K; one can be computed from the other one, as follows:

$$K_B = \frac{\pi (B^2 - B_{\min}^2)}{\text{area}(I)} \rho_{\min} |\mathcal{M}| \qquad B_K = \sqrt{\frac{K \, \text{area}(I)}{\pi \rho_{min} |\mathcal{M}|} + B_{\min}^2} \tag{4.8}$$

In case we discover a posteriori that $\rho < \rho_{\min}$, B_K has to be expanded (e.g., $\rho_{min} = \rho_{min}/2$) and the algorithm has to be rerun (possibly reusing information about already-found gVLD-consistent match pairs). If we still get $\rho < \rho_{\min}$ after 5 reruns ($\rho_{\min}$ is 32 times smaller than its original value), we consider there are no more matches and stop the algorithm. In all our experiments, we set $\rho_{\min} = 3\%$ and $B_{\min} = 10$ pixels. Then, given a set of matches $M \subset \mathcal{M}$ and a match $m \in M$, we define the following neighborhoods:

- $\mathbf{N}_M(m) = \{m' \in M \mid m \text{ and } m' \text{are neighbors}\}$
- $\mathbf{N}_{M,\text{geom}}(m) = \{m' \in M \mid m \text{ and } m' \text{are geometry-consistent neighbors}\} \subset \mathbf{N}_M(m)$
- $\mathbf{N}_{M,\text{gvld}}(m) = \{m' \in M \mid m \text{ and } m' \text{are gVLD-consistent neighbors}\} \subset \mathbf{N}_{M,\text{geom}}(m)$

Input: images I and I', detected features in I and I', and set of potential matches M
Output: selected subset of matches in M

1. Compute pyramid of scaled images for I and I'
2. Set up tables $t_{|M|\times|M|}$, $T_{|M|\times 1}$, $C_{|M|\times 1}$ and initially mark all matches as inliers
3. Do
4. Set T and C to zero values
5. For every inlier match m_i
6. For every inlier match m_j, a neighbor of m_i
7. If $t[m_i,m_j]$ is undefined
8. If (m_i,m_j) are gVLD-consistent
9. then $t[m_i,m_j] \leftarrow \tau(m_i,m_j)$ and $t[m_j,m_i] \leftarrow \tau(m_j,m_i)$
10. else $t[m_i,m_j] \leftarrow$ discard and $t[m_j,m_i] \leftarrow$ discard
11. If $t[m_i,m_j] \neq$ discard
12. $T[m_i] \leftarrow T[m_i] + t[m_i,m_j]$
13. $C[m_i] \leftarrow C[m_i] + 1$
14. If $C[m_i] \geq N_{\max}$ then consider next m_i (line 5)
15. $T[m_i] \leftarrow T[m_i]/C[m_i]$ if $C[m_i] \neq 0$
16. Sort inliers by increasing value of C, or increasing T if C-equal
17. (a) For every inlier m_i in sort order
18. If $C[m_i] < K$ then mark m_i as outlier
19. (a') For every inlier m_i and every inlier m_j in sort order
20. If m_j conflicts with m_i, mark as outlier the match with lowest C,
. or lowest T if C-equal
21. (b) For every inlier m_i in sort order
22. Compute the proportion ω of geometric-consistent neighbors of m_i
23. Compute the average transformation error $\bar{\chi}$ amongst neighbors of m_i
24. Mark m_i as outlier if $\omega < \omega_{\min}$ or $\bar{\chi} > \bar{\chi}_{\max}$
25. while some matches have been marked as outliers (in this iteration)
26. If $\rho < \rho_{\min}$ go back to line 3 with $\rho_{min} = \rho_{min}/2$
27. Return matches marked as inliers

Table 4.1: K-VLD filtering algorithm.

4.4.2 Problem statement

Experimentally, requiring that good matches have at least K gVLD-consistent neighbors eliminates many mismatches, but some may still remain, especially with ambiguous matches. We found that adding an extra constraint on the proportion ω of geometry-consistent neighbors and on the average score $\bar{\chi}$ of geometric consistency for neighbors helped in removing many of these remaining mismatches. Formally, given a set of potential matches $\mathcal{M}$, we look for a subset $M \subset \mathcal{M}$ such that, for all $m \in M$,

$$|\mathbf{N}_{M,\text{gvld}}(m)| \geq K \quad \text{and} \quad \left(\frac{|\mathbf{N}_{M,\text{geom}}(m)|}{|\mathbf{N}_M(m)|} \geq \omega_{\min} \quad \text{or} \quad \frac{\sum_{m' \in \mathbf{N}_M(m)} \chi(m,m')}{|\mathbf{N}_M(m)|} \leq \bar{\chi}_{\max} \right). \tag{4.9}$$

We are actually interested in a set M^* with maximum cardinality satisfying this condition. The absence of ambiguous matches in M can also be imposed (see below).

4.4.3 Algorithm

The detailed K-VLD pseudo-code is given in Table 4.1. For efficiency reasons, we actually only look for sets M of large cardinality satisfying equation (4.9). Our algorithm starts with $M = \mathcal{M}$ and repeatedly performs the following operations:

(a) remove all $m \in M$ such that $\mathbf{N}_{M,\text{gvld}}(m) < K$

(b) remove all $m \in M$ such that $\dfrac{|\mathbf{N}_{M,\text{geom}}(m)|}{|\mathbf{N}_M(m)|} < \omega_{\min}$ and $\dfrac{\sum_{m' \in \mathbf{N}_M(m)} \chi(m,m')}{|\mathbf{N}_M(m)|} > \bar{\chi}_{\max}$

until no match is removed. Upon termination, which always occurs as $|M|$ strictly decreases (after around 3 iterations, at most 5 in practice), either $M = \emptyset$ or M satisfies condition (4.9). In practice, this yields a large set of matches for M, almost never empty. gVLD-consistency is enforced first because it is the strongest condition. In all our experiments, we use $K = 3$, $\omega_{\min} = 30\%$ and $\bar{\chi}_{\max} = 1.2$.

Dealing with ambiguity

Ambiguous matches, i.e., matches that share a point in one images or the other, correspond to a special kind of mismatches. Treating them as ordinary matches when eliminating outliers does not guarantee an ambiguity-free set of final matches. For this reason, a heuristic elimination is often performed by matching methods to keep at most one match per point, generally by keeping only the one with the best score. But sometimes there is no easy choice, e.g., with ambiguous points on an epipolar line. In that case we can use VLD information to improve disambiguation. For this, we sort the matches in M so that matches m with the highest number of gVLD-consistent neighbors are preferred or, if equal, the highest average of VLD score among these gVLD-consistent neighbors, i.e., $T(m) = \sum_{m' \in \mathbf{N}_{M,\text{gvld}}} \tau(m,m') / |\mathbf{N}_{M,\text{gvld}}(m)|$. More formally, matches are sorted (less likely matches first) according to the order relation: $m \prec_M m'$ iff

$$|\mathbf{N}_{M,\text{gvld}}(m)| < |\mathbf{N}_{M,\text{gvld}}(m')| \text{ or } \big(|\mathbf{N}_{M,\text{gvld}}(m)| = |\mathbf{N}_{M,\text{gvld}}(m')| \text{ and } T(m) > T(m')\big) \quad (4.10)$$

Match disambiguation can then be addressed with the following step added to the algorithm:

(a′) for each $m \in M$ in $\prec_M$-sort order, remove m if $\exists m' \in M \backslash \{m\}$ s.t. m' conflicts with m

If two matches have same values for $|\mathbf{N}_{M,\text{gvld}}|$ and T (ie. $m =_M m'$), we keep both of them. There is a chance to make a decision in later iterations where more false matches are removed. The sorting can also be used in step (a) and (b) of the algorithm for picking $m \in M$, updating M as matches are removed. However, sorting in (a) and (b) does lead to much better results experimentally.

4.4.4 Optimizations and heuristics

Given a match m, we need to count the number of gVLD-consistent neighbors m', which requires computing $\tau(m,m')$. To avoid recomputation, we keep theses values in a cache ($t_{|M|\times|M|}$ in Table 4.1). Besides, to speed up the algorithm, we do not have to enumerate all gVLD-consistent neighbors. It is enough to stop after $N_{\max} \gg K$ neighbors are found, as m is then extremely unlikely to be later removed (e.g., $N_{\max} = 20$ for $K = 3$).

4.5 Evaluation

We experimented with existing matching methods, some of which were also augmented with VLD, and we compared with K-VLD. We evaluated matching accuracy in various imaging conditions. We also tested K-VLD as a prefilter to RANSAC-based calibration.

All extracted features are SIFT keypoints, as implemented in VLFeat [80]. We selected a range of state-of-art methods presenting the rich variety of graph matching approaches. We use the authors' code for probabilistic hypergraph matching (HGM) [88], hypergraph matching via reweighted random walks (RRWHM) [43] and tensor matching (TM) [24], and our own implementation of spectral matching (SM) [44], which computes the same matching results (but with different speed) as integer projected fixed point (IPFP) [45], and game-theoretic matching (GTM) [3]. Besides, VLD was incorporated to SM (2^{nd}-order method) and HGM. For calibration experiments, we used the IPOL implementation of ORSA [58, 57], which is a parameterless, state-of-art RANSAC variant.

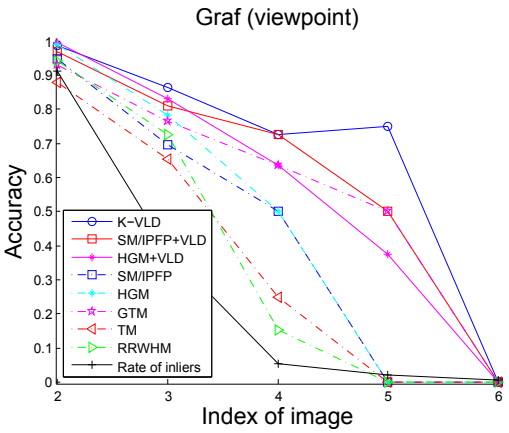
4.5.1 Changing imaging conditions

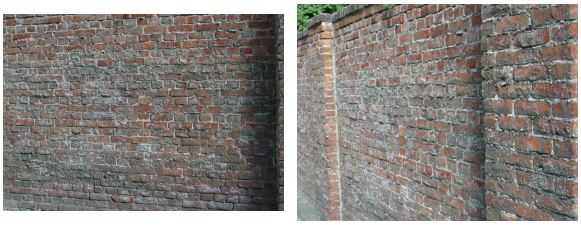
We use Mikolajczyk et al.'s dataset, that evaluates feature detectors and descriptors under different image transformations, including change of viewpoint and illumination, zoom, blur and rotation [53]. It is composed of 8 sequences of 6 images with increasing variation. For each sequence, we successively match image 1 with all other images in the sequence. We extract the best 400 SIFT matches (i.e., with lowest descriptor distance) as candidates for each image pair. 400 features was about the limit that methods TM and RRWHM could handle on a 24 GB computer, running in 200 s; K-VLD runs in 1s, with a performance quasi linear in $|\mathcal{M}|$. For each method, we extract the N best matches according to the method, where N is the number of ground truth inliers. Matches with less than 5-pixel transformation error are considered as inliers, and accuracy is the proportion of correct matches among the N returned matches [43]. This dataset features image transformations that can be described by a single homography. As lines are preserved, VLDs are expected to be relatively stable. In fact, results in Figures 4.7 and 4.8 show that K-VLD often outperforms other methods. Besides, VLD significantly improves existing methods, especially for scenes with viewpoint or scale changes.

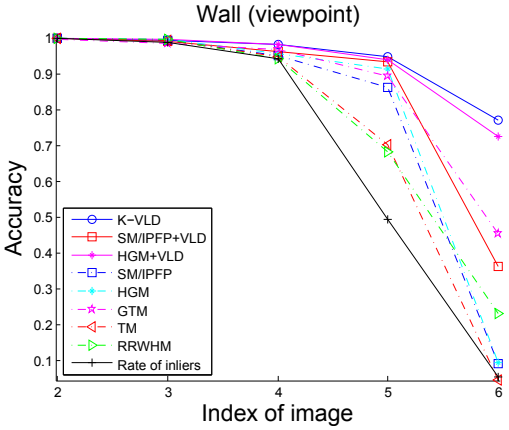
4.5.2 Strong occlusions

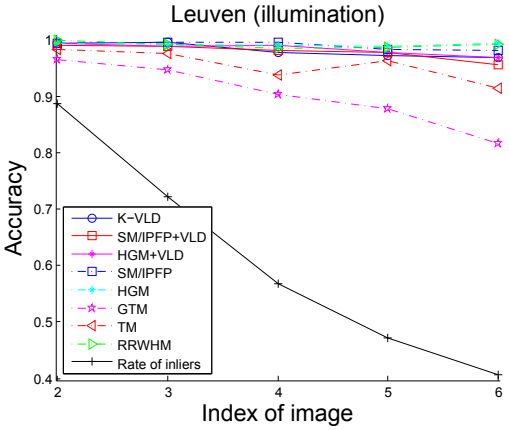
To evaluate the case of occlusions, we use the Dětenice fountain dataset [13], from which we took a sequence of 43 images. The occluding foreground (a statue) creates strong variations in the background. A ground truth calibration is first constructed by selecting 50 correct matches by hand. As above, we extract the best 400 SIFT matches for each pair of successive images. We then measure the actual inlier rate and the accuracy. Results are shown in Figure 4.9, where strong local variations have been smoothed and where the 42 measures have been re-sampled in plotted graphs for readability.

K-VLD creates clusters of consistent matches despite occlusions, outperforming other methods most of the time. VLD improves SM moderately (5–10% more inliers) and HGM only slightly, as it already has an excellent performance.









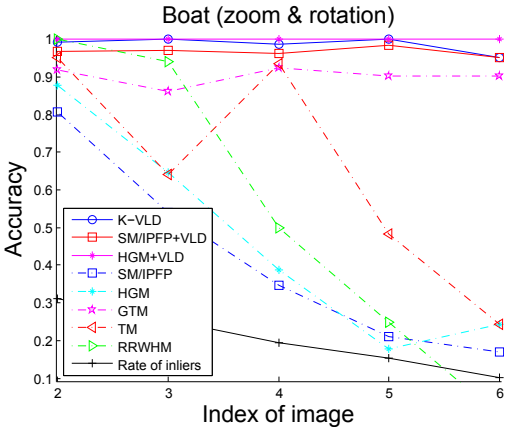
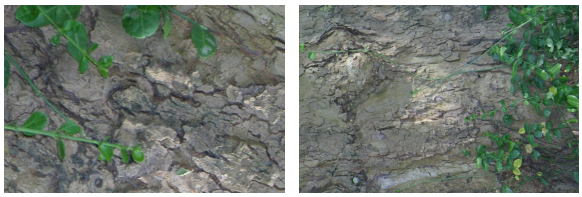
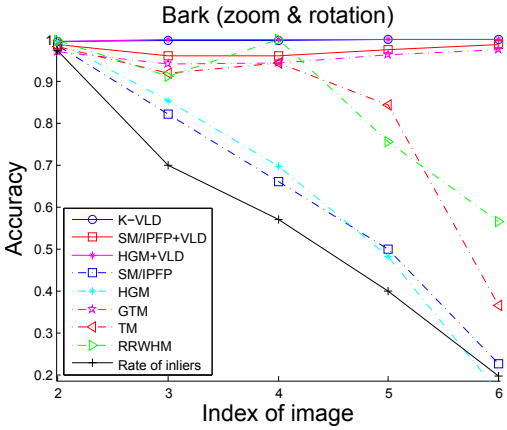
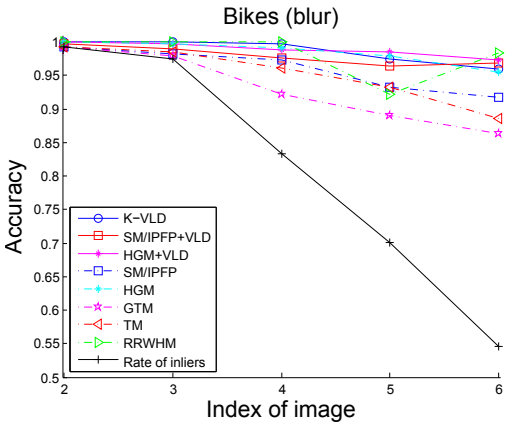
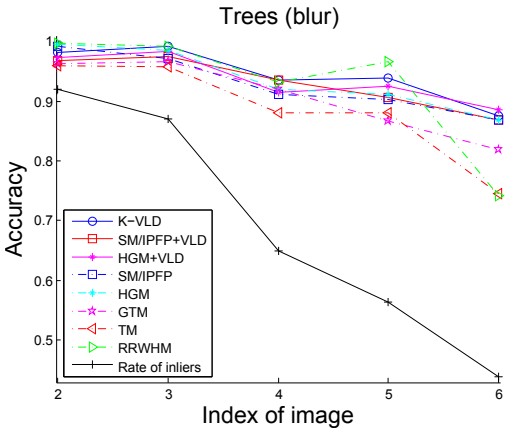


Figure 4.7: Matching accuracy measured on Mikolajczyk et al.'s [53] dataset (part 1)









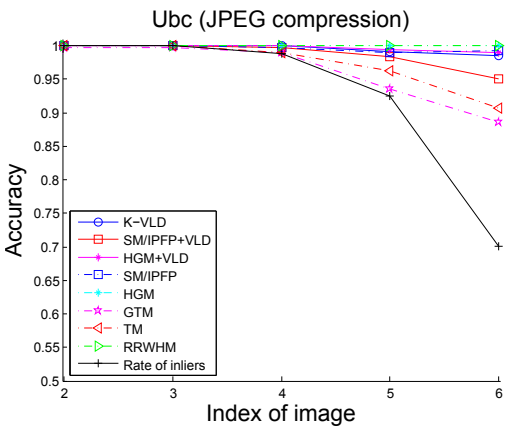


Figure 4.8: Matching accuracy measured on Mikolajczyk et al.'s [53] dataset (part 2)

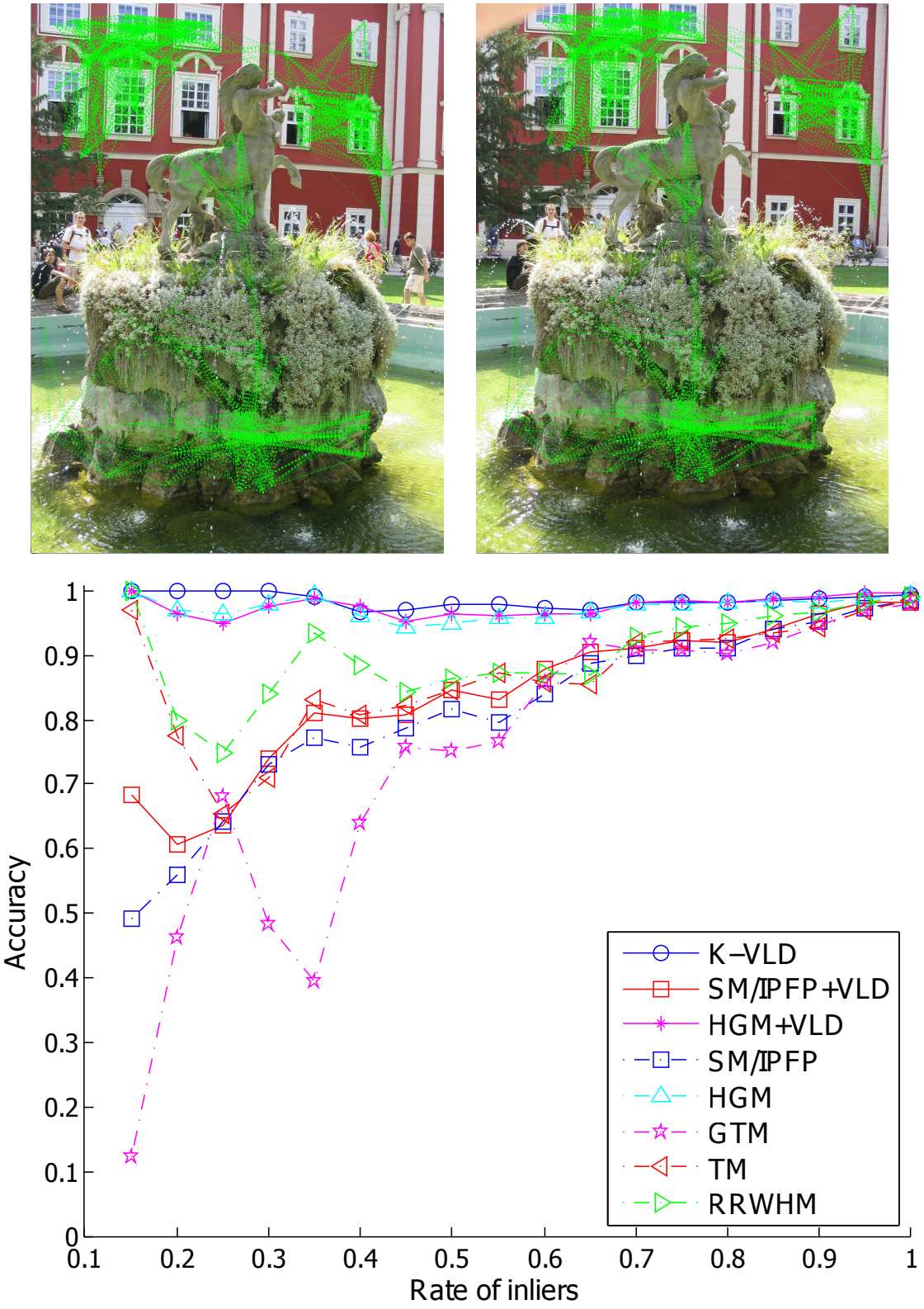


Figure 4.9: Dětenice fountain. Top: K-VLD clusters on one image pair (green lines). Bottom: & average accuracy on all pairs. K-VLD out performs other methods (though the HGM method also achieves good results with or without VLD pairwise constraints in this occlusion test, it is shown in previous experiment that HGM+VLD outperforms HGM under re-scaling and viewpoint changes.)

4.5.3 Ambiguity and RANSAC prefiltering

To evaluate the benefits of K-VLD as a preprocessing filter for RANSAC-based calibration, we use Strecha's Castle-P19 dataset [73]. It is composed of a looping sequence of 19 images of a courtyard and provides ground truth for both internal and external camera calibration. The scene is highly ambiguous due to many repeated windows, which degrades registration, as illustrated on Figure 4.10.

Around 5,000 to 7,000 SIFT points are extracted in each image. We vary the Lowe rejection threshold [49], i.e., the maximum distance ratio of closest to second closest keypoint (often 0.8 in the literature) to generate different potential matches for each pair of consecutive images. A high rejection threshold (≈ 1) allows more ambiguous matches but also increase possible matches. Conversely, a low rejection threshold rejects ambiguous matches but also decreases the number of potential matches. We also test the case of symmetric matching, i.e., points P whose match P' in the second image has P as match in the first image. This yields 300 to 3,000 matches per image pair.

We measure the average rotation error w.r.t. the ground truth rotation over each pair after camera calibration based on K-VLD filtered matches, as well as the standard deviation of this error. As the last image can be compared with the first one, we also measure the accumulated angle error independently of the ground truth by multiplying all the rotation matrices and measuring the angle difference with identity. We compare two methods for estimating the fundamental matrix: using ORSA [57] alone, or prefiltering the matches with K-VLD before ORSA. Results are shown in Figure 4.11. The use of K-VLD as a match prefilter greatly improves stability (deviation) and precision.

Figure 4.10: Matches on pair of images from Strecha's Castle-P19 dataset [73]. Left: inliers by ORSA. Middle: false matches near epipolar lines not eliminated by ORSA but rejected by K-VLD. Right: inliers by K-VLD + ORSA. (Symmetric matches; Lowe criterion threshold = 0.8; for readability, only 1/4 matches shown, thus matches may show or hide in different image pairs.)

4.5.4 Comparison of ASIFT and K-VLD

We compare the results of our K-VLD method with ASIFT [59] on a difficult scene with large view-point change. This is a synthetic scene for which we have an exact ground truth; images are available on IPOL [60]. We use the ASIFT source code from the authors, available on IPOL [87].

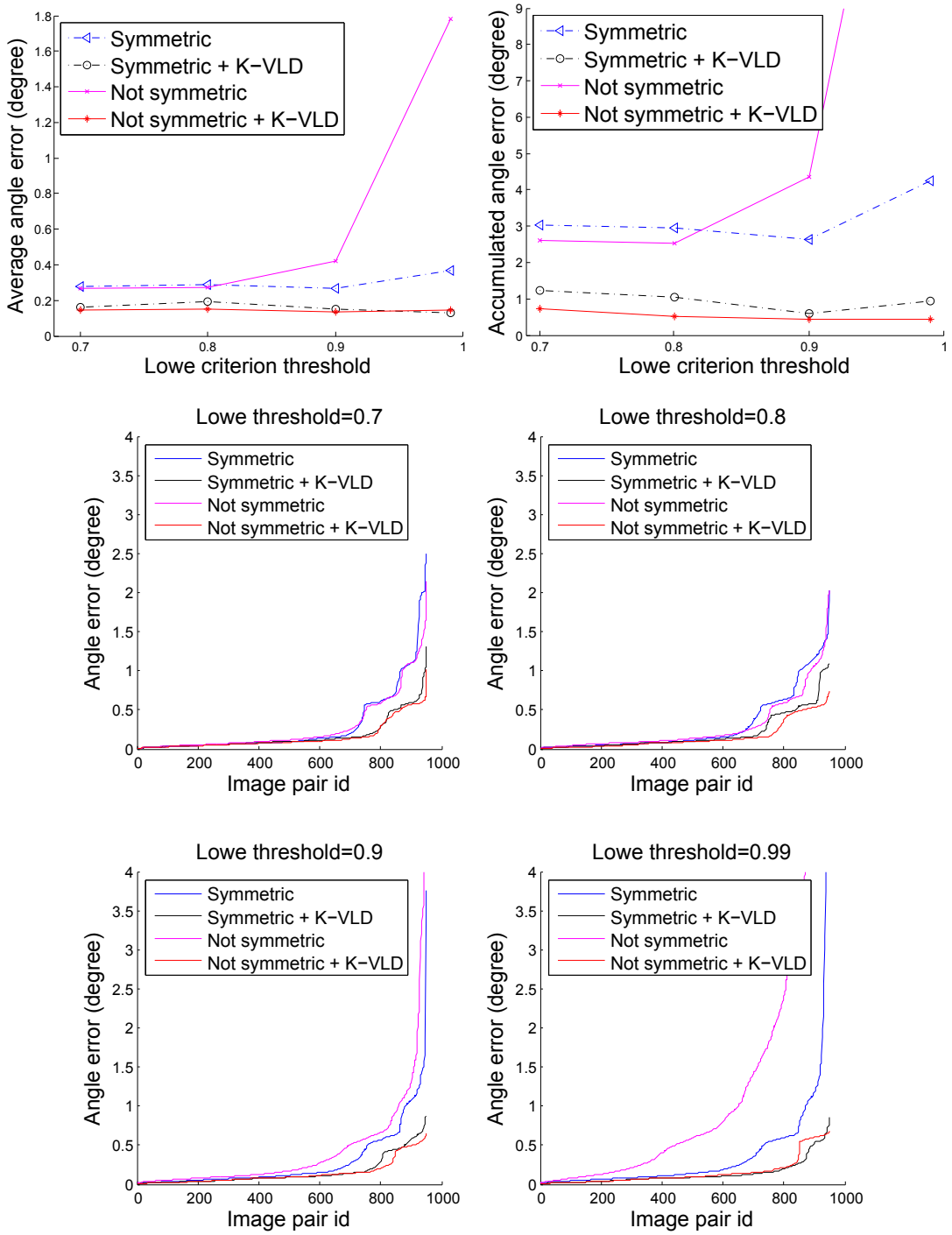


Figure 4.11: Angle error on Strecha's Castle-P19 dataset [73]. Top-Left: average rotation error over 19 image pairs. Top-right: accumulated rotation error after one loop over all 19 image pairs. Bottom: detailed rotation error for each image pair and iteration; results are sorted in increasing error order (50 test results per image pair).

Matching results depend on a chosen threshold for the Lowe criterion [49]. In our test, we use for ASIFT a maximum Lowe score of 0.73, which is the default value of the provided implementation. For SIFT [49], to put us in the worst situation (many ambiguities, hence many mismatches), we use a maximum Lowe score of 0.99.

Figure 4.12 shows the matches found by SIFT (without any symmetric selection criterion). Filtering those matches with K-VLD removes most mismatches. The remaining "mismatches" are rather due to the imprecision of the detections that slightly misplaces corresponding points than to wrong positions.

Figure 4.13 similarly shows matches selected by ASIFT. Due to ambiguities and viewpoint change, there are many false matches. However, using K-VLD as a post-filter to ASIFT removes most mismatches.

Figure 4.14 shows the result of filtering the ASIFT matches with ORSA [57]. As discussed before, the output of ORSA still contains several mismatches near the epipolar lines. But if we filter the ASIFT matches by K-VLD before feeding them into ORSA, most remaining mismatches are removed.

4.6 Parameters

Our VLD descriptor may seem to have many parameters, but many of them actually are directly imported from SIFT, where they have been set to default values after experimenting on a dataset [49]. Our own experiments just taught us that SIFT standard values, e.g., the number of sampling scales per octave, could be weakened in our case for a lighter but still discriminant and robust descriptor.

Most parameters that are specific to VLD are "sanity" bounds, to discard meaningless matches. Their actual value has a low impact on robustness and discrimination. It includes the maximum score for geometric (in)consistency $\chi_{\max}$, the minimum disk radius $r_{\min}$, the maximum distance between descriptors $\tau_{\max}$ and the threshold for high contrast suppression $\kappa_{\max}$. The number of disks U on a virtual line results mainly from a compromise between the computation time and the accuracy in line description.

The main parameter of VLD is the balance β between the consistency of the gradient histograms and the consistency of disk orientations (cf. Section 4.3.5). This parameter, as well as the suggested weight λ when used as a pairwise score in a graph-matching method (cf. Section 3.7.4 and 4.3.6) can be learned as described in [45]. Instead, we looked at the distribution of consistent match pairs vs inconsistent pairs.

More precisely, using images from Mikolajczyk's dataset [53] with ground truth matches, we plotted separately the probablity distribution of consistent match pairs and the probablity distribution of inconsistent match pairs, depending on the gradient consistency term x and the orientation consistency term y:

$$x = \sum_{u=1}^{U} \sum_{v=1}^{V} |h_{u,v} - h'_{u,v}|$$

$$y = \sum_{u=1}^{U} \left(\frac{\gamma_u + \gamma'_u}{2} \cdot \frac{\min(|w^*_u - w'^*_u|, W - |w^*_u - w'^*_u|)}{W/2} \right)$$

These two terms x and y are balanced using parameter β in the definition of the VLD distance τ (see Equation 4.5). Both distributions happen to consist of a single mode, whose support is clearly separated from the support of the other distribution. Figure 4.15 illustrates the superposition of both distributions on a single graph, with an indication of the corresponding distribution on each peak. The choice of a value for β corresponds

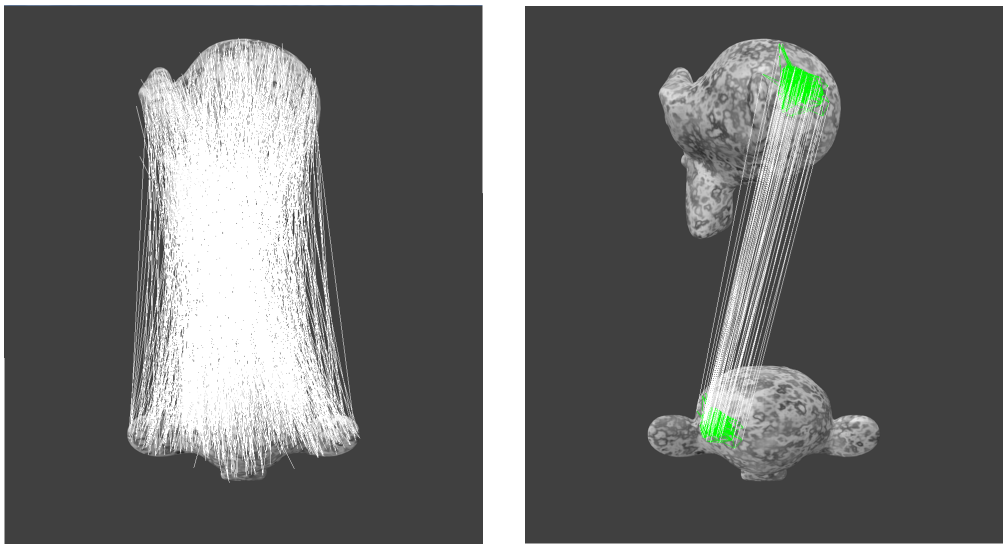


Figure 4.12: Left: SIFT matches (with Lowe score = 0.99, no symmetry).
Right: K-VLD filtering of these matches (+ K-VLD clusters).

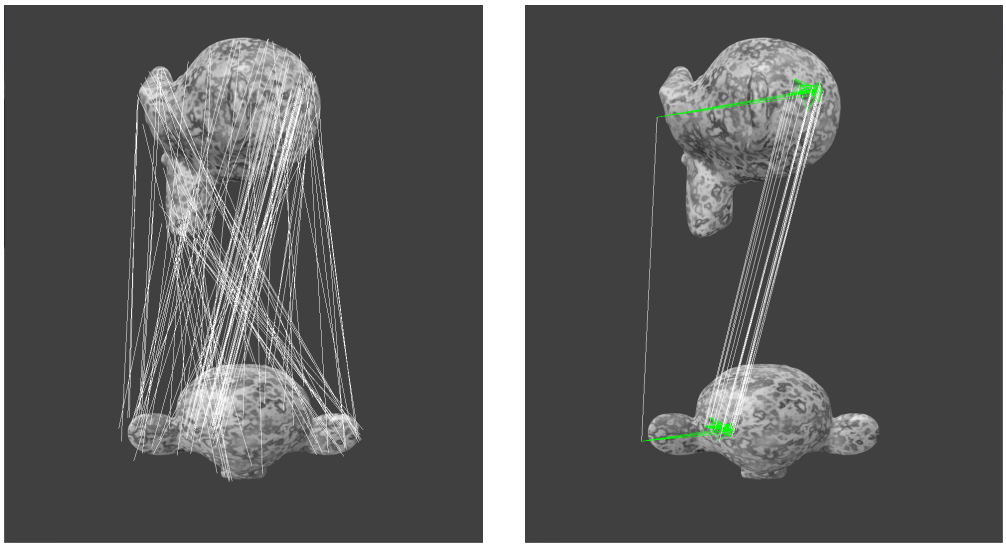


Figure 4.13: Left: ASIFT matches (with default Lowe score = 0.73).
Right: K-VLD filtering of these matches (+ K-VLD clusters).

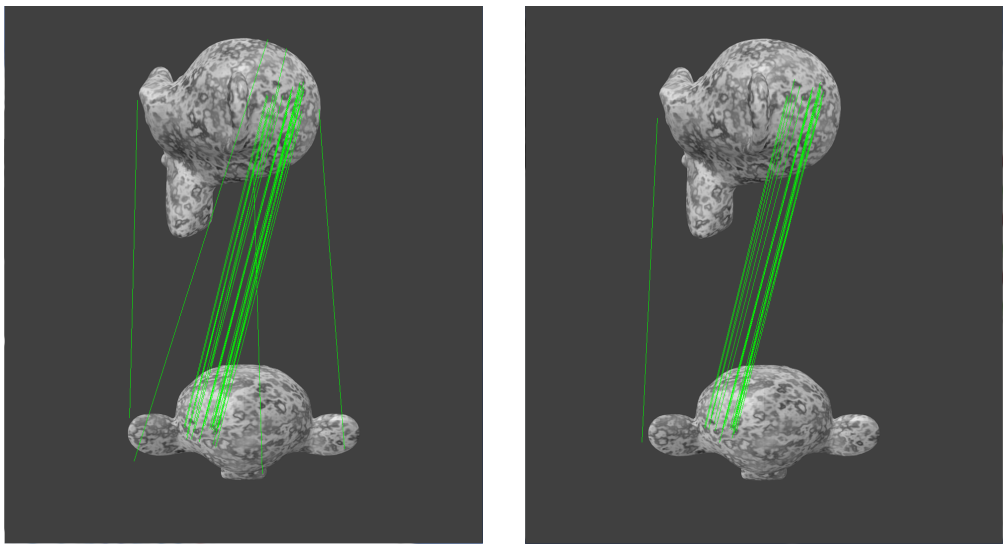


Figure 4.14: Left: ASIFT matches filtered by ORSA.
Right: ASIFT matches filtered by K-VLD, then by ORSA.

to the choice of a safe separation line between the correct and incorrect match pairs, i.e., a line corresponding to $\beta x + (1-\beta)y = \tau_{max}$.

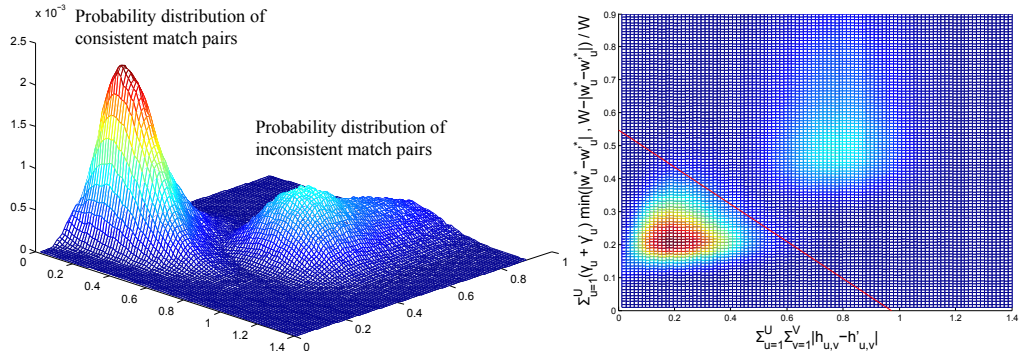


Figure 4.15: Left: side view of the superposed probability distributions. Right: top-down view of the superposed probability distributions. The red line represent $\beta x + (1-\beta)y = \tau_{max}$, with $\beta = 0.36$ and $\tau_{max} = 0.35$.

An important parameter for the K-VLD matching method is the minimum number K of neighboring matches that are gVLD-consistent. The choice of $K = 3$ in our experiments results from a compromise between robustness (keeping enough inliers for later use) and discrimination (removing a significant number of mismatches). It also has an impact on computation time.

As VLD, K-VLD also makes use consistency thresholds and "sanity" bounds, i.e., the minimum inlier rate ρ_{min}, which can be very low (e.g., 3%), the neighbor exclusion radius B_{min}, the minimum proportion of geometry-consistent neighbors ω_{min} and the maximum average score of geometric consistency for neighbors $\bar{\chi}_{max}$. When choosing values for $\bar{\chi}_{max}$ and ω_{min}, we want to keep the resulting angle error as small as possible while producing a number of inliers is as large as possible. Figure 4.16 illustrates how they are chosen, setting all parameters but one to plausible values for good performance, and letting the remaining parameter under study vary. The test is based on image pairs of Strecha et al.'s Castle-P19 dataset over 20 iterations with the same measuring as used in Section 4.5.3.

4.7 K-VLD's contribution for N-view SfM

The K-VLD method cleans up wrong matches even in the case of match ambiguity, e.e., when there are repeated objects in a scene. It increases the match quality for later use, such as N-view camera calibration.

To illustrate it, we consider both a well-known incremental SfM method (Bundler [72]) and a state-of-art global SfM method (Moulon et al. [62]). Tables 4.2 and 4.3 presents an evaluation of the accuracy of these calibration methods on Strecha et al.'s datasets [73]. The rotation error measures the average angular difference of the cameras from the true rotation (in degrees). The translation error measures the average distance of the cameras to their true positions (in mm).

Using matches filtered first by K-VLD results in a more accurate camera calibration (rotation and position) for scenes with repetitive patterns (Castle-P19, Castle-P30 and Entry-P10). For scenes with little or no repetition such as Herz-Jesu-P8 and Herz-Jesu-P25, the incremental method using K-VLD has worse results, whereas the more accurate global method using K-VLD performs slightly better.

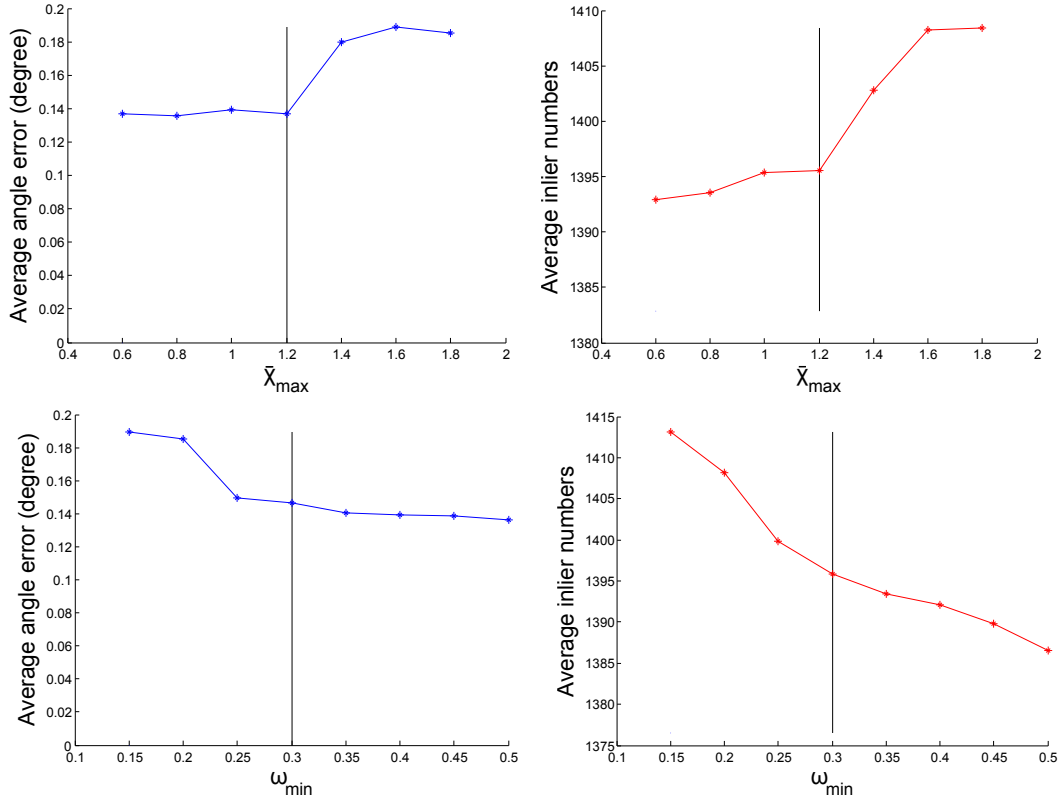


Figure 4.16: Performance evaluation for parameters $\bar{\chi}_{max}$ and ω_{min}. The default parameter value is in black, the average angle evaluation is in blue and the average number of inliers is in red.

The numbers in Tables 4.2 and 4.3 differs from ones in [62] because we use a peak saliency threshold of 0.04 and Lowe ratio threshold of 0.8, which is the default SIFT setting, while in [62] the peak threshold is equal to 0.02, to generate a larger number of features. What matters most here is the relative improvement brought by K-VLD to both incremental and global methods.

4.8 Running time

K-VLD is semi-local in that it considers only a limited number of match neighbors to evaluate consistency, before deciding whether a match is an inlier or an outlier. As a result, its running time is more or less linear in the number of matches.

This is observed experimentally on the images pairs used for the N-view calibration of the scenes in Strecha et al.'s dataset [73], as shown in Figure 4.17. For a given image pair and an input set of potential matches, the most computationally expensive part of K-VLD is the photometric consistency evaluation, as opposed, e.g., to geometry consistency checking. The dominant requirement in memory is the score table $t_{|M|\times|M|}$. Still, our algorithm runs smoothly even on image pairs with 30,000 matches on a computer with 24GB of RAM.

Unit: degree	Bundler	K-VLD + Bundler	Global	K-VLD + Global
Castle-P19	0.158	0.054	0.038	0.023
Castle-P30	0.094	0.039	0.027	0.026
Entry-P10	0.208	0.062	0.035	0.035
Fountain-P11	0.173	0.146	0.019	0.019
Herz-Jesu-P8	0.148	0.164	0.058	0.055
Herz-Jesu-P25	0.036	0.088	0.032	0.032

Table 4.2: Average angle error.

Unit: mm	Bundler	K-VLD + Bundler	Global	K-VLD + Global
Castle-P19	210.5	74.7	28.0	24.6
Castle-P30	73.1	49.3	23.4	22.9
Entry-P10	61.7	31.9	6.7	5.9
Fountain-P11	13.1	9.7	2.6	2.6
Herz-Jesu-P8	20.6	26.0	3.8	3.7
Herz-Jesu-P25	11.4	16.7	5.7	5.6

Table 4.3: Average camera position error with respect to ground truth.

4.9 Limitations of VLD and K-VLD methods

4.9.1 Limitation w.r.t. detection inaccuracies

The design of our virtual line descriptor is inspired by the SIFT descriptor [49]. By consequence, it inherits more or less the same advantages and drawbacks as SIFT. Typically, the SIFT descriptor is tolerant to a few pixel misplacement of the gradients around the feature, which is the same for VLD. By consequence, VLD is not sensitive to detection misplacements (within a few pixel range) and may keep matches that are inaccurate. Figure 4.18 shows a misplaced feature that has been kept because it is consistent with its neighbors.

Consequently, VLD is robust against false matches but is not efficient against slightly misplaced matches. The matching accuracy still mainly relies on feature detection.

4.9.2 Limitation w.r.t. repetitive patterns

Repetitive patterns have always been a problem for feature matching. One approach is to exclude repetitive features. Lowe criterion can eliminate ambiguous matches to a certain extent at the cost of losing many matches. However, some images such as urban scenes contain mostly repetitive structures, like windows and balconies, and excluding many of them is a significant loss. We have seen an example of the limitation of the Lowe criterion and the impact of repetitive features in Section 4.5.3.

By design, the K-VLD method requires every match to be consistent with its neighboring matches (geometric and photometric consistency), which we called "semi-local" validation. This validation strategy works very well under strong occlusion and ambiguities. However, there are two situations for which repetitive patterns can still persist after K-VLD filtering. We have set the Lowe criterion score to 1.0 with no symmetry verification to amplify these situations in Figure 4.19.

The first situation happens for a repetitive pattern when the true correspondences of a group of features are occluded (or missing). Lacking the correct matches, the

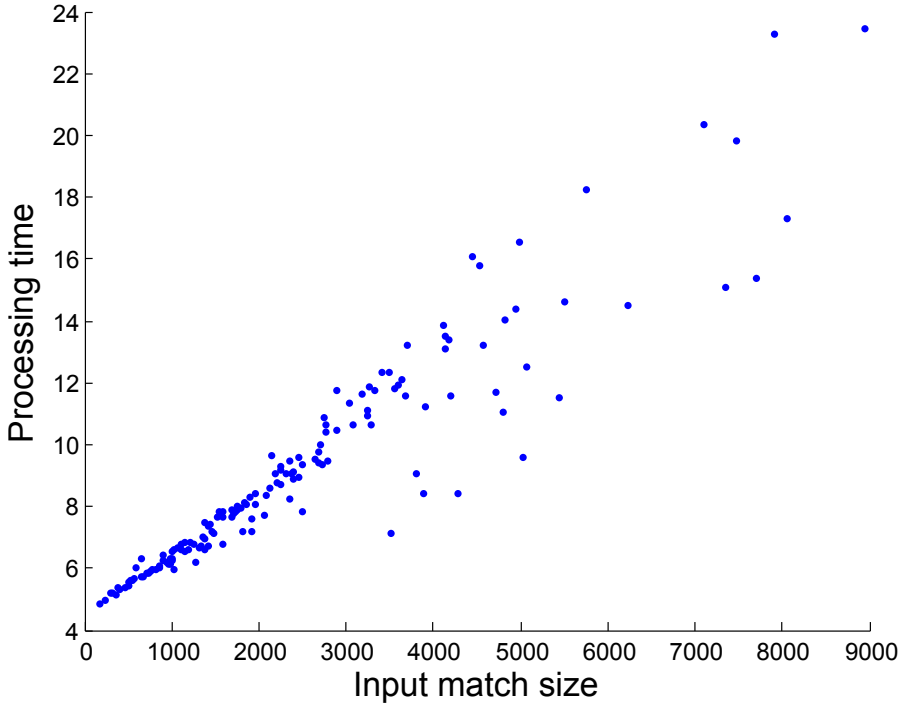


Figure 4.17: K-VLD running time in seconds for contiguous image pairs on Strecha et al.'s Castle-P19 dataset [73], depending on the number of match candidates as input. Each image pair is tested with Lowe ratio threshold equals to 0.7, 0.8, 0.9 and 0.99, with and without symmetric matching criterion.

nearest neighbor principle for matching can assign a set of wrong matches to the occluded group, which could also be self-consistent. It is hard to remedy this situation by strengthening K-VLD parameters because a too severe geometric constraint would then lose matches in the case of large view-point changes.

The second situation occurs for a repetitive pattern when a subset of features are mismatched with a set of neighboring repeated entities. In this case, the geometric and photometric consistency might be achieved although the matches are wrong. The final geometric filter (step b in Table 4.1) partially avoids this situation, however this case can happen when the mismatched subset contains a large-enough number of features. Fortunately, one clue is observed when this situation happens: the output matches is divided into two gVLD-consistent clusters, and features in one image of these two clusters overlap. A solution to this problem could thus be to add a cluster overlapping detection, and either keep the most important cluster or grow the existing clusters by re-assigning matches for mismatched features. This is future work.

4.10 Conclusion

For 2^{nd}-order graph matching, distinctive pairwise constraints are crucial, just as distinctiveness is crucial for ordinary, 1^{st}-order feature matching. As our experiments show, our virtual line descriptor (VLD) provides such distinctiveness, offering a better accuracy to 2^{nd}-order graph matchers. Besides, our K-VLD matching method is scalable (in time and space) to a large number of points, contrary to high-order graph matchers (cf. Section 3.7.4). Compared to other graph matching methods, our K-VLD matcher also provides among the best accuracies, especially when the inlier rate drops,

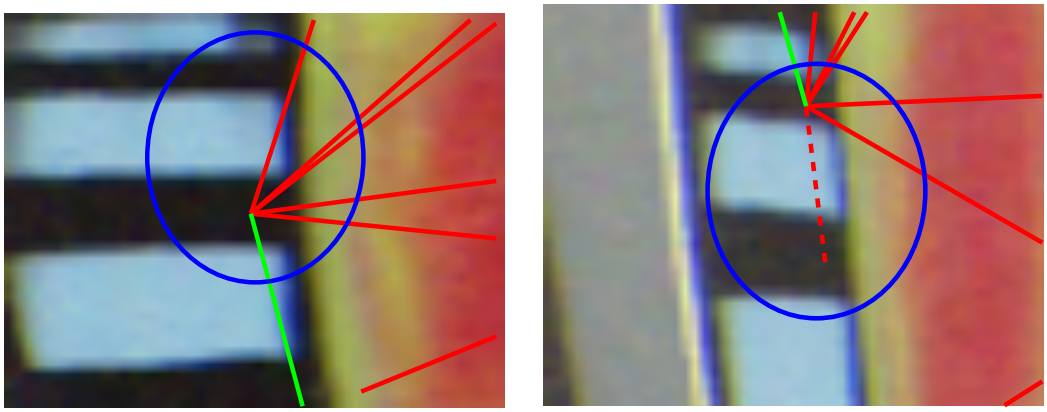


Figure 4.18: Illstration of VLD limitation. We consider the matching result of two images from the Graf scene of Mikolajczyk et al.'s dataset [53]. The dashed line on the right marks the distance between the projection of the detected feature in the left image and the matching detected feature in the right image. Although the two points are not the same, K-VLD wrongly validates the match as inlier because it has enough gVLD-consistent neighbors.

even down to a few percents (less than 5 or 10%) and despite strong occlusions, or strong viewpoint or scale changes. Moreover, used as a preprocessor to RANSAC, K-VLD eliminates most mismatches, including near epipolar lines and despite possible ambiguities, which greatly improves calibration precision.

All these results were achieved with unchanged parameter values after moderate experimentation to set the, but a more systematic study *à la* SIFT [49] on a larger benchmark would be valuable.

Figure 4.19: Illustration of K-VLD limitation. Outliers are shown in yellow, and validated VLD are in red. Left: First case, where the correct matches of a repetitive group of features are missing. The nearest neighbor principle assigned another set of consistent entities to this group, which is very hard to remove by K-VLD Right: second case, where a subset of features of a repetitive pattern is mismatched, the mismatched cluster of matches overlaps with another cluster of matches.

Chapter 5

Various applications of the K-VLD method

5.1 Introduction

The K-VLD method is applied to various fields thanks to some of its specific properties. In this section, we present several applications using either virtual line descriptor or the K-VLD method.

Contents

5.1 Introduction . . . 75
5.2 Image Color Blending . . . 76
5.2.1 Introduction . . . 76
5.2.2 Related work . . . 76
5.2.3 K-VLD dense common region detection . . . 76
5.2.4 Application to the color coherence . . . 77
5.2.5 Conclusion . . . 78
5.3 Line segment matching . . . 80
5.3.1 Introduction . . . 80
5.3.2 Related work . . . 80
5.3.3 Our contribution . . . 81
5.3.4 Line segment detection and description . . . 81
5.3.5 Initial candidate matches . . . 81
5.3.6 Pairwise geometric constraint . . . 82
5.3.7 Semi-local photometric constraint . . . 83
5.3.8 K-VLD method for line segments . . . 84
5.3.9 Experiments . . . 84
5.3.10 Computation time . . . 89
5.3.11 Conclusion . . . 89
5.4 Deformable object matching . . . 91
5.4.1 Introduction . . . 91
5.4.2 Experiments . . . 91
5.5 Urban localization from Google street views . . . 97
5.5.1 Introduction . . . 97
5.5.2 K-VLD contribution in air-ground image matching . . . 97
5.5.3 Experiment . . . 99

5.2 Image Color Blending

5.2.1 Introduction

Digital cameras enable various multiple-image applications, such as 3D reconstruction, image panoramas and image blending, etc. All these applications need to merge information from several photos. However, it is often the case that objects in the scene appear with slightly different colors in different photos. Merging images becomes non trivial.

When taking a photo, light first goes through a series of lenses, before it hits the optical sensors at the focal plane; many aspects can affect the produced image.

In terms of changing environment, photos may be taken at different times of day, with different weather conditions.

In terms of instruments, photos may be taken by different cameras, or even by the same camera, but with different setups. Camera lenses have different apertures, controlling the amount of coming light per time unit. The shutter time could be different as well. The optical sensors from different manufacturers do not have the same response to the same spectral input.

In terms of software, digital cameras are equipped with post-processing algorithms, including white-balancing and tone-mapping. Additionally, many independent software such as photoshop can easily modify the rendering of images.

Consequently, when it comes to assemble several images in various applications such as panorama making and 3D scene reconstruction, one has to handle the problem of inconsistent colors across different images, as in Figure 5.1.

5.2.2 Related work

According to Xu and Mulligan [86], much work has been done to minimize the inconsistency in color for two or multiple images. The state-of-the-art algorithms consist of two parts: common region identification and color difference minimization. We concentrate on the problem of common regions identification as K-VLD can contribute to address this issue. For images taken from different view points, it is reasonable to only consider the common area between images instead of the whole pictures. Traditional sparse feature matching strategies provide only local spot-like region correspondences and do not identify common areas. In order to find a sufficiently large common area between two images, dense matching methods have been proposed as SIFT-flow [47], Generalized PatchMatch (GPM) and the Non-rigid Dense Correspondence (NRDC) [31], see Figure 5.2. However, the dense matching methods are computationally expensive, and are thus used mainly on miniature images. They are not appropriate for larger images nor multiple images. For multiple images, Hacohen [32] proposes several techniques to reduce the pairwise image comparison.

5.2.3 K-VLD dense common region detection

We now describe an efficient way to extract dense common areas between images using the K-VLD matching method. To find the common area between two images, we do not need to know for each pixel its exact position in the other image, but only a binary indicator, meaning whether it is in the common region. As explained in Chapter 4, K-VLD simutaneously extracts a consistent subset of matches as well as virtual lines between matches, that are similar, i.e., that have similar VLDs. For a valid VLD between

Figure 5.1: Illustration of inconsistent color in panorama reconstruction by Brown and Lowe [11] and 3D scene reconstruction by Allene et al. [5]. Images borrowed from [11] and [5].

correct matches, the regions in the two images that lays in the VLD pathes are likely to represent the same area as they are photometrically consistent, see Figure 5.3.

Formally, we consider correct matches $m_{i,i'}=(\mathbf{x}_i,\mathbf{x}'_{i'})$ and $m_{j,j'}=(\mathbf{x}_j,\mathbf{x}'_{j'})$ such that $l_{i,j}=(\mathbf{x}_i,\mathbf{x}_j)$ in I and $l'_{i',j'}=(\mathbf{x}'_{i'},\mathbf{x}'_{j'})$ in I' are gVLD-consistent. We suppose $l_{i,j}$ is of length d and $l'_{i',j'}$ is of length d', we consider the rectangle of size $d\times\frac{2d}{U+1}$ in I and the rectangle $d'\times\frac{2d'}{U+1}$ in I', i.e., the regions used to construct the VLDs $l_{i,j}$ and $l'_{i',j'}$, where U is number of disks on a VLD. Given a pair of images, the dense common region in each image is the union of all rectangles on valid VLDs. Figure 5.5 shows the detection result of images from Strecha et al.'s dataset [73].

Note that K-VLD may not detect all common regions since its detection is based on feature matches. However, it detects "large enough" dense regions with a complexity not greater than sparse feature point matching.

5.2.4 Application to the color coherence

This common region detection algorithm has been used by Moulon et al. [63] as the basis to enforce color consistency in an SfM context. It is illustrated in Figure 5.4.

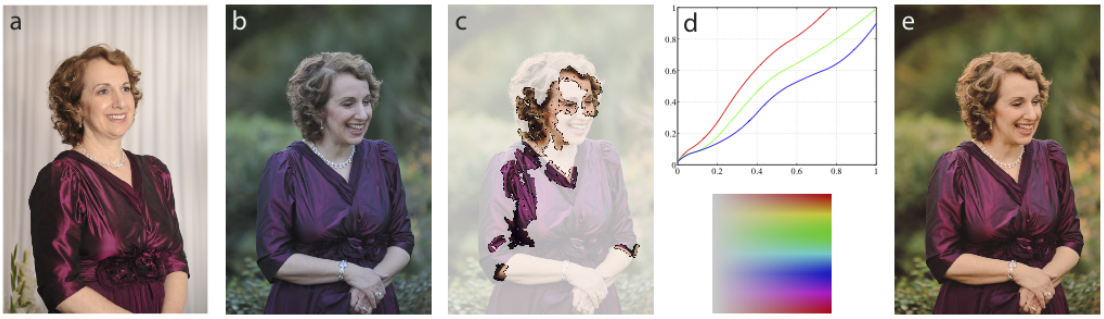


Figure 5.2: Illustration of selected common area by NRDC. Image borrowed from [31].

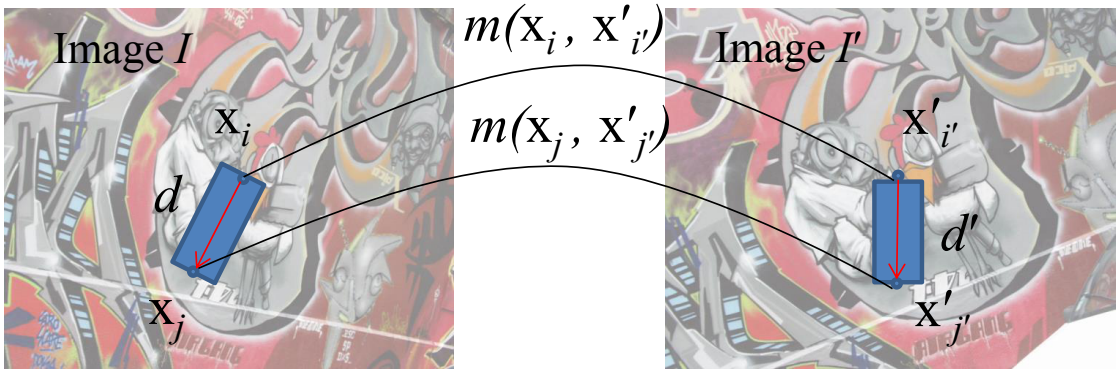


Figure 5.3: Illustration of common region selected by a valid VLD. If neighboring matches $m_{i,i'}=(\mathbf{x}_i,\mathbf{x}'_{i'})$ and $m_{j,j'}=(\mathbf{x}_j,\mathbf{x}'_{j'})$ are correct, and if $l_{i,j}=(\mathbf{x}_i,\mathbf{x}_j)$ in I and $l'_{i',j'}=(\mathbf{x}'_{i'},\mathbf{x}'_{j'})$ in I' are gVLD-consistent, then the blue regions in two images can be considered as common regions.

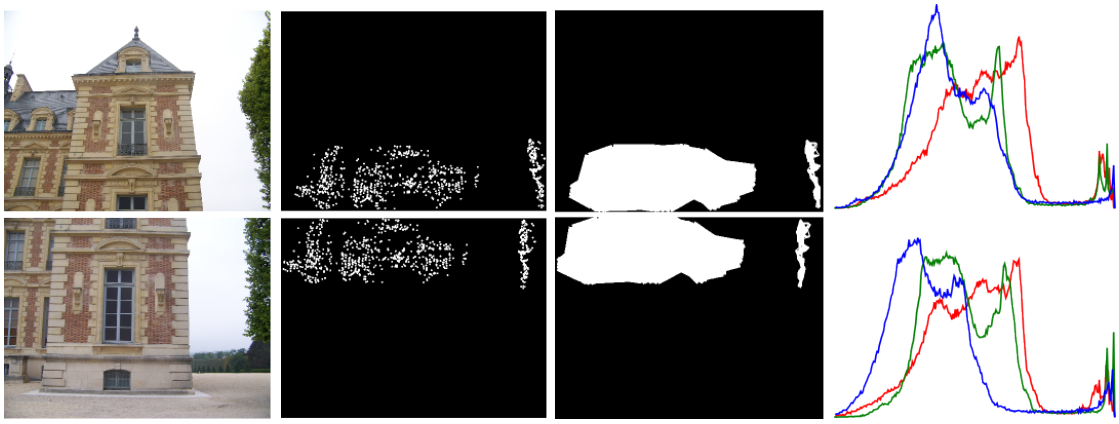


Figure 5.4: Common region detection for color blending (courtesy of P. Moulon [63]). From left to right: input images, filtered SIFT correspondence, common region detection by the K-VLD method, histograms of the 3 color channels.

5.2.5 Conclusion

The K-VLD method offers the possibility to detect dense common region for large images and large datasets at the cost of sparse (but semi-local) feature matching, which makes dense common region detection and color blending practical for realistic datasets.

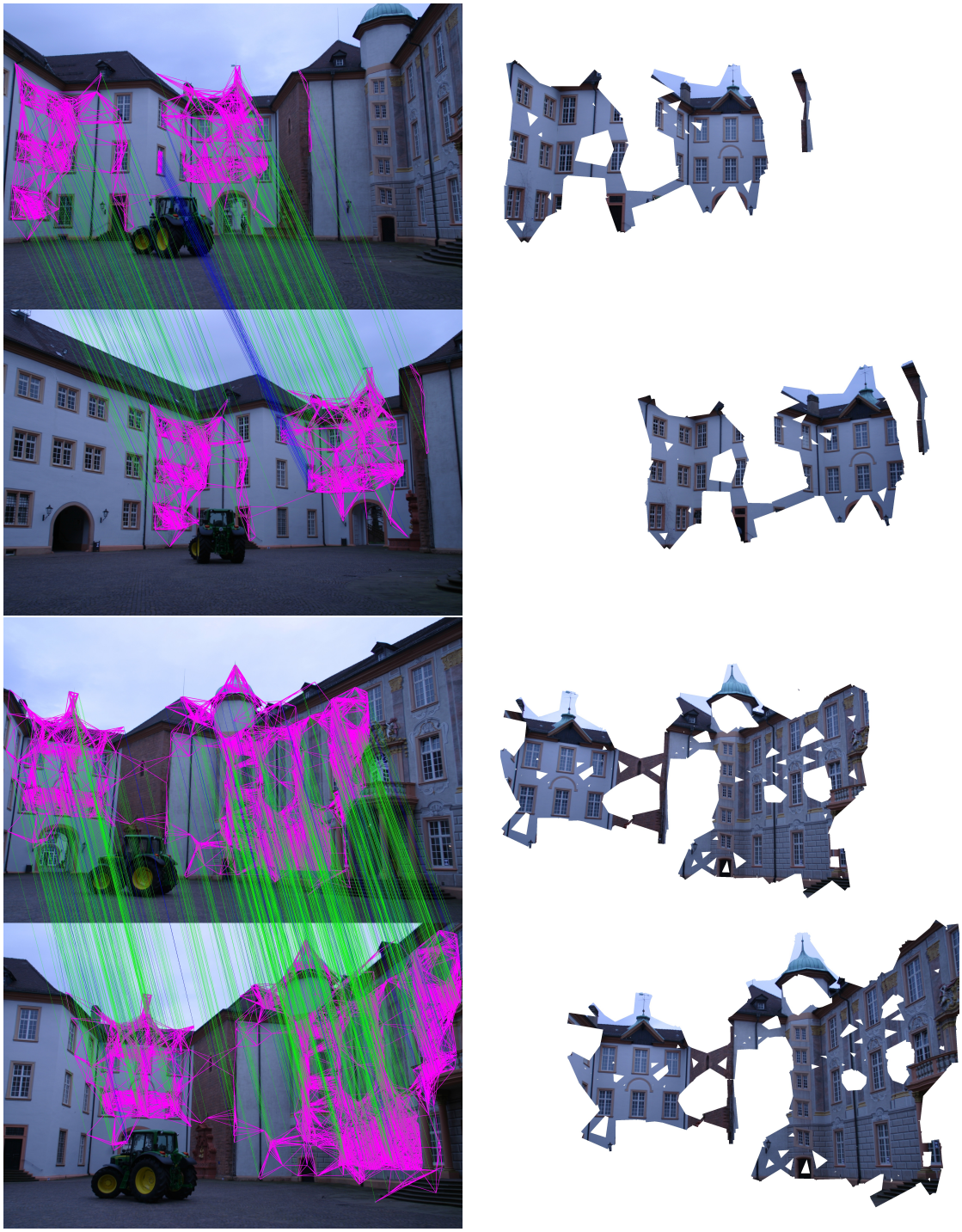


Figure 5.5: Demonstration of common region detection using the K-VLD method. Left: the K-VLD filtered matches, with ORSA post processing. Matches kept by ORSA are in green, others in blue. Valid VLDs are in magenta. Right: Reconstructed common region by all valid VLDs.

5.3 Line segment matching

5.3.1 Introduction

The following is a joint work with Yohann Salaun. It is an application of K-VLD method for line segment matching. Line segment matching is a good complement to feature point matching, especially for indoor scenes where textures are poor, yielding little point saliency, and transformations are strong, from different points of view. Line segments almost always exist in indoor scenes and can be robustly detected even under rough view-point change. However, one of the difficulty of line segments is their description and matching. Segments mostly lay on the border of surfaces, where occlusion is common, making descriptors unreliable for matching. To solve this problem, we propose to apply only a weak selection according to segment descriptors and then to use semi-local information between lines to increase the number of matches as well as to eliminate mismatches.

5.3.2 Related work

Many methods have been proposed to improve line segment matching at different stages of the matching process: feature description, geometric constraints and features completion etc.

Descriptor

Wang et al. [83] have generalized the SIFT method for segment matching purpose. Their descriptor, MSLD (for Mean-Standard deviation Line Descriptor), divides the area around the segments and sums up information with mean and standard deviation. Lines are then matched the same way as SIFT with a similar Lowe criterion. Zhang et al. [90] used a similar but simpler descriptor in which they added scale and geometric components. It gives good results with a moderate baseline motion but it is computation heavy with larger pictures. These methods give better results for textured scenes, but also generate many mismatches in more general scenes. Besides their performance is affected by occluded parts along segments.

Matching with geometric constraints

Since descriptors for segment are not fully reliable, pairwise or higher-order geometric constraints have also been used in the segment matching process. Schmid and Zisserman [71] or Hofer et al. [35] first estimate the epipolar geometry with point correspondences. Then the search area for segment correspondences is limited by epipolar geometry on segment extremities. Fan et al. [25] use local affine transformations between lines and neighboring points to compute the matches. These methods get good results in textured scenes but heavily depend on point correspondences. Their performance is thus limited in low-texture scenes. Wang et al. [82] introduced a pairwise geometric constraint termed as “line signature”. The line signature describes segment position relation using angles and segment length ratios. Although this performs well on the authors’ test images, in our experiments on indoor scenes images with significant baseline and viewpoint change, the performance is limited.

RANSAC for lines

Another drawback of line segments is that their extremities are not accurate for model fitting. Only line constraint are thus used to build equations to be solved by RANSAC. This requires segment matching among a minimum of three images (tracks of length 3) and needs 13 random selections of tracks (cf. [34, §15.1]). This is dramatic for two reasons. First, it is already difficult to have right segment correspondences between two images, a track of length 3 requires 3 correct pairings of segments, the number of correct tracks thus drops rapidly. Second, for point features, RANSAC requires 7 correct matches, to compare with 13 correct matches among contaminated tracks, which requires many more iterations. For instance, using a possible recommended iteration number formula as $K = \frac{\log(1-P)}{\log(1-\rho^n)} + \frac{\sqrt{1-\rho^n}}{\rho^n}$ and supposing we have a dataset with inlier rate $\rho = 0.5$, the recommended iteration number for 7-sample RANSAC equals 216, but for 13-samples RANSAC, it equals 13869. By consequence, RANSAC-based methods perform poorly for line segments and are impractical for difficult scenes.

5.3.3 Our contribution

In this section, we extend the K-VLD method to adapt it to line segment matching, in order to take both pairwise geometric consistency and semi-local photometric constraints into account. Given the unreliability of initial matches, and the capacity of K-VLD to treat a large number of candidate matches, our algorithm can take a larger amount of potential matches for input, in which more correct matches are included (but also much more mismatches). Our algorithm outperforms state-of-the-art methods both in number of matches and correctness.

5.3.4 Line segment detection and description

Our K-VLD method for segment does not require a specific line segment detection method as long as the detection provides line orientation and scale. In terms of description, we implemented the method developed by Zhang [90], an extension of MSLD (SIFT-like) descriptor [83], to gain some robustness against the line fragmentation issue. The main idea of this descriptor is to compute the gradient information for both sides of the segment. The neighboring area around a segment is divided into stripes whose gradient information is condensed into the mean and the standard deviation.

5.3.5 Initial candidate matches

In the following, we consider a pair of image I, I', where we have n line segments detected in image I and n' line segments in image I'.

Strategies used for feature point matching are often transposed to generate initial candidate segment matches, such as considering the first neighbor in the other image, with optionally the symmetric matching criterion and Lowe ratio criterion. However, descriptors for segments are not as reliable as for point features. A radical alternative method would be to consider the all-to-all matches as input for K-VLD. This gets around the step of descriptor matching but it is computationally too expensive. Our heuristics is a trade-off that considers a large enough number of matches that have a good chance to include many correct ones.

For this, we first generate the all-to-all score matrix by comparing descriptors, then we set up a heuristic number N of initial matches, which is much less than that of the all-to-all strategy but much higher than what the nearest neighbor would produces. N satisfies:

$$\max(n,n') \ll N \ll n \times n' \tag{5.1}$$

Our initial matches consist of N matches of best scores.

Besides, we also tried to use other graph matchers to pre-select matches before K-VLD, but results are more time consuming with no observable improvements.

5.3.6 Pairwise geometric constraint

Compared to a point feature, a line segment contains more reliable information in terms of orientation and re-scaling about its transformation between image I and I', which allows us to make a new measurement of geometric consistency error.

We note detected oriented segments as $L_i = \overrightarrow{A_i Z_i}$, $i \in \{1,\dots,n\}$ in image I and $L'_{i'} = \overrightarrow{A'_{i'} Z'_{i'}}$, $i' \in \{1,\dots,n'\}$ in image I'. The line orientation is given by the orientation of gradients on the line, as discovered by the line segment detector.

Heuristically, the scale of the line is estimated by its length. It is wrong and it may have a negative impact if the line is oversegmented or partially occluded in one image. We actually only consider here one-to-one correspondences between image segments, as opposed to many-to-many or partial matches. However, in practice, the impact of possible scale errors is moderate and although we may miss a few actual correspondences, we still find many matches (see Section 5.3.9).

For a segment L_i in I which has a correspondence $L'_{i'}$ in I', we note the transformation from L_i to $L'_{i'}$ as $P_{(i,i')}$, and $P'_{(i',i)}$ its reverse transformation:

$$\begin{aligned} L'_{i'} &= P_{(i,i')}(L_i) \\ L_i &= P'_{(i',i)}(L'_{i'}) \\ \forall j, L_j &= P'_{(i',i)} P_{(i,i')}(L_j) \end{aligned} \tag{5.2}$$

Note that $P_{(i,i')}$ and $P'_{(i',i)}$ consist of a translation, a rotation and a re-scaling. We also define the line segment distance

$$d_{seg}(L_i, L_j) = max(||A_i - A_j||, ||Z_i - Z_j||) \tag{5.3}$$

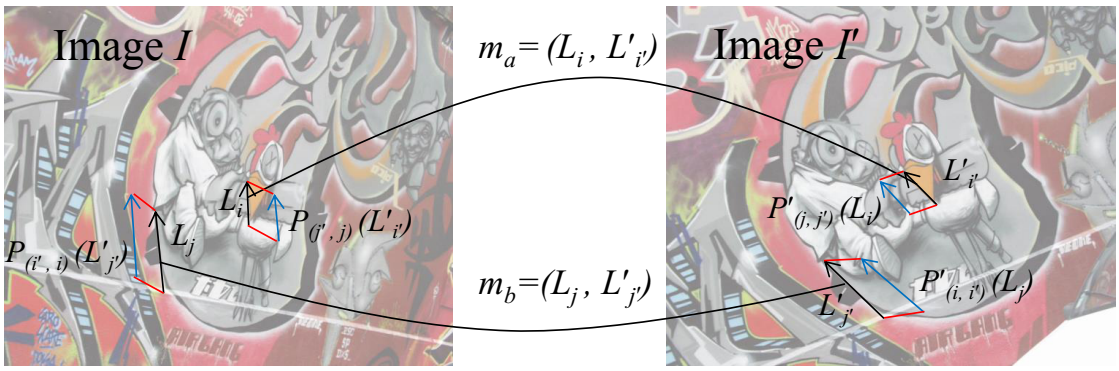


Figure 5.6: Projected oriented lines are in blue and $d(m_a, m_b)$ is the maximum length of red lines.

For a pair of matches $m_a = (L_i, L'_{i'})$ and $m_b = (L_j, L'_{j'})$, our pairwise consistency error measure $d(m_a, m_b)$ is the maximum segment distance of possible line projections as defined by Equation (5.4), which is also illustrated in Figure 5.6.

$$\begin{aligned} d(m_a, m_b) = max(&d_{seg}(P_{(i,i')}(L_j), L'_{j'}), \\ &d_{seg}(P_{(j,j')}(L_i), L'_{i'}), \\ &d_{seg}(P'_{(i',i)}(L'_{j'}), L_j), \\ &d_{seg}(P'_{(j',j)}(L'_{i'}), L_i)). \end{aligned} \tag{5.4}$$

This error features some scale invariance in the sense that re-scaling the smaller scale images will not affect its value as long as the scale is still smaller than that in the other image. To adapt to line segment we use this error measure to replace the ones used for points in the geometric constraint of K-VLD.

5.3.7 Semi-local photometric constraint

Local photometric segment description is hardly robust because gradients are dominated by the normal direction near the line segments. Moreover, a side of the line segment could be occluded. Using photometric information between two line segments can reinforce the matching constraints. This calls for an adaptation to line segment of the K-VLD method defined for points.

The basic idea of K-VLD for point matching is: For each match m_a, it checks the photometric consistency with respect to a number of neighboring matches m_b through a virtual line descriptor (VLD) between m_i and each m_b, see Figure 5.7. m_i and m_j are likely to be gVLD-consistent (geometric and photometric consistency) if m_a and m_b are both correct matches. If there are more than K neighbors (a heuristic constant) that are gVLD-consistent with m_a, then m_a is considered as a correct match.

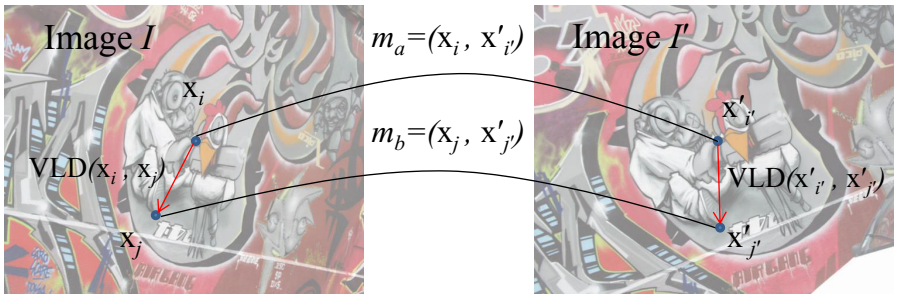


Figure 5.7: illustration of virtual line descriptors for a pair of point matches m_a and m_b, they are gVLD-consistent when the two VLDs differ little.

The objective is to modify VLD to adapt it to line segment matching. A crucial problem to apply the K-VLD method to line segments is to find a stable point on the segments that lays on the same location in I and I', in order to build a reliable VLD descriptor.

Technically, to check the photometric consistency for a pair of matches $m_a = (L_i, L'_{i'})$ and $m_b = (L_j, L'_{j'})$, we need to build a pair of VLDs for (L_i, L_j) and $(L'_{i'}, L'_{j'})$. Our algorithm uses the line segment end-point that has the least projection error.

Formally, we note $E_{a,b}(L_i)$ the end-point that L_i of m_a uses for constructing the VLD with $L_j \in m_b$

$$E_{a,b}(L_i) = \begin{cases} A_i \text{ if } ||P_{(j',j)}(A'_{i'}) - A_i|| < ||P_{(j',j)}(Z'_{i'}) - Z_i|| \\ Z_i \text{ otherwise} \end{cases}$$

By the similarity property, the choice of the end-point is always consistent for L_i and $L'_{i'}$.

$$\begin{aligned} E_{a,b}(L'_{i'}) = A'_{i'} \quad &\Leftrightarrow \quad E_{a,b}(L_i) = A_i \\ E_{a,b}(L'_{i'}) = Z'_{i'} \quad &\Leftrightarrow \quad E_{a,b}(L_i) = Z_i \end{aligned}$$

We build the pair of VLDs for (L_i, L_j) and $(L'_{i'}, L'_{j'})$ as

$$\begin{aligned} VLD(L_i, L_j) &= VLD(E_{a,b}(L_i), E_{b,a}(L_j)) \\ VLD(L'_{i'}, L'_{j'}) &= VLD(E_{a,b}(L'_{i'}), E_{b,a}(L'_{j'})) \end{aligned} \tag{5.5}$$

Note that for different neighbors, different extremities of lines of m_a can be chosen to construct VLDs.

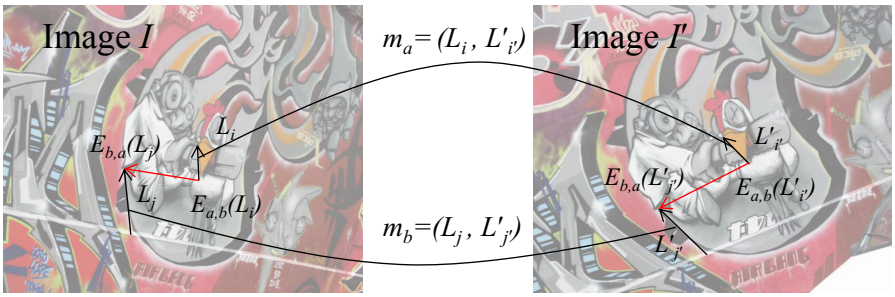


Figure 5.8: illustration of VLD for a pair of line segment matches m_a and m_b. Red lines are the paths for the VLDs.

5.3.8 K-VLD method for line segments

We consider a set of initial candidate matches $\{m_1, m_2, \ldots\}$. For each match $m_a = (L_i, L'_{i'})$ and each $m_b = (L_j, L'_{j'})$, we check if m_b is a neighbor of m_a in the same way as K-VLD, using the middle point of each segment such that $||C_i - C_j|| < B_K$ or $||C'_{i'} - C'_{j'}|| < B'_K$, where B_K and B'_K are defined as previously in Chapter 4, and C_i, C_j, $C'_{i'}$, $C'_{j'}$ are the respective center points of line segments L_i, L_j, $L'_{i'}$, $L'_{j'}$.

One additional constraint is that if m_a and m_b are nearly co-linear in one image, the VLD will not be discriminant, so we exclude neighbors on the same line. More formally, m_a and m_b are neighbors if the following three conditions are satisfied.

1. $||C_i - C_j|| < B_K$ or $||C'_{i'} - C'_{j'}|| < B'_K$
2. $angle(\overrightarrow{A_i Z_i}, \overrightarrow{C_i C_j}) > \alpha_{colinear}$ or $angle(\overrightarrow{A_j Z_j}, \overrightarrow{C_i C_j}) > \alpha_{colinear}$
3. $angle(\overrightarrow{A'_{i'} Z'_{i'}}, \overrightarrow{C'_{i'} C'_{j'}}) > \alpha_{colinear}$ or $angle(\overrightarrow{A'_{j'} Z'_{j'}}, \overrightarrow{C'_{i'} C'_{j'}}) > \alpha_{colinear}$

Our K-VLD segment matching algorithm is detailed by Table 5.1.

5.3.9 Experiments

For our experiments, we have considered two state-of-the-art algorithms for line segment detection: Line Segment Detector (LSD) [81] and ED-Lines [2]. Both methods are based on the *a contrario* theory, which allows an automatic adaptation of the main parameters of the detectors. In our experiments, we have found that the two detectors have different but roughly similar performance in terms of quality and number

Input: images I and I',
detected line segments in I and I',
set of weak (potential) initial matches M
Output: set of selected matches from M

1. Compute pyramids of scaled images for I and I'
2. Set up tables $t_{|M|\times|M|}$, $T_{|M|\times 1}$, $C_{|M|\times 1}$ and initially mark all matches as inliers
3. Do
4. Set T and C to zero values
5. For every inlier match m_i
6. For every inlier match m_j, a non-colinear neighbor of m_i
7. If $t[m_i, m_j]$ is undefined
8. If (m_i, m_j) are gVLD-consistent
9. then $t[m_i, m_j] \leftarrow \tau(m_i, m_j)$ and $t[m_j, m_i] \leftarrow \tau(m_j, m_i)$
10. else $t[m_i, m_j] \leftarrow$ discard and $t[m_j, m_i] \leftarrow$ discard
11. If $t[m_i, m_j] \neq$ discard
12. $T[m_i] \leftarrow T[m_i] + t[m_i, m_j]$
13. $C[m_i] \leftarrow C[m_i] + 1$
14. If $C[m_i] \geq N_{\max}$ then pass to the next m_i
15. $T[m_i] \leftarrow T[m_i]/C[m_i]$ if $C[m_i] \neq 0$

16. Sort inliers by increasing values of C, or increasing values of T if C-equal
17. (a) For every inlier m_i in sort order
18. If $C[m_i] < K$ then mark m_i as an outlier

19. (a') For every inlier m_i and every inlier m_j in sort order
20. If m_j conflicts with m_i, mark as outlier the match with lowest C,
. or lowest T if C-equal

21. (b) For every inlier m_i in sort order
22. Compute the proportion ω of geometric-consistent neighbors of m_i
23. Compute the average transformation error $\bar{\chi}$ amongst neighbors of m_i
24. Mark m_i as outlier if $\omega < \omega_{\min}$ or $\bar{\chi} > \bar{\chi}_{\max}$
25. while some matches have been marked as outliers (in this iteration)
26. If $\rho < \rho_{\min}$ go back to line 3 with $\rho_{min} = \rho_{min}/2$
27. Return matches marked as inliers

Table 5.1: The K-VLD segment matching algorithm

of detections. In practice, we have used LSD because the code is open-source, which facilitates the implementation of our algorithm.

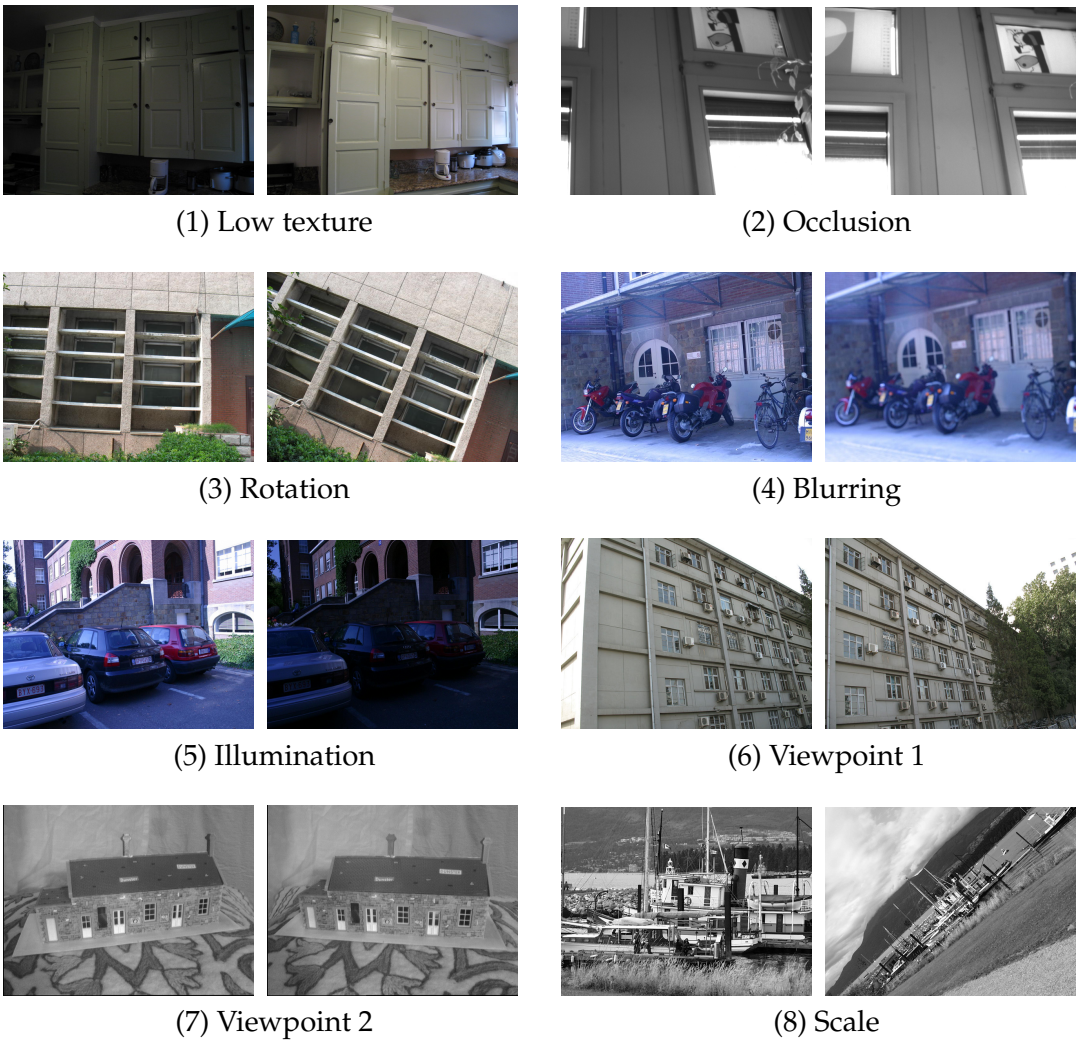
The number of line segments generated from the tested images varies approximately from 300 to 3000. For simplicity, we choose in our experiments a fixed value of N=15000 initial potential matches (see Equation 5.1), rather than rely on a function depending on the number of detections in both images. Finding a good heuristics to discover a large number of correct matches while keeping the computation time low is left for future work. .

We tested our algorithm on two datasets. The first dataset is the one that is commonly used in related work. It allows us to compare with other methods in various situations. The second dataset is more difficult: it features indoor scenes with little and repetitive texture. Our algorithm in this case outperforms the state-of-the-art algorithm

both in number of matches and in precision.

Common dataset, covering various situations

The first dataset is a collection of image pairs of various scenes used for evaluating line-matching algorithms. The corresponding images are shown on Figure 5.9. This dataset is used for comparison in [25]. It also has a large overlap with test images used in [90, 82]. We compare our algorithm 4 other methods: LBD [90], MSLD [83], LP [25] and LS [82]. We compute both the number of output matches and the match precision. The correctness of matches is verified manually. (Line segment detections and corresponding matches are not part of the dataset.)



(1) Low texture (2) Occlusion

(3) Rotation (4) Blurring

(5) Illumination (6) Viewpoint 1

(7) Viewpoint 2 (8) Scale

Figure 5.9: Image pairs of various scenes used in [90, 82, 25].

Results are illustrated in Figure 5.10. We observe that our algorithm retrieves more or a comparable number of true positive matches in most cases. But sometimes, our algorithm gives a slightly higher number of false positives, compared to the other methods Still, this dataset does not present the kind of difficulty we are trying to address, such as indoor scenes with little texture and wide baselines.

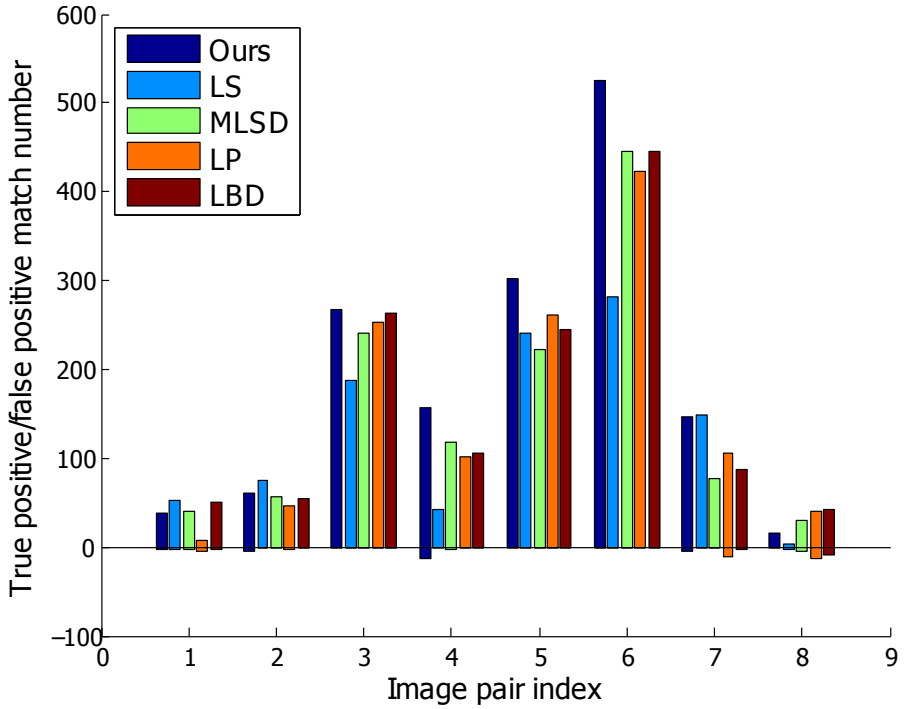


Figure 5.10: Comparison of K-VLD for segments with the state-of-the-art line-matching algorithms. Above 0: numbers of true positive matches. Under 0: numbers of false positive matches. (There were only a few of them in this test.)

Difficult scenes

As a matter of fact, line segment matching has less interest for outdoor or textured scenes since feature point matching is more suitable in this situation. Segment matching is indeed more helpful in the case of indoor and low-textured scenes.

We took a sequence of 10 pictures taken indoors, going round a long meeting room. The images are shown in Figure 5.11. We consider all pairs of contiguous images, plus image pair 10-1. In this dataset, the baseline is large as well as the rotation angle (36 degrees on average) and the variation in depth. Besides, there are repeated elements: similar tables, widows and chairs, and similar features on the ceiling. Note also that there are many textureless areas, where a point matching algorithm such as SIFT has a limited performance. With this dataset, we compare our algorithm for line segment matching with the LBD matching method, as LBD has a good and stable performance in the previous tests.

The detailed comparison is shown in Figure 5.12. Unlike with the previous dataset, where our results are similar as the other methods, in this dataset our method performs significantly better, especially when the baseline between the two pictures is wide. Our algorithm generates up to 10 times more matches than LBD. For pairs 7-8 and 8-9, all LBD matches are false, whereas our precision is higher than 82%, with a large number of true positive matches.

On Figure 5.13, we further visually illustrate the difference in comparison using pairs 1-2 and 3-4, where we have a larger output. We see that under a wide baseline and with a large disparity of depth in images 3 and 4, LBD yields false results for most matches. We also test both methods on the totally uncorrelated image pair 1-5, where no match should be generated, as our method does, while LBD finds 39 false matches.

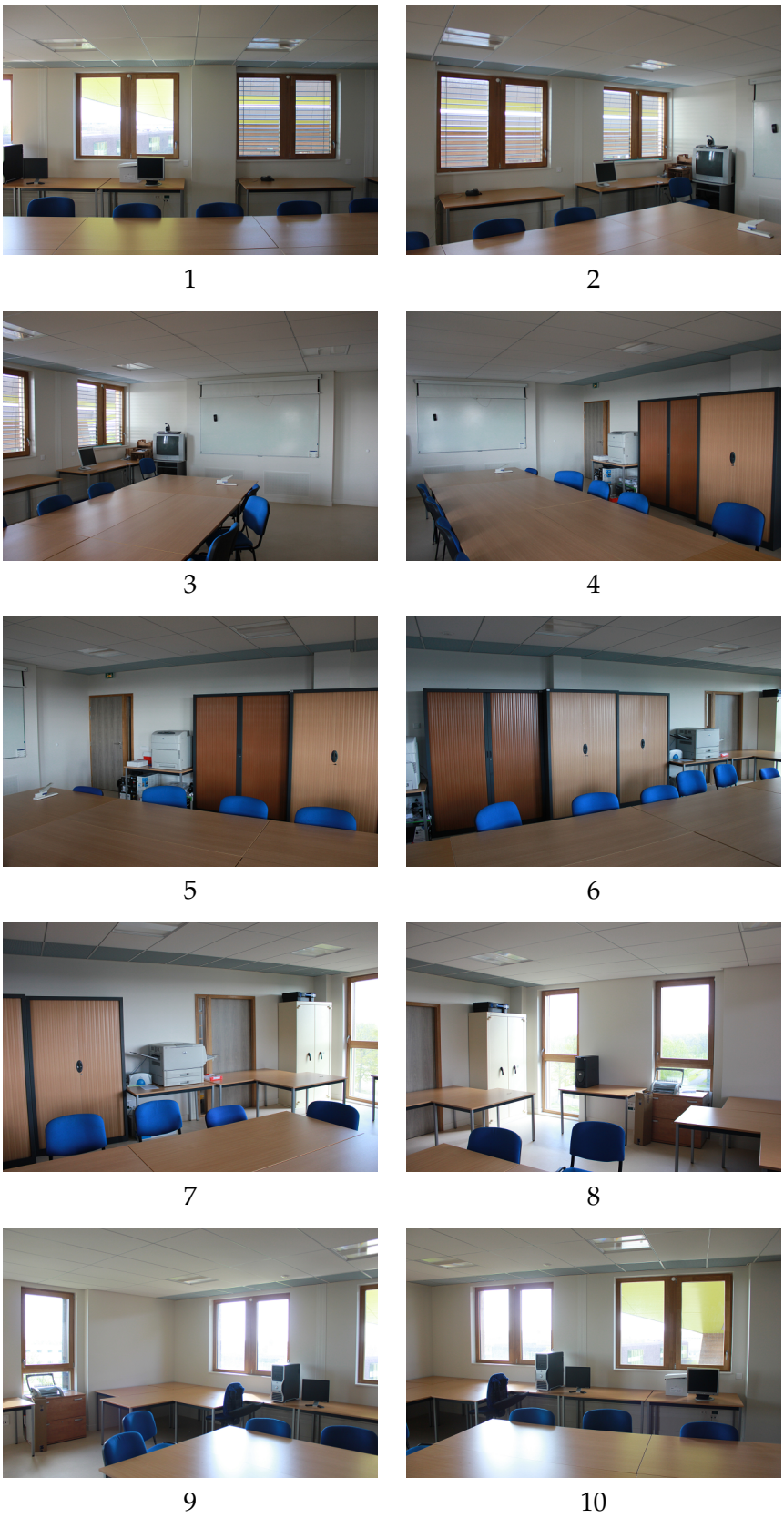


Figure 5.11: Difficult indoor dataset made of pictures of a long meeting room, with a wide baseline, repetitive patterns, large variation in depth and few textures. We consider all pairs of contiguous images, plus pair 10-1.

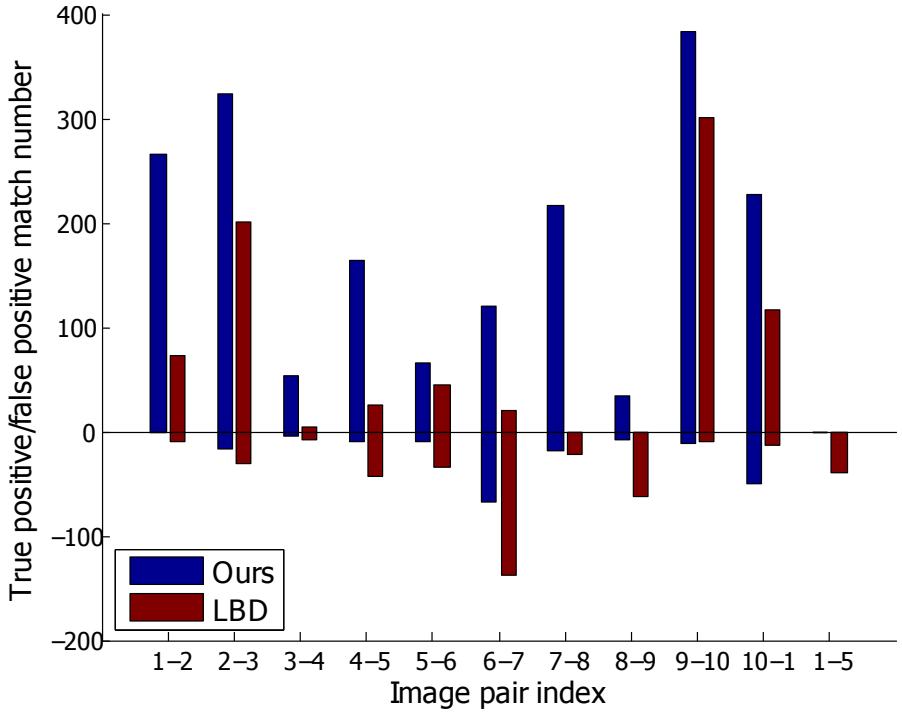


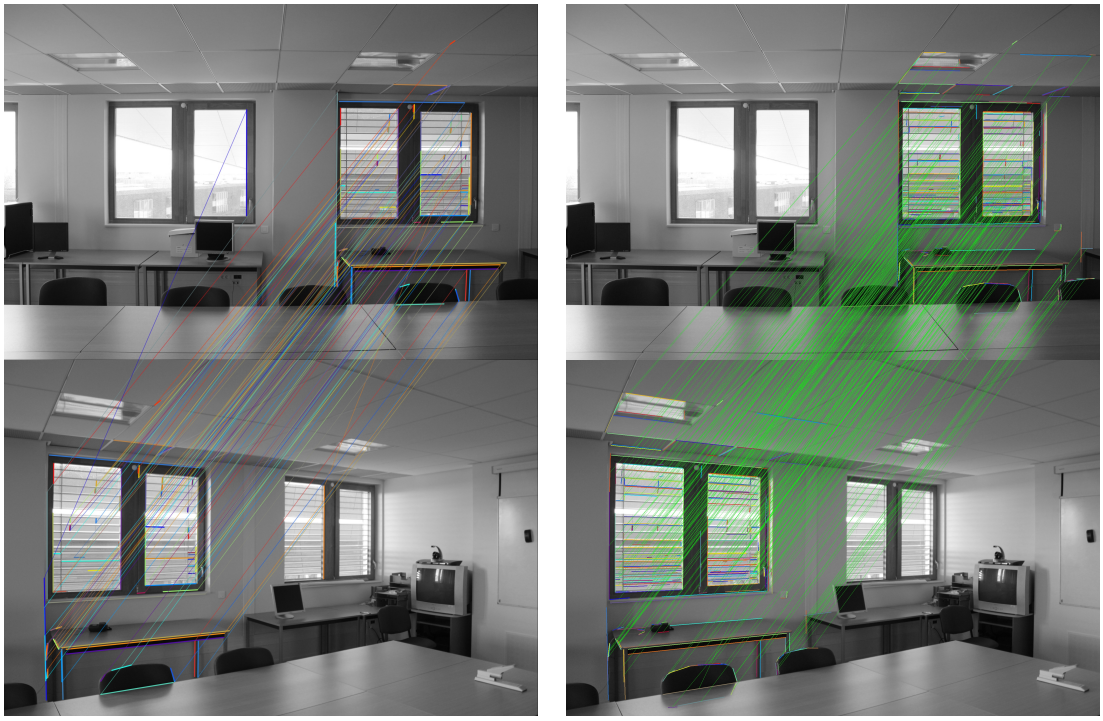
Figure 5.12: Comparison of line segment matching between K-VLD for segments and LBD, on the difficult meeting room dataset. Above 0: numbers of true positive matches. Under 0: numbers of false positive matches.

5.3.10 Computation time

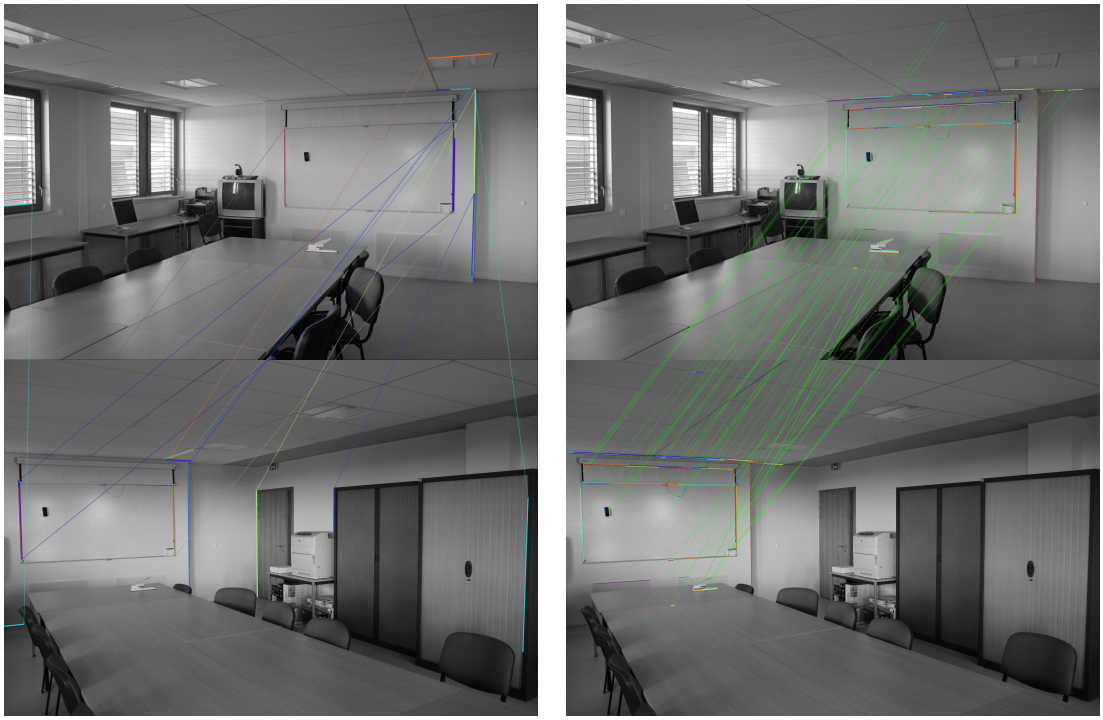
The computation time of our algorithm depends essentially on the number of initial (potential) matches, which we set heuristically to a fixed value. We are slower than LBD for small images, and much faster than LBD for large images. (LP and LS are far slower than LBD.)

5.3.11 Conclusion

We have presented here a modified K-VLD algorithm for line segment matching. In our algorithm, a new geometric consistency criterion for segments is introduced and a strategy for constructing VLDs between line segments has been proposed. In our experiments, our algorithm has comparable performance in various ordinary situations, but it outperforms the state-of-the-art algorithms for difficult indoor and textureless scenes.



pair 1-2



pair 3-4

Figure 5.13: Detailed comparison between the LBD method (left) and our algorithm (right). K-VLD for segments outperforms LBD in both cases, especially for pair 3-4, where little texture is available and perspective transformation is important due to depth variation.

5.4 Deformable object matching

5.4.1 Introduction

We now experiment the K-VLD algorithm on non rigid scenes. In the case of a non rigid transformation, graph matching methods are commonly used to select correct matches. K-VLD, as explained before, only checks gVLD-consistency with neighbors, and thus also adapts to deformed situation. This flexibility enables more applications. We first illustrate the matching results of weakly and strongly deformed sheets of paper to compare the performance of K-VLD's semi-local matching method with available graph matching methods as (HGM) [88], (IPFP) [45] and (GTM) [3]. Then we compare the performance between selected methods in uncorrelated scene, where the second image is completely different apart from a small cropped area; a good graph matching method should detect only matches inside the cropped area.

5.4.2 Experiments

We compare K-VLD with HGM, IPFP and GTM, for they have relatively good results in previous tests. There are no scale changes in the used dataset, which makes point distance a valid pairwise geometric constraint for HGM, IPFP and GTM. HGM follows a "matching until conflict" strategy, which includes many mismatches. IPFP and GTM output a ranking of matches. To be able to compare K-VLD with them, we extract the same number of best matches as the number of matches in K-VLD's output.

Weak deformation: The first scene is a paper map, after being moderately creased. We consider in Figure 5.14 feature matches from the upper image to the lower image. There are 3438 features in the upper image and 6225 images in the lower image. 3438 matches (upper to lower image) are generated for processing by the nearest neighbor strategy.

K-VLD returns 427 filtered matches within 8 seconds; matches are visually correct as shown in Figure 5.14. We show HGM's "matching until conflict" strategy is prone to mismatches, and its best 427 matches in Figure 5.15. The best 427 matches of IPFP and GTM are shown in Figure 5.16. The best matches of HGM, IPFP and GTM are globally very close to the right matches, however a closer look indicates that some matches actually are not correct.

Strong deformation: The second scene (cf. Figure 5.17) is a strongly deformed scene, that additionally contains a relatively large number of candidate matches. There are 11350 features in the upper image and 8842 images in the lower image. 11350 matches (upper to lower image) are generated for processing by the nearest neighbor strategy.

K-VLD returns 662 filtered matches in 17 seconds. Matches are visually correct (cf. Figure 5.17) apart from a small self-consistent group of features whose correct matches are hidden due to paper deformation, but that match exactly-identical text somewhere else on the sheet of paper. Compared to the moderate deformation case, performance drops in the strong deformation case for HGM, IPFP and GTM, because these methods are not robust to a drop of global geometric consistency.

Uncorrelated scene: Finally we test the performance of K-VLD, HGM, IPFP, GTM methods for a pair of images that are almost uncorrelated. Compared to the upper images, the lower image contains a totally different sheet of paper, apart from the cropped

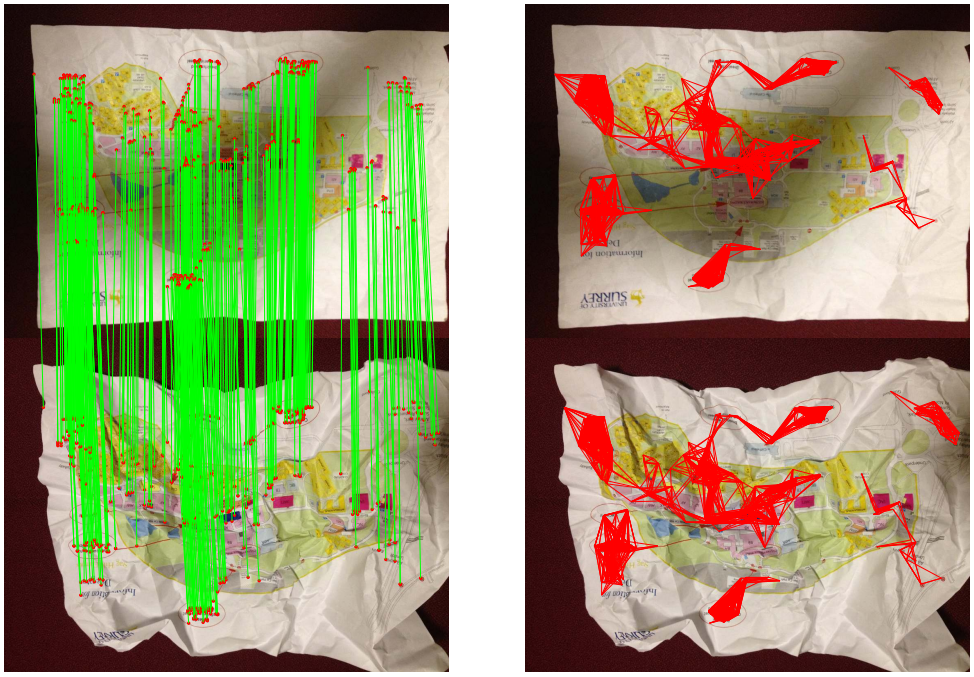


Figure 5.14: Illustration of K-VLD result for a weakly deformed scene. K-VLD returns 427 filtered matches out of 3438 matches in 8 seconds. Left: feature correspondence by K-VLD is in green. Right: Valid gVLD-consistent virtual lines are in red. (N.B. Not all agreeing neighbors are marked; for each match, only the K first gVLD-consistent neighbors are checked by K-VLD and drawn in the image.)

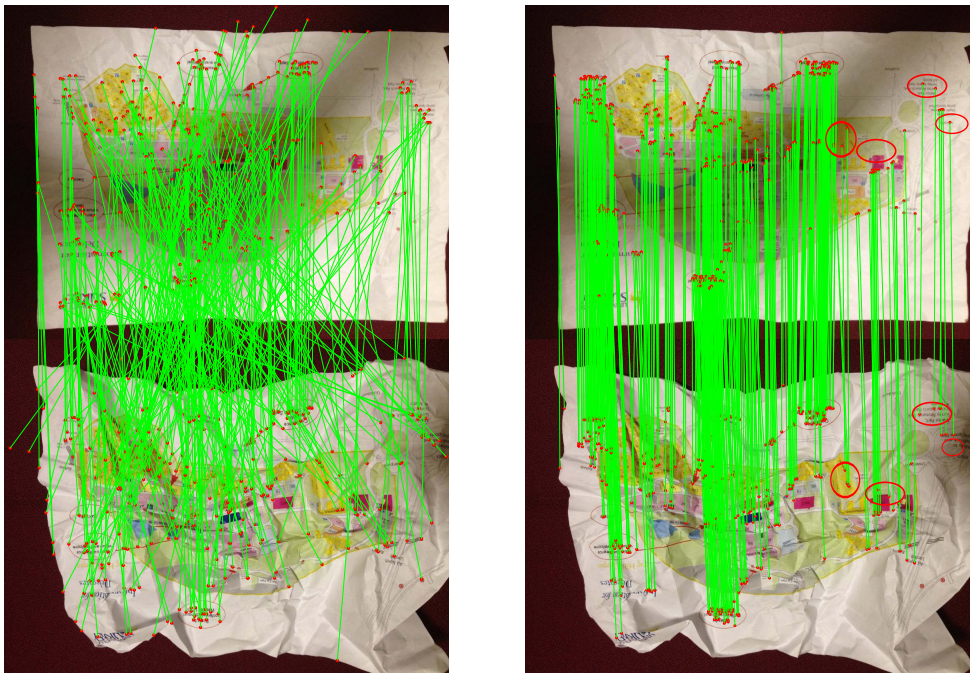


Figure 5.15: Illustration of HGM's result for a weakly deformed scene. Left: HGM's "matching until conflict" strategy selects 1966 matches, and is prone to mismatches. (Only one in every six matches is shown for visibility). Right: keeping only the best 427 matches by HGM for comparing with K-VLD, the matches seem correct. However, a finer look shows that a number of matched features are not at the right location (cf. areas circled in red, not exhaustive).

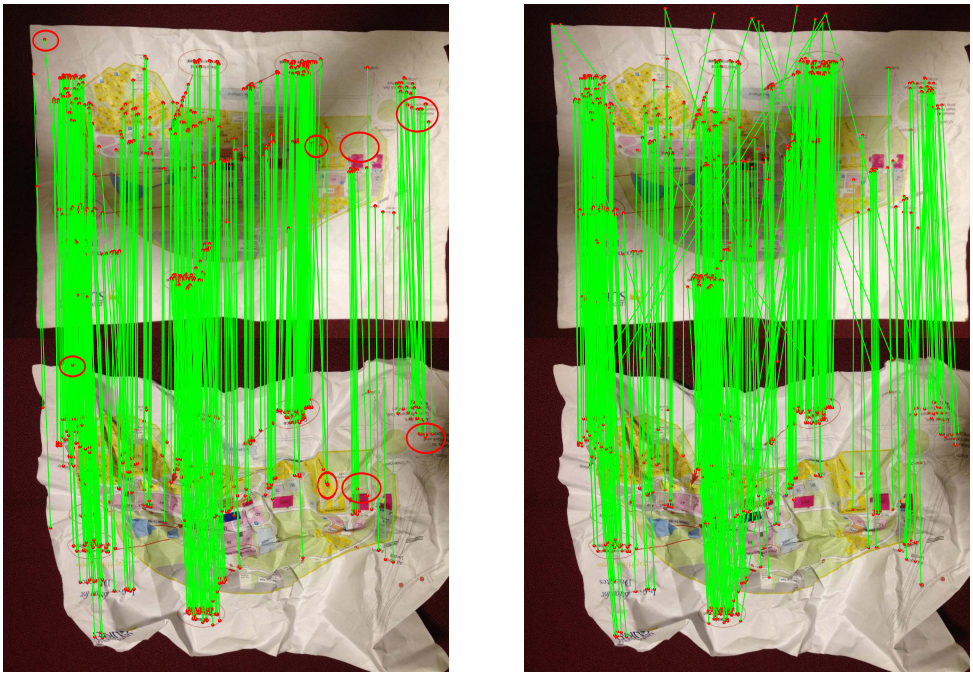


Figure 5.16: Illustration of IPFP's and GTM's results for a weakly deformed scene, keeping only the 427 best matches for comparing with K-VLD. Left: as for HGM, thebest IPFP matches seem correct, but a finer look shows that a number of matched features are not at the right location (cf. areas circled in red, not exhaustive). Right: in the best matches found by GTM, mismatches are visible.

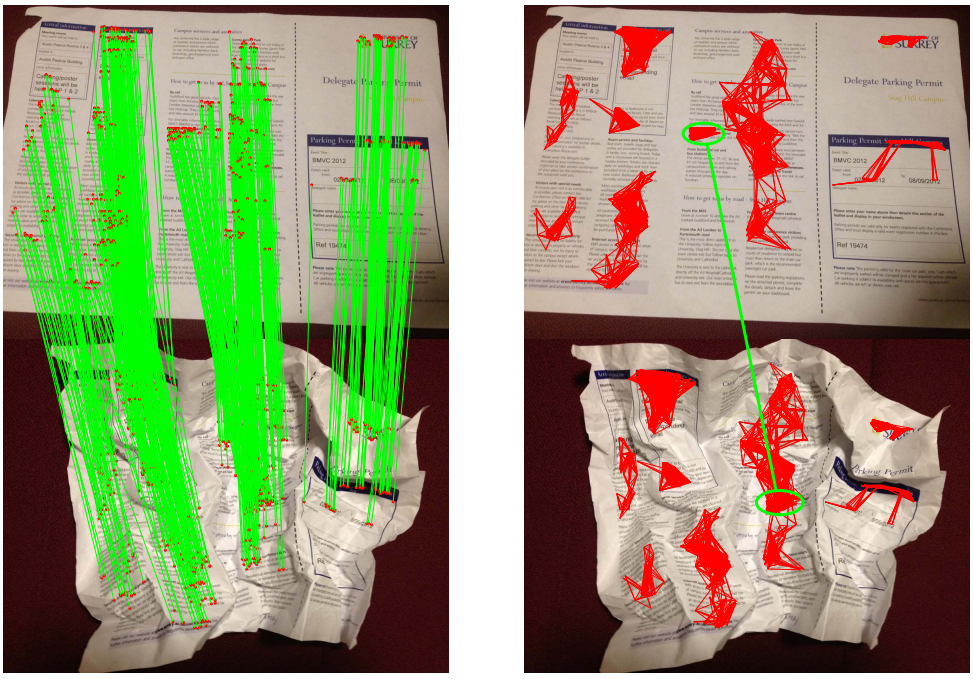


Figure 5.17: Illustration of K-VLD's result for a strongly deformed scene. K-VLD returns 662 filtered matches out of 11350 candidate matches in 17 seconds. Left: feature correspondence by K-VLD in green. Right: gVLD-consistent virtual lines in red. (For each match, only the K first gVLD-consistent neighbors checked by K-VLD are drawn.) Only a small self-consistent group of features are mismatched, to another exactly-identical text area, due to the lack of correct matches (cf. areas circled in green).

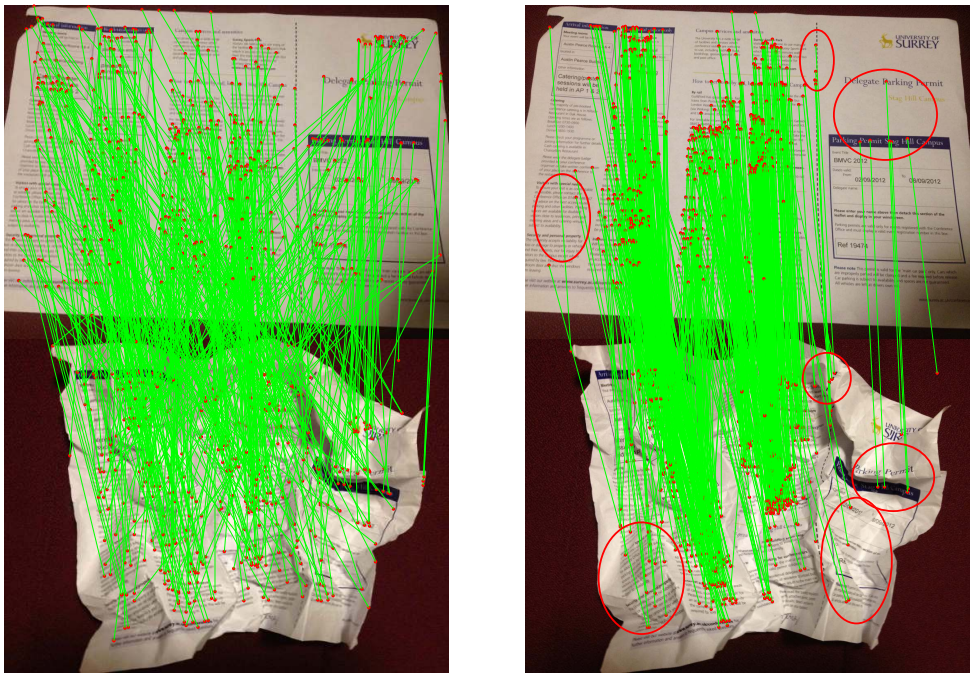


Figure 5.18: Illustration of HGM's result for a strongly deformed scene. Left: HGM's "matching until conflict" strategy is again prone to mismatches. (Only one in every six matches is shown for visibility.) Right: the best 662 matches by HGM contain visible mismatches. Mismatches are still present because global geometric consistency is not preserved due to the strong deformation. Only separate small patches, as identified by K-VLD (cf. Figure 5.17), are consistent enough.

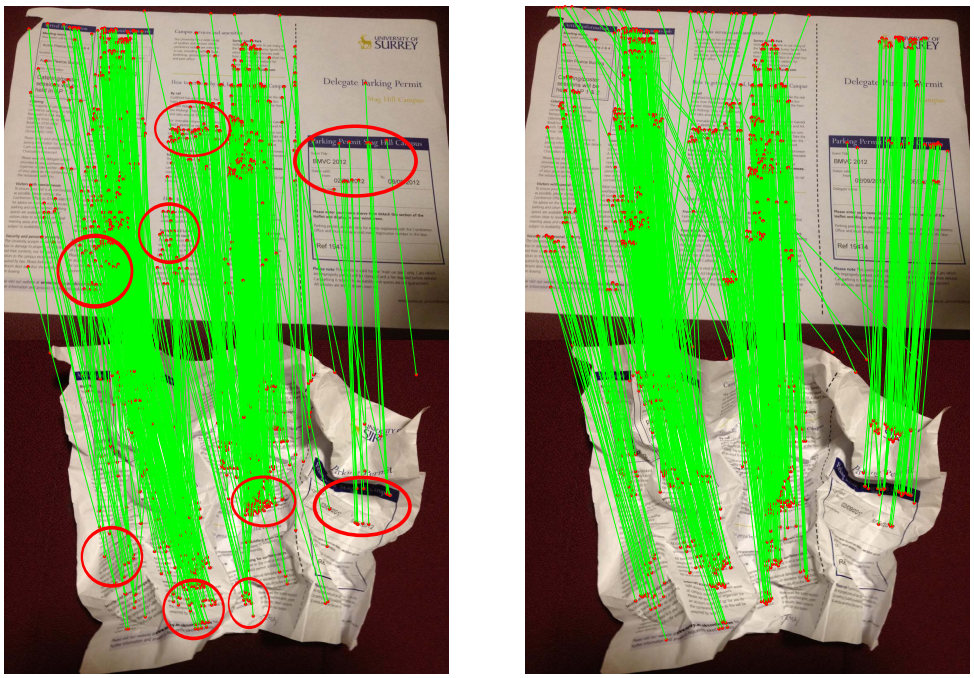


Figure 5.19: Illustration of IPFP's and GTM's results for a strongly deformed scene. The 662 best matches are extracted for each method. Left: the best matches by IPFP contains visible mismatches (cf. areas circled in red, not exhaustive). Right: in the best matches by GTM, mismatches are visible too. As above, the performance is poor because the methods are not robust to a drop of global geometric consistency.

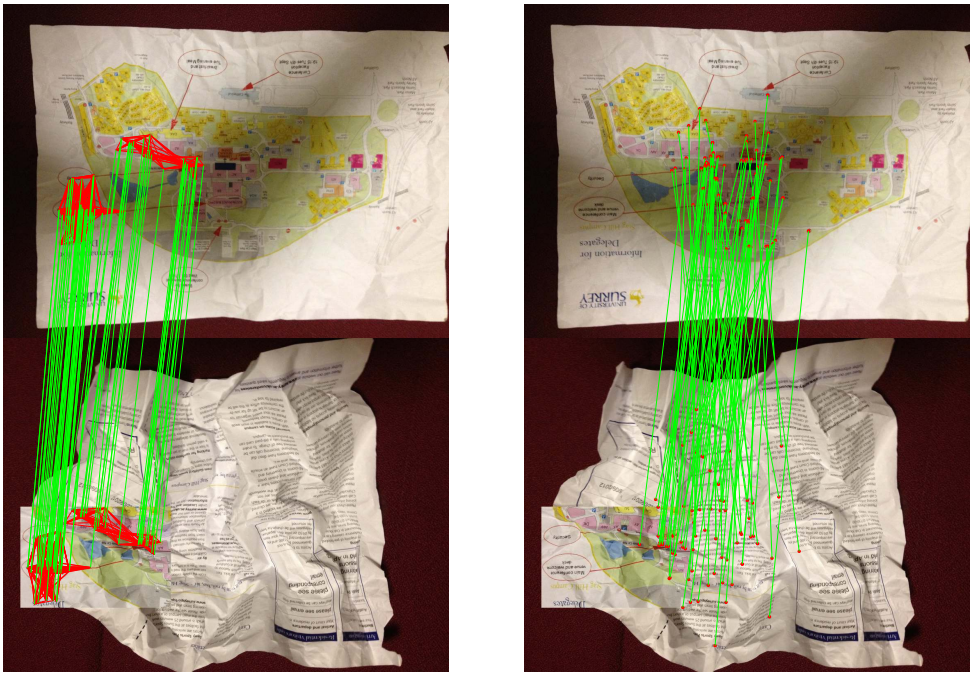


Figure 5.20: Illustration of K-VLD's and HGM's result on uncorrelated images (except the cropped part). Left: feature correspondence by K-VLD in green. Right: in the best matches by HGM, mismatches are visible.

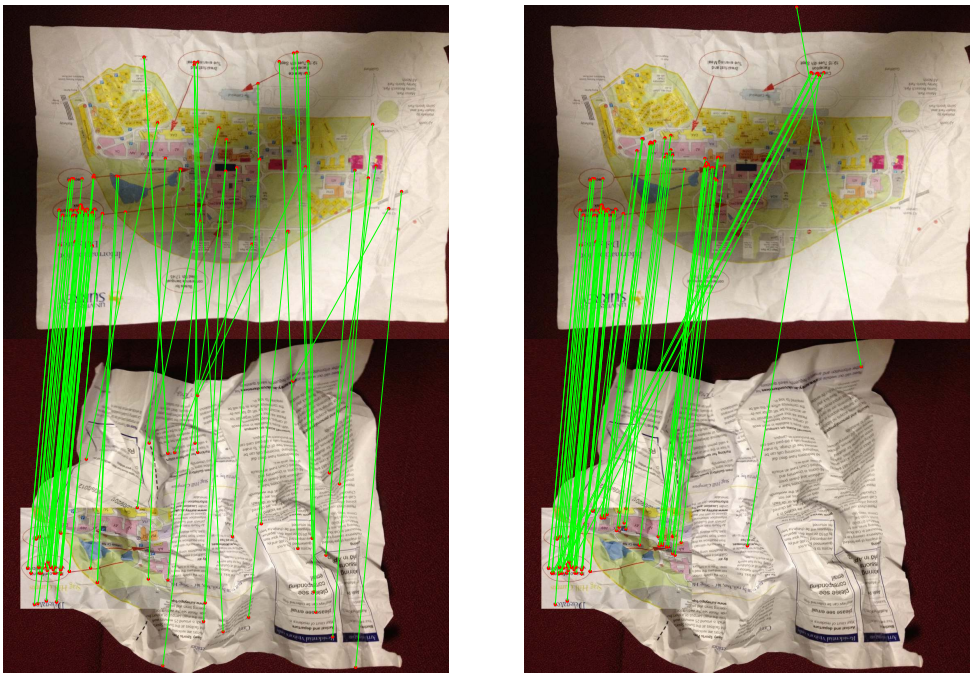


Figure 5.21: Illustration of IPFP and GTM's result on uncorrelated images (except the cropped part). Left: in the best matches by IPFP's, mismatches are visible. Right: in the best matches by GTM, mismatches are visible too.

part (a deformed region containing relevant information). The reason to include thus cropped region is to add just a small number of actual matches, which we can easily verify in the output of the matchers. This test is important, because a good graph matching method should not only find common structures, but also be discriminant to exclude unrelated features. SIFT features are extracted and matched by the nearest neighbor strategy, before the matching methods are applied. The result (cf. Figures 5.20 and 5.21) demonstrates the relevance of a photometric constraint verification, as geometric constraint can accidentally be satisfied.

5.5 Urban localization from Google street views

5.5.1 Introduction

In recent years, many efforts have been made in the development of autonomous vehicles, resulting in innovative technologies such as the Google car and unmanned aircraft. Localization is a crucial problem for autonomous vehicles. Localization systems such as GPS is a popular solution, however a parallel independent visual localization system is expected as well, to face challenging environment. For instance, the GPS signal in urban environment can be shadowed, unavailable or echoed. Besides, GPS positioning is not very accurate. Alternative technologies become thus necessary for vehicle localization.

5.5.2 K-VLD contribution in air-ground image matching

With the database of Google Street View images becoming available, visual urban localization has become reachable. However, the state-of-the-art visual-localization systems may perform poorly due to feature matching challenges because of significant viewpoint changes, changes of illumination, over-season variation, lens distortions, etc...

RANSAC-based methods are one option to clean up feature correspondences, however the lens distortion due to the wide angle dramatically breaks the rigid transformation hypothesis, making RANSAC performing poorly.

In previous sections, we have illustrated the performance of K-VLD for deformable object matching. By design, K-VLD uses a soft semi-local geometric constraint so that image distortion does not perturb the result. Recent work has been carried out using K-VLD to match features for MAV geo-referencing. The work of Andras et al. [51] uses K-VLD to match features between micro aerial vehicle (MAV) images and Google Street View images. The work of Karel et al. [39, 38] applies K-VLD as an efficient matching method to overcome complex situations (e.g., large number of features, matching with a low inlier rate, images from different cameras, of different scales, etc...).

For more details in practical applications, the readers are kindly referred to related work [51, 39, 38].

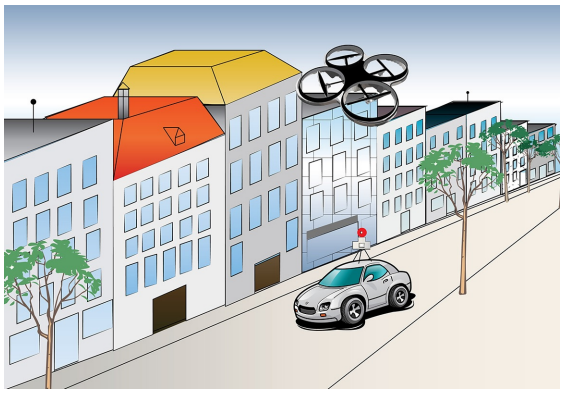


Figure 5.22: Illustrating image of MAV urban scene geo-localization borrowed from Andras et al. [51]. The global position of the MAV is computed by matching the aerial image taken by the flying vehicle with the closest ground-level geotagged Google Street View image.

Figure 5.23: Illustration of air-ground matching between images from AR Drone 2.0 and images from Google Street View. Images undergo lens distortion as well as significant viewpoint and scale changes. Left: K-VLD is robust against distortion, and extracts correct matches. gVLD-consistent paths are in red, matches are in green. Right: ORSA fails to find a model within 200,000 iterations in all three cases.

5.5.3 Experiment

We use the dataset provided by Andras et al. [51] to illustrate the robustness of K-VLD compared to RANSAC. The images are very challenging because of the lens distortion as well as the significant viewpoint and scale changes. Still, these images correspond to common cases for the application and it is crucial to be able to treat them appropriately.

We compare K-VLD to ORSA [57]. Whereas Andras et al. [51] used a modified ASIFT feature detection and description to generate a large number of features, we just use here a standard SIFT detector and descriptor and a first neighbor matching strategy to also show the robustness of K-VLD under a low rate of inliers, below 3%. Results are shown on Figure 5.23: K-VLD can still provide correct matches when RANSAC-based methods fail.

Chapter 6

Match refinement for highly accurate two-view SfM

In this chapter, we propose a variant of least squares matching to refine feature match locations based on a focused grid and a multi-scale exploration to enhance the accuracy of structure from motion (SfM) in the two-view case. We also propose a method to directly estimate the surface normal of refined blob-like features as SIFT when direct perspective transformation estimation is not reliable. Experiments show that our match refinement contributes to a better accuracy. Once combined with match selection introduced in the next chapter, it reduces SfM errors by a factor of 1.1 to 2.0 for rotation, and 1.6 to 3.8 for translation.

Contents

6.1 Introduction . . . 102
6.2 Least Squares Matching (LSM) . . . 102
6.2.1 Local geometric transformation . . . 103
6.2.2 Local photometric transformation . . . 103
6.2.3 Local transformation extimation . . . 104
6.3 Sampling density and coverage zone . . . 104
6.4 Least Square Focused Matching . . . 105
6.4.1 Image intensity variation . . . 106
6.4.2 Focused grid . . . 106
6.4.3 Scale exploration . . . 108
6.5 Parameters . . . 109
6.6 Impact of the kind of feature detector . . . 109
6.7 Experiments . . . 110
6.7.1 Covering factor evaluation . . . 110
6.7.2 LSM vs. LSFM . . . 110
6.8 Extension: Surface normal estimation . . . 111
6.8.1 Introduction . . . 111
6.8.2 Homography Recovery . . . 112
6.8.3 Normal illustration . . . 112
6.9 Conclusion . . . 112

6.1 Introduction

One way to obtain a better SfM accuracy is to improve the accuracy of feature detection. Due to differences in imaging conditions, in particular changes of viewpoint or illumination, a salient point or region detected in one image is not detected in the other image at the exact matching location. The most popular features are by design only invariant (at most) to affine transformation (e.g., Harris-affine [54], MSER [52], ASIFT [59]), or to small affine transformation (e.g., SIFT [49]). But they are not invariant under perspective transformation, which is enough to offset most detections. Methods to add some perspective invariance to existing feature detectors have been proposed, but they require full 3D information (depth map or mesh) [41, 84], which is computationally expensive or requires more than just image data; others are suited to specific classes of scenes only, mostly urban environments, as they strongly rely on the presence of vanishing points and large planar surfaces [6, 15]. Besides, they have been designed to improve the repeatability of feature detection and matching, which is generally measured using a threshold on the relative overlap of corresponding regions [54], not in terms of the closest distance between feature centers.

Traditionally, two detected feature points can still be considered as matching although their position in the images does not correspond exactly to the same 3D point in the observed scene. For a number of tasks, being close is enough. But for highly accurate calibration, it is not satisfactory. In fact, we do not care whether a specific 3D point is accurately identified in both images, such as the very tip of a corner. What we need is possibly arbitrary pairs of points in the images as long as they correspond to extremely close 3D points. In this sense, feature detection and matching is just a way for us to identify corresponding regions rather than corresponding points: their center generally corresponds only to close but different 3D points. The match refinement we propose here only uses them as initial estimates to find pairs of 2D points that are likely to correspond to closer 3D points because they have a better local photometric consistency (assuming an unknown affine transformation).

Finely relating image regions can be addressed with optical flow methods [7]. But these methods are not well adapted here because they suppose small variations, both in viewpoint (very small baseline, quasi-affine transformation) and in illumination (controlled light scenes). Refining the position of image regions to overlap them better has been studied in the photogrammetry community. One of the most popular methods is adaptive least squares matching (LSM), that tries simultaneously to find radiometric and geometric corrections to best fit two images patches [29]. The most complex geometric correction generally considered in this framework is affine transformation, because projective transformations are assumed to be well approximated by an affinity. We present here an improvement of LSM based on a focused irregular grid and made robust with coarse-to-fine exploration. We show that it outperforms affine correction.

6.2 Least Squares Matching (LSM)

LSM, originally called adaptive least squares correlation [29], is based on the hypothesis that, *locally*, the region around the feature center is mostly planar, so that two matching regions are approximately related by a homography, which in turn can be approximated by an affinity if the change of viewpoint is moderate.

Given a match $m = (\mathbf{x}, \mathbf{x}')$, LSM tries to best adjust the region around $\mathbf{x}$ in I to the region around $\mathbf{x}'$ in I', using some unknown affine transformation A, which also needs to be estimated. The initial value A_{init} is the similarity defined by the translation from $\mathbf{x}$

to $\mathbf{x}'$ and the rotation given by the difference of angles of the feature orientations. It is iteratively refined by choosing affinity parameters that minimize the dissimilarity between the two regions.

For all the following, we note G and G' the sampling grids used to interpret regions around $\mathbf{x}$ and $\mathbf{x}'$, with G' defined as the transformation of G by the affinity A.

LSM samples points on a regular grid of size $(2n+1)\times(2n+1)$. For a point (x,y) on G, its coordinate is calculated from $x = \mathbf{x}_x + su$, $y = \mathbf{x}_y + sv$, where $(\mathbf{x}_x, \mathbf{x}_y)$ is the coordinate of the feature point in I and s and s' are re-scaling factors such that $s/s' = scale(\mathbf{x})/scale(\mathbf{x}')$ and $\min(s,s') = 1$:

$$G = \{(\mathbf{x}_x + su, \mathbf{x}_y + sv) \,|\, u,v \in \{-n,\dots,0,\dots,n\}\}. \tag{6.1}$$

We use a spline interpolation of order 5 to retrieve subpixel intensity at each node of the grids.

6.2.1 Local geometric transformation

The points on grid G' in image I' are defined by the projection of the points on grid G in image I via the transformation A.

To be consistent with the dimensions of the homography matrix used later, we represent the affine transformation as a 3×3 matrix $A = (a_{ij})_{1\le i\le 3, 1\le j\le 3}$, with $a_{31} = 0, a_{32} = 0, a_{33} = 1$. We write $(x',y') = A(x,y)$ the transformation by the affinity A:

$$\begin{aligned} x' &= a_{11}x + a_{12}y + a_{13} \\ y' &= a_{21}x + a_{22}y + a_{23} \end{aligned} \tag{6.2}$$

The grid G' can thus be defined as $G' = A(G) = \{A(x,y) \,|\, (x,y) \in G\}$, i.e., the transformation of the grid G by the affinity A.

To make sure A is not badly conditioned (large values for a_{13} and a_{23}), the coordinates of the features are actually first shifted in both images to center the reference frames of I and I' at $(\mathbf{x}_x, \mathbf{x}_y)$ and $(\mathbf{x}'_x, \mathbf{x}'_y)$, i.e., so that $(\mathbf{x}_x, \mathbf{x}_y) = (\mathbf{x}'_x, \mathbf{x}'_y) = (0,0)$ in the new frames. Moreover, the feature coordinates are divided by the geometric mean of the image dimensions. The initial affinity A_{init} represents the rotation (difference of feature orientation) and re-scaling.

6.2.2 Local photometric transformation

To reduce the influence of light changes, a transformation of the intensity between the two image regions can also be considered. It is assumed that this transformation is affine, i.e., that there are a radiometric scale r_s and a radiometric shift r_t such that

$$I(x,y) \approx r_s I'(x',y') + r_t \tag{6.3}$$

In practice, the scale r_s and shift r_t are re-estimated at each iteration of the estimation process, as described by Potůčková [66, §1.2.1.2, pp.23-24]:

$$(r_s, r_t) = \underset{b_0,b_1}{\arg\min} \sum_{\substack{(x,y)\in G \\ (x',y')=A(x,y)}} (I(x,y) - b_1 I'(x',y') - b_0)^2 \tag{6.4}$$

Alternatively, r_s, r_t can be introduced as unknown variables instead of priors. They can then be estimated together with dA by a vector (r_s, r_t, dA). But it increases the dimension of the system and leads to slower convergence, which then requires more iterations [70].

6.2.3 Local transformation extimation

The image dissimilarity on the two grids is defined as the sum of square differences (SSD) of the photometric-transformed intensities:

$$\eta(A) = \sum_{\substack{(x,y)\in G \\ (x',y')=A(x,y)}} \left[r_s I'(x',y') + r_t - I(x,y)\right]^2 \tag{6.5}$$

To find in I' a better location than $\mathbf{x}'$ for the point corresponding to feature $\mathbf{x}$ in I, we look for a small displacement (dx',dy') of (x',y') that minimizes the dissimilarity $\eta(A)$:

$$\underset{dx',dy'}{\arg\min} \sum_{\substack{(x,y)\in G \\ (x',y')=A(x,y)}} \left(r_s I'(x'+dx',y'+dy') + r_t - I(x,y)\right)^2 \tag{6.6}$$

We actually look for a small change dA of the affinity A that would induce such a displacement (dx',dy'). Following Gruen [29], we consider a first-order approximation of A: given a small variation $dA = (da_{ij})_{1\le i\le 3,\, 1\le j\le 3}$ with $da_{31} = 0, da_{32} = 0, da_{33} = 1$, we look at the impact on dx', dy':

$$\begin{aligned} dx' &= da_{11}\,x + da_{12}\,y + da_{13} \\ dy' &= da_{21}\,x + da_{22}\,y + da_{23} \end{aligned} \tag{6.7}$$

We also consider a first-order expansion of the intensity in I':

$$I'(x'+dx',y'+dy') \approx I'(x',y') + g'_x(x',y')dx' + g'_y(x',y')dy' \tag{6.8}$$

based on gradients computed by finite difference:

$$\begin{aligned} g'_x(x',y') &= \frac{I'(x'+s',y') - I'(x'-s',y')}{2s'} \\ g'_y(x',y') &= \frac{I'(x',y'+s') - I'(x',y'-s')}{2s'} \end{aligned} \tag{6.9}$$

Wrapping up, using in (6.6) the expansion of $I'(x'+dx',y'+dy')$ in (6.8) as well as the expression of (dx',dy') in (6.7), we obtain a convex system for dA:

$$\begin{aligned} \underset{dA}{\arg\min} \sum_{\substack{(x,y)\in G \\ (x',y')=A(x,y)}} \big[& r_s \left(I'(x',y') + g'_x(x',y')\left(da_{11}\,x + da_{12}\,y + da_{13}\right)\right. \\ & \left. + g'_y(x',y')\left(da_{21}\,x + da_{22}\,y + da_{23}\right)\right) + r_t - I(x,y)\big]^2 \end{aligned} \tag{6.10}$$

By any classic regression method, we can obtain a solution for dA. We then adjust the affinity A correspondingly for the next iteration: $A_{i+1} \leftarrow A_i + dA$. LSM indeed iterates the computation of A. At each iteration, the radiometric correction r_s, r_t as well as the gradients g'_x, g'_y are recomputed in function of the new estimation for A. The algorithm iterates as long as the dissimilarity $\eta(A)$ decreases with a newly interpreted A, or until a fixed maximum number of iterations is reached.

The LSM algorithm is pictured in Figure 6.1. To refine a set of matches between image I and I', the LSM algorithm is applied to each match independently.

6.3 Sampling density and coverage zone

In LSM, and in many other methods using a grid of intensity samples, the reference grid G (the grid that does not change) has a configuration of 1 sample per pixel square. Yet, we have to distinguish the sampled area, of size $k \times k$ pixel square, and the sampling density $(2n+1) \times (2n+1)$, for they have different impacts on LSM:

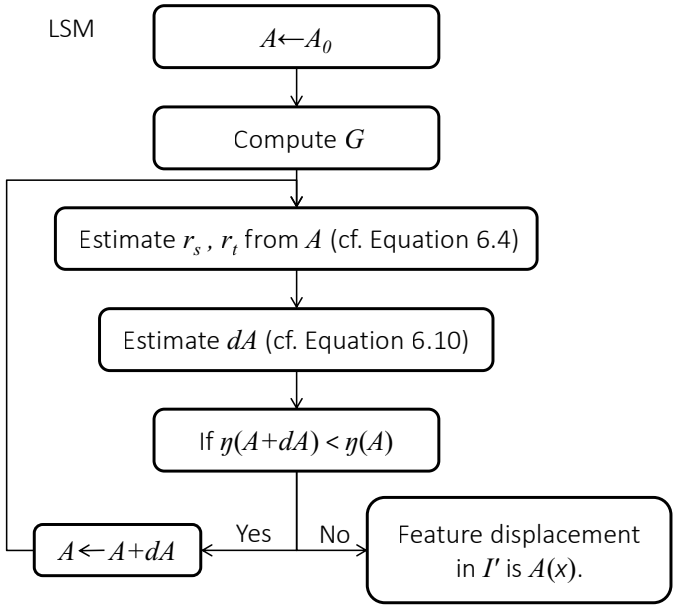


Figure 6.1: LSM algorithm (after .

- If the sampled area is too large (large k), there are more chances for the covered region to be non-planar, or approximately non-planar; it will then appear dissimilar under a different viewpoint, even if corrected by an affine or homography transformation. If the area is too small (small k), the pixel information is insufficient to provide a reliable transformation estimation.

- Besides, the larger the sampling density (large n, within resolution bounds), the more accurate the estimation of the transformation, but also the more computationally expensive.

Setting the values of k and n thus requires heuristic compromises.

In the following, we choose to set first the value of n, as a balanced compromise with respect to computation time. Given this relative sampling density, we then define the size of the sampled area as the best observed size based on experiments with images in a dataset. Note that the size of the sampled areas, modulo the relative scales of the detected features (cf. Equation 6.1), is the same for all matches, both in our learning experiments to define the best (average) size, and in the setting of our test experiments.

6.4 Least Square Focused Matching

We now present a Least Square Focused Matching (LSFM), which is an extension of Least Square Matching (LSM) that better adjusts the location of matched features.

With LSFM, we propose two improvements. First, instead of using a regular sampling grid around the features, we use an irregular grid focused on the center of the region to match. Second, we perform an image-scale traversal, to make transformation estimation more robust to local minima. (We also tried estimating a homography rather than just an affinity, but it did not produce substantial improvements, cf. Section 6.8.1.)

Note that feature detection covariance [10, 37, 74, 89] is irrelevant here. What we do is, as LSM, given a position p in I for which we know a roughly corresponding position p' in I', adjust p' so that the regions around p in I and p' in I' correlate better, under some geometric and photometric affinity to estimate. Feature points that match

just happen to provide good initial correspondences (p, p') for the refinement process. This is also widely different from refining the location of features detected as salient [49, 54]. Moreover, refining given matches leads to a better accuracy than refining detections before matching.

6.4.1 Image intensity variation

LSM calculate the values of r_s, r_t by means of a linear regression from Equation (6.4). We do it in a different way, by normalizing the average intensity and the intensity variance on grid G', so that it the same as on grid G. Concretely, we define the average intensity and the intensity variance as:

$$\bar{I}_G = \frac{1}{|G|}\sum_{(x,y)\in G} I(x,y) \qquad \bar{I}'_{G'} = \frac{1}{|G'|}\sum_{(x',y')\in G'} I'(x',y')$$
$$\sigma_G = \sqrt{\frac{\sum_{(x,y)\in G}(I(x,y)-\bar{I}_G)^2}{|G|}} \qquad \sigma'_{G'} = \sqrt{\frac{\sum_{(x',y')\in G'}(I'(x',y')-\bar{I}'_{G'})^2}{|G'|}}.$$

where $|G| = |G'| = (2n+1)^2$ is the number of samples in the grid. We look for r_s, r_t such that:

$$\begin{aligned} \bar{I}_G &= \bar{I}'_{G'} + r_t \\ \sigma_G &= r_s \sigma'_{G'} \end{aligned} \tag{6.11}$$

In our experiments, using this estimation for r_s, r_t provides a slightly better robustness (and speed). This is what we have used in LSFM.

6.4.2 Focused grid

The image dissimilarity measure η in LSM is based on a regular sampling grid centered on interest points. This assumes a uniform transformation of the associated image area, which is basically true close to the grid center and slowly breaks down when moving away towards the grid periphery. Besides there is an anisotropy due to the grid shape and alignment.

For this reason, in LSFM we propose two improvements. We use a grid G_F focused on the patch center, i.e., denser in the center than in the border, which is additionally weighted by a Gaussian kernel to further concentrate on the center and provide rotation invariance. Concretely, we use a grid whose nodes are placed in a geometric progression with respect to the grid center, with fixed ratio. As previously, the grid is possibly also scaled up with respect to the detection scale ratio of the matching features:

$$\begin{aligned} G_F = \{(\mathbf{x}_x + \triangle x(u), \mathbf{x}_y + \triangle y(v)) \,|\, u,v \in \{-n,\dots,n\}, \\ \triangle x(u) = s\lambda\, sign(u)\frac{\rho^{|u|}-1}{\rho-1}, \\ \triangle y(v) = s\lambda\, sign(v)\frac{\rho^{|v|}-1}{\rho-1}\} \end{aligned} \tag{6.12}$$

where λ is a covering factor determining the extension of the sampled area, s is the re-scaling factor (cf. Section 6.2) and ρ is the fixed ratio of the geometric progression. The points of G'_F in image I' are defined by the projections of points of G via the transformation A: $G'_F = A(G_F)$.

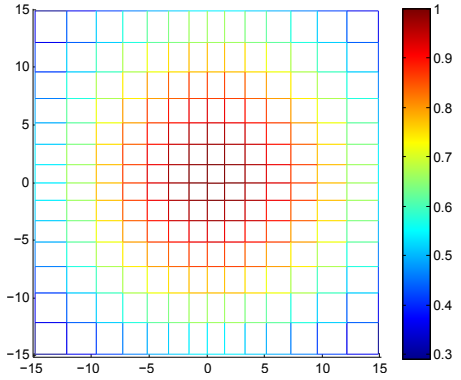


Figure 6.2: Representation of a focused grid. Sampling is denser and heavier at the center. Colors represent the Gaussian weight.

Besides, for comparing the intensities, we use the following Gaussian weight:

$$w(u,v) = \frac{1}{2\pi\sigma^2}\exp(-\frac{\triangle x(u)^2 + \triangle y(u)^2}{2\sigma^2}) \tag{6.13}$$

where σ is chosen such that $w(u,v) \approx \frac{1}{2}$ for $(u,v) \in \{(0,n),(0,-n),(n,0),(-n,0)\}$ (cf. Section 6.7.1). Figure 6.2 illustrates the shape of our grid for the case of n=7.

As above (cf. Section 6.2.1), we actually consider a change of reference frames of the pixel coordinates in I and I' such that $(\mathbf{x}_x, \mathbf{x}_y) = (\mathbf{x}'_x, \mathbf{x}'_y) = (0,0)$ in the new frames. Expressi

Given the change of reference frames, the dissimilarity η we minimize at each iteration can be expressed as:

$$\eta(A) = \sum_{\substack{(x,y)\in G_F \\ (x',y')=A(x,y)}} \frac{1}{2\pi\sigma^2}\exp(-\frac{x^2+y^2}{2\sigma^2})\left[r_s I'(x',y') + r_t - I(x,y)\right]^2 \tag{6.14}$$

As above, we look for a small displacement (dx', dy') of (x', y') that minimizes the dissimilarity $\eta(A)$, i.e., a small change dA of the affinity A that would induce such a displacement (dx', dy'). We similarly obtain a convex system for dA that is a variant of Equation (6.10), with an added weight and a different grid support:

$$\begin{aligned}\arg\min_{dA} \sum_{\substack{(x,y)\in G_F \\ (x',y')=A(x,y)}} \frac{1}{2\pi\sigma^2}\exp(-\frac{x^2+y^2}{2\sigma^2}) \big[r_s \big(I'(x',y') \\ + g'_x(x',y')\,(da_{11}\,x + da_{12}\,y + da_{13}) \\ + g'_y(x',y')\,(da_{21}\,x + da_{22}\,y + da_{23})\big) \\ + r_t - I(x,y)\big]^2\end{aligned} \tag{6.15}$$

Also as above, the algorithm iterates as long as the dissimilarity $\eta(A)$ decreases with a newly interpreted A, or until a fixed maximum number of iterations is reached. As the optical flow constraint is only a first order approximation, and as we can be initially far from the optimal solution, we can fall into a local minimum from which we cannot escape. To reduce the impact of such cases, if $A + dA$ does not decrease η, we heuristically try successive smaller modifications $A + \frac{1}{2}dA$ and $A + \frac{1}{4}dA$ to possibly update A. Otherwise, the LSFM algorithm explores the next (finer) image scale (see following Section 6.4.3), or stops if it is the last (finest) scale. The pseudo-code for the LSFM iterative optimization at a single scale is shown in Algorithm 1.

Input: I, I': images at a given scale
A_{init}: initial transformation matrix
Result: Refined transformation A

$A \leftarrow A_{init}$
$\eta_{best} \leftarrow \eta(A)$, according to Eq. (6.14)
repeat
  compute dA by Eq. (6.15)
  $z \leftarrow 1$
  for $i \leftarrow 0$ **to** 2 **do**
    $\eta_{new} \leftarrow \eta(A + z\,dA)$
    if $\eta_{new} < \eta_{best}$ **then**
      $A_{best} \leftarrow A + z\,dA$
      break for
    else
      $z \leftarrow z/2$
    end
  end
until *A does not change*;
return A

Algorithm 1: Pseudo-code for the LSFM iterative optimization at a given scale

6.4.3 Scale exploration

To provide an additionnal robustness, rather than directly adjusting the feature positions at their original scale, we perform a coarse-to-fine refinement. For this, we explore two pyramids of scaled images, for I and I', similar to the pyramids used in SIFT detection. These pyramids are independant of the matches; they are created once for a given pair of images, and explored for each match, possibly differently.

Initialization. Given a feature match in I, I', for every precomputed image scale we initialize a grid G_F and estimate a corresponding initial affinity A_{init} based on the relative position and orientation of the two considered features. We measure the initial dissimilarities $\eta(A_{init})$ at each of the scales and choose as the starting scale the one that yields the smallest $\eta(A_{init})$.

Coarse-to-fine process. We begin adjusting the point location in I' at this starting scale, and progressively refine the location by reducing the scale until we reach the original image scale. Note that we do not search here an optimal scale as in [40]; for our problem, the original scale is the best one regarding the final accuracy.

More precisely, after convergence to an affinity A_{up} at at a given scale s_{up}, we restart the affinity refinement at the scale below s_{low} using as initial estimate A_{init} the corresponding transformed (scaled) affinity $downscaled(A_{up})$. As a high blur may cause deviation from the optimal solution, we make sure there is an actual improvement when we restart the refinement process at the lower scale. Concretely, at the lower scale s_{low}, if the measure of dissimilarity η computed with the inherited scaled affinity $downscaled(A_{up})$ is larger than the dissimilarity computed using the affinity A_{low} defined by just the relative position and orientation of the matched features at this scale, then $A_{init} = A_{low}$ is used as the initial estimate instead of $downscaled(A_{up})$.

In practice, in our experiments, we explore 5 octaves of scale, with a geometric progression of ratio $\sqrt{2}$. For better understanding of the scale exploration, please refer to Algorithm 2.

This coarse-to-fine strategy improves robustness, preventing some affinity refinements to be caught in a poor local minimum if it were performed directly at the original scale, and thus leading to a better average accuracy.

The complete LSFM method with scale exploration is described with pseudo-code by Algorithm 2, based on the above definitions.

Input: $(I_j)_{0\leq j\leq S}$ and $(I'_j)_{0\leq j\leq S}$: pyramids of images at different scales,
with $I_0 = I, I'_0 = I'$
$m = (\mathbf{x}, \mathbf{x}')$: match to refine
Result: Affinity A, to correct the position of $\mathbf{x}'$ as $A(\mathbf{x})$

for $j \leftarrow S$ **to** 0 **do**
  $\hat{A}_j \leftarrow$ affinity defined by relative position and orientation of $\mathbf{x}, \mathbf{x}'$ at scale j
end
$j* \leftarrow \arg\min_{0\leq j\leq S} \eta(\hat{A}_j)$, with η as defined by Equation (6.14)
$A_{init} \leftarrow \hat{A}_{j^*}$
for $j \leftarrow (j^* - 1)$ **to** 0 **do**
  $A_j \leftarrow$ affinity computed from initial estimate A_{init} at scale j by Algorithm 1
  if $j > 0$ *and* $\eta(downscaled(A_j)) < \eta(\hat{A}_{j-1})$ **then**
    $A_{init} \leftarrow downscaled(A_j)$
  else
    $A_{init} \leftarrow \hat{A}_{j-1}$
  end
end
return A_{init}

Algorithm 2: Pseudo-code for LSFM with scale exploration

6.5 Parameters

In our experiments, we used a spline interpretation of order 5 to get sub-pixel intensity. We set $n = 7$ as a trade-off between accuracy and computation time. For the grid focus, we used $\rho = 1.1$ in order to make the samples at the grid center roughly twice as dense (in one direction) as at the border, since $1.1^n = 1.95 \approx 2$.

The σ and λ parameters are related to region size; here we decided first to set the Gaussian weight to $\sigma = 0.9\lambda\frac{\rho^n-1}{\rho-1}$ depending on the scale factor λ. Then $\lambda = 1.57$ via experiments described in Section 6.7.1.

6.6 Impact of the kind of feature detector

LSM-like refinement is more accurate for regions with high gradients. SIFT does not necessarily detect points in such regions, but its robustness under various conditions (re-scaling, view-point change, light condition change, etc.) and its generally large number of detections and matches compensate. Besides SIFT tends to find points within objects, where relative intensity is more stable, compared to corners that have strong but less reliable gradients because they often correspond to occlusion edges.

6.7 Experiments

We compare our focused matching (LSFM, Section 6.4) with standard least square matching (LSM, Section 6.2). Rather than just considering planar scenes and measuring reprojection errors, we directly measure the impact in terms of camera rotation and translation error e_R, e_t after calibration.

SIFT features are first detected and matched. The correspondences are cleaned by the K-VLD filter (cf. Chapter 4). They are then refined by a specific method, LSM or LSFM. Finally, ORSA+IRLS uses the refined matches to recover the camera position. The ORSA algorithm [57] is a state-of-the-art RANSAC variant, suited for accuracy (as opposed, e.g., to robustness or speed); IRLS tries to minimize the sum of squares of geometric errors between points in the right image and the epipolar line of their corresponding points in the left image. For the evaluation, we compare e_R and e_t to the ground truth. In order to produce reliable measures, all the results we provide are averaged over 20 runs.

Datasets. Only datasets with highly accurate ground-truth calibration can be used for validation. We experimented with the full dataset of Strecha et al. [73], a de facto standard in camera calibration: 6 groups of 8 to 30 images totaling 95 pairs of successive images. For each pair, SIFT feature points are detected and matched with the usual setting [49] (no tweaking as in Sect. 7.2.2), i.e., a ratio of descriptor distance to next best match at most 0.8. We ran the same experiment with the DTU robot dataset [1]. However, as it is huge (about 0.5 To), we only considered 9 of the 60 groups of images, covering various themes (scenes 1, 2, 4, 9, 10, 12, 21, 28, 52), in the reduced format (fewer images, yielding 12 images pairs: 1-12, 12-24, 24-25, 25-26, 26-37, 37-49, 50-57, 57-64, 57-65, 57-94, 64-95, 64-119), with identical illumination condition (number 08 for all tests), but full-size images.

6.7.1 Covering factor evaluation

The first set of experiments is designed to estimate optimal coverage zones (cf. Section 6.3) for LSM and LSFM methods when $n = 7$. By varying the covering factor λ, LSM and LSFM may perform differently. We test both with different covering factors in a fixed range, and compare them under their best setting. Note, when we vary the covering factor, we also modify the parameter σ of LSFM since σ depends on λ (cf. Section 6.4.2).

We test both with $\lambda = \frac{k}{2n+1} \in \{1.00, 1.29, 1.57, 1.86, 2.14, 2.43\}$, which corresponds to coverage zones for LSM of 14×14, 18×18, 22×22, 26×26, 30×30 and 34×34 pixel square.

Detailed results are provided in Table 6.1. For LSM, we obtained the best results on average with $\lambda = 1.86$. For LSFM, the best results were obtained for $\lambda = 1.57$. These are the (different) values we kept for the two methods in all the following experiments. Note that the size of the coverage area for the regular LSM grid is $2\lambda n$ while the size of the focused grid is $2\lambda(\rho^n - 1)/(\rho - 1)$. Although the covering factor of LSM is larger than the one for LSFM, the corresponding grid is smaller.

6.7.2 LSM vs. LSFM

In the second set of experiments, matches are refined by the following methods: LSM, LSFM without scale exploration (i.e., LSM with focused grid), and full LFSM (with both focused grid and scale exploration). Methods are tested using their corresponding best

Strecha et al. [73]	Coverage zone length k (Covering factor λ)					
LSM	14 (1.00)	18 (1.29)	22 (1.57)	26 (1.86)	30 (2.14)	34 (2.43)
e_R (deg $\times 10^{-2}$)	9.04	8.79	8.21	7.86	7.56	7.76
e_t (deg)	1.02	0.97	0.91	0.82	0.79	0.83
LSFM	19 (1.00)	24.5 (1.29)	29.6 (1.57)	35.3(1.86)	40.6 (2.14)	46.1 (2.43)
e_R (deg $\times 10^{-2}$)	7.00	6.24	5.86	6.16	5.71	5.88
e_t (deg)	0.73	0.64	0.59	0.60	0.55	0.58
DTU robot [1]	Coverage zone length k (Covering factor λ)					
LSM	14 (1.00)	18 (1.29)	22 (1.57)	26 (1.86)	30 (2.14)	34 (2.43)
e_R (deg $\times 10^{-2}$)	21.52	21.27	20.90	21.16	21.10	21.31
e_t (deg)	0.89	0.85	0.83	0.88	0.91	0.89
LSFM	19 (1.00)	24.5 (1.29)	29.6 (1.57)	35.3(1.86)	40.6 (2.14)	46.1 (2.43)
e_R (deg $\times 10^{-2}$)	20.67	20.52	20.69	20.69	20.30	20.78
e_t (deg)	0.70	0.65	0.65	0.67	0.68	0.73

Table 6.1: Match refinement evaluation for LSM and LSFM with various values of the covering factor λ. The different values of λ correspond different values of the length k of the coverage zone, in $\{14, 18, 22, 26, 30, 34\}$. The best λ for LSM is in red, and the best λ for LSFM is in green.

e_R (deg $\times 10^{-2}$)	LSM	LSM + foc. grid	LSFM	gain
Strecha et al. [73]	7.56	6.73	5.86	1.29
DTU robot [1]	21.05	20.83	20.68	1.01
e_t (deg)	LSM	LSM + foc. grid	LSFM	gain
Strecha et al. [73]	0.79	0.72	0.59	1.33
DTU robot [1]	0.83	0.69	0.65	1.28

Table 6.2: Match refinement evaluation using LSM, LSM with focused grid and LSM with focused grid and scale exploration (LSFM). The gain factor measures the improvement of LSFM over LSM.

covering parameters given by previous experiments (cf. Section 6.7.1). This allows us to see the contribution of the different components of LSFM.

For LSFM, we use again a focused grid with $n = 7$, $\lambda = 1.57$, $\rho = 1.1$ and $\sigma = 0.9\lambda\frac{\rho^{|n|}-1}{\rho-1}$. We explore 5 octaves of scale, dividing each octave in two, i.e., with a geometric progression of ratio $\sqrt{2}$.

Table 6.2 shows that, apart from a poor reduction of rotation error in the DTU robot dataset, LSFM has a gain factor from 1.28 to 1.33 in accuracy. It also show that both the focused grid and the scale exploration contribute to a better accuracy.

6.8 Extension: Surface normal estimation

6.8.1 Introduction

Suppose that for a match $m = (\mathbf{x}, \mathbf{x}')$, the small regions around two blob-like features $\mathbf{x}$ and $\mathbf{x}'$ are locally planar. Then the two regions around features at $\mathbf{x}$ and $\mathbf{x}'$ are locally related by a homography H. According to (3.8) (see also [34, §13, Eq.(13.2)]), the

homography matrix can be expressed as:

$$H = c\mathbf{K}'(R - tn^T/d)\mathbf{K}^{-1}, \text{ for some } c \in \mathbb{R}^* \tag{6.16}$$

where the vector n represents the surface normal, satisfying $n^T\mathbf{X} + d = 0$ for all points on this plane. If the camera intrinsic parameter $\mathbf{K}$ and $\mathbf{K}'$ are known and if the transformation H and camera position can be estimated, we can also estimate the surface normal n for the features.

We have tried to refine the match location accuracy by estimating the local planar perspective transformation (homography H), but the results did not show significant improvements. On the contrary, the use of H leads to slower convergence and more local minima. This is why, in LSFM, we choose to only estimate the affine approximation of the transformation.

Still, in this section, we present a method to recover the missing part of the homography transformation by an iterative affine estimation method, which can estimate local surface normals over refined matches when the viewpoint change is not too large.

6.8.2 Homography Recovery

We consider that the affine transformation A is a first order approximation of the perspective transformation H. For a given homography matrix, we rewrite (6.16) as

$$\mathbf{K}'^{-1}H\mathbf{K} = Rc - t(\frac{c}{d}n^T). \tag{6.17}$$

Equation (6.17) is better conditioned than (6.16). We note $W^T = (\frac{c}{d}n^T) = [w_0, w_1, w_2]$. With estimated R and t, (6.17) becomes an overdetermined linear system in variables (c, w_0, w_1, w_2). Under a moderate view-point change, the estimated affine transformation is not too far from the perspective transformation. The surface normal is then given by W after normalization.

In order to better estimate n, we recover the homography H via (6.16), and refine the match from the initial transformation $\tilde{H}$ with H_{31}, H_{32} fixed. In our experiment, R and t are not re-estimated after each iteration, since we did not observe any impact on the accuracy for R and t with re-estimation. Experimentally, three iterations of the refinement-estimation process are more than enough to estimate n; more iterations do not yield a significant improvement. The algorithm for the surface normal estimation is described by Algorithm 3.

6.8.3 Normal illustration

Lacking quantitative data for surface normal evaluation, we just illustrate a few qualitative results. In Figure 6.3, we present a pair of images from Strecha et al. [73] and reconstruct the surface normals at match locations.

Normals are reasonnably well reconstructed with or without homography-recovering iterations. However, results are slightly better with homography-recovering iterations: normals are more perpendicular to the wall with our algorithm than with the direct result from match refinement.

6.9 Conclusion

In this chapter, we have proposed an extension of the least square matching method to refine match locations. According to experiments involving real data with ground-

Input : I and I': given images
$\mathbf{K}'$ and $\mathbf{K}'$: intrinsic parameter matrix
R and t: estimated camera rotation matrix and translation vector
$m = (\mathbf{x}, \mathbf{x}')$: match where to estimate the normal
Output: the surface normal vector n

initialize H by feature rotation, re-scaling and translation;
for $i \leftarrow 0$ **to** 2 **do**
  perform LSFM to refine H with fixed H_{31} and H_{32} (cf. Algorithm 2);
  solve in the least square sense the overdetermined linear system in (c, W):
  $Rc - tW^T = \mathbf{K}'^{-1} H \mathbf{K}$;
  compute $H \leftarrow \mathbf{K}'(Rc - tW^T)\mathbf{K}^{-1}$ (update H_{31} and H_{32});
end
$n \leftarrow \frac{W}{|W|}$;
return n;

Algorithm 3: Pseudo-code for estimating the normal for matched features

truth calibration, our extended match refinement leads to a significant reduction of SfM errors.

This method is valuable for two-view stereovision. However, extending it to the multiple-view case is not trivial because of internal track consistency: the location of points in a track would need to be optimized simultaneously in all associated images. This would involve a more complex *track refinement*.

We will see in the next chapter another method to improve two-view SfM accuracy (match selection), which can be combined with match refinement to achieve even better accuracy.

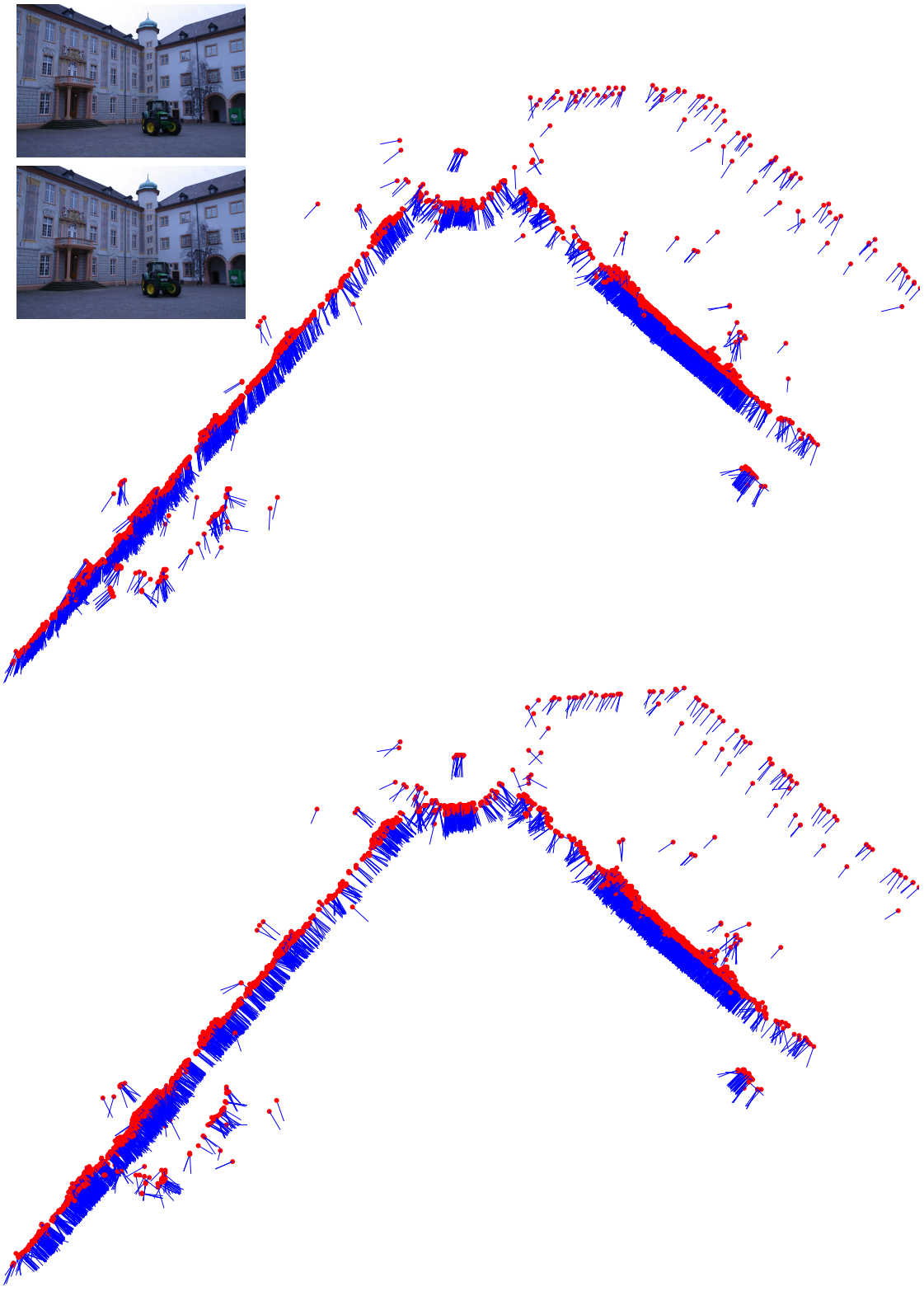


Figure 6.3: Surface normal estimation (in blue) for feature points reconstructed in 3D (in red) by SfM from the displayed image pair. Views from above; points behind the walls correspond to features on the roof. Top: estimated normals from match refinement without homography-recovering iterations. Bottom: estimated normals with homography-recovering iterations. We can observe that normals are more perpendicular to the wall with our algorithm than with the direct result from match refinement. The difference is observable in particular in the left bottom area of the scene.

Chapter 7

Match selection for highly accurate two-view SfM

In this chapter, we propose a method to greatly enhance the accuracy of two-view structure from motion (SfM), which is independent from that of Chapter 6. We first answer the question: "fewer data with higher accuracy, or more data with less accuracy?" For this, we establish a relation between SfM errors and a function of the number of matches and their epipolar errors. Using an accuracy estimator of individual matches, we then propose a method to select a subset of matches that has a good quality vs. quantity compromise. It can be combined with match refinement techniques of Chapter 6, where both selection and refinement contribute independently to a better accuracy.

Contents

7.1 Introduction . . . 117
7.2 Statistical behavior of SfM errors . . . 117
7.2.1 Theoretical results . . . 117
7.2.2 Empirical Results . . . 118
7.2.3 Realistic, semi-synthetic dataset . . . 119
7.2.4 Analysis . . . 119
7.3 Match Selection to Improve Accuracy . . . 123
7.3.1 Cleaning up input matches . . . 123
7.3.2 Comparing subsets of matches . . . 123
7.3.3 Exploring subsets of matches . . . 124
7.4 Ranking matches . . . 125
7.4.1 SIFT ranking function . . . 125
7.5 Algorithm . . . 126
7.5.1 Match selection alone (without match refinement) . . . 126
7.5.2 Match selection with match refinement . . . 127
7.5.3 Comparison to related methods . . . 128
7.6 Experiments . . . 128
7.6.1 Match selection and refinement . . . 129
7.7 Visual illustration of 3D reconstruction accuracy . . . 130
7.8 Number of matches kept by match selection . . . 134
7.9 Various possible bias in alternative algorithms . . . 134
7.9.1 Cleaning up matches with RANSAC before selection is biased . 134
7.9.2 Distance to the epipolar line is biased for ranking matches . . . 135

7.10 Conclusion 136
7.11 Annex 137
7.11.1 Regression correlation coefficient 137
7.11.2 Maximum crushing expression derivation 137

7.1 Introduction

As feature detection and matching is not perfect, two main hurdles can disturb the SfM process: the matches can be either incorrect or inaccurate. There is actually a grey area between these notions: incorrect matches reduce SfM accuracy, sometimes to the point of making it fail, while inaccurate matches are considered as good enough for calibrating, even though they also degrade SfM accuracy. Incorrect matches are generally dealt with using RANSAC [27] or one of its numerous variants [20, 21, 57, 77]. It separates inliers (supporting matches) from outliers (conflicting matches), trying to find the largest consensus on an estimated fundamental matrix, using a threshold (fixed or adaptive) to assess consistency. While this robust selection method can eliminate many mismatches, a number of false positives can remain among the selected inliers because the rejection criterion is based on the distance to epipolar lines, which provides a necessary but not sufficient condition (because of ambiguity along epipolar lines).

Compromises at two different levels impact SfM accuracy. First of all, statistically, the more matches to calculate the fundamental matrix, the more accurate the estimation. A first compromise thus concerns the RANSAC selection criterion: if it is too permissive, matches considered as inliers are more numerous but are also more likely to be contaminated by wrong matches, and accuracy drops; if the criterion is too strict, there are too few inliers to get a good accuracy. The second compromise concerns the accuracy heterogeneity of individual inliers: keeping only the most accurate inliers can naturally improve SfM accuracy; but it can also degrade it as the estimation is based on fewer points. The first compromise has indirectly been widely studied: people try to select as many good matches as possible, while excluding as many wrong matches as possible. But the second compromise, quality vs. quantity, has been poorly addressed. This chapter presents an original method to find a good balance between the number of inliers to consider for SfM estimation and their expected accuracy.

Organization of this chapter

We establish an empirical statistical relationship between the inaccuracy of matches, their number, and various indicators of SfM inaccuracy (Section 7.2). We describe an original method that exploits this relationship to select matches that are likely to improve SfM accuracy (Section 7.3). Section 7.4 address to the ranking function used in our algorithm to select better subsets of matches. Section 7.5 presents two variants of our match selection algorithm, one for independent use and one to be used in combination with match refinement. We show in Section 7.6 that this method by itself improves substantially the accuracy of structure from motion, an effect that is amplified when combined with match refinement. We try to illustrate the impact our algorithm can have in point cloud reconstruction in Section 7.7 and measure the size of selected subsets of matches in Section 7.8. Finally, a discussion about the possible alternative algorithms is give in Section 7.9.

7.2 Statistical behavior of SfM errors

7.2.1 Theoretical results

We consider a pair of images I, I', obtained by cameras C, C' with 3×4 projection matrices P, P' and 3×3 calibration matrices K, K'. We also consider a set of matches M between I and I', i.e., pairs of points $m = (\mathbf{x}, \mathbf{x}')$ where $\mathbf{x}$ is the projection of a 3D point $\mathbf{X}$ on I, i.e., $\mathbf{x} = P\mathbf{X}$ in homogeneous coordinates, and where $\mathbf{x}'$ is a point in I' considered

as matching with $\mathbf{x}$, possibly with some inaccuracy. In the general case, a fundamental matrix F_M between I and I' can be estimated from matches M, and F_M may in turn be used with K, K' to estimate projection matrices P_M, P'_M on I, I'. The resulting reprojection error of $\mathbf{X}$ in I', i.e., the discrepancy in I' between the exact reprojection of $\mathbf{X}$ by P' and the estimated reprojection of $\mathbf{X}$ by P'_M is the pixel distance $e_{2D}(M,m) = d(P'\mathbf{X}, P'_M\mathbf{X})$.

In case images I and I' are related by a homography H, and considering matching points $\mathbf{x}'$ as possibly inaccurate measurements of reprojected points $P'\mathbf{X} = H\mathbf{x}$ in I', Hartley and Zisserman [34, §5.1.3, Eq.(5.5)] show that, if these measurements are subject to independent Gaussian noise with standard deviation $\sigma_{2D}(M)$, then the estimation error $e_{2D}(M)$ of reprojected points in I' by the estimated homography H_M, or equivalently via P'_M, is:

$$e_{2D}(M) = \mathbb{E}_{m\in M}[e_{2D}(M,m)^2/|M|]^{1/2} = 2\sigma_{2D}(M)/\sqrt{|M|}. \tag{7.1}$$

Dividing the estimation error by 2 thus requires 4 times as many matches, or matches with location error divided by 2. This bound is optimal (assuming no other errors such as distortion), and achieved for the Maximum Likelihood Estimator (MLE). Finding a similar bound for the fundamental matrix is impractical because it is a non convex problem in very high dimension. We do not try to solve it, but we draw inspiration of the MLE bound in what follows.

Another reading of (7.1) is that if we can find a subset $M_{\text{sub}} \subset M$ such that matching points $\mathbf{x}'$ in M_{sub} are subject to independent Gaussian noise with standard deviation $\sigma_{2D}(M_{\text{sub}}) < \sigma_{2D}(M)$ compared to their expected location $H\mathbf{x}$, and if $\sigma_{2D}(M_{\text{sub}})/\sqrt{|M_{\text{sub}}|} < \sigma_{2D}(M)/\sqrt{|M|}$, then $e_{M_{\text{sub}}} < e_M$, and $H_{M_{\text{sub}}}$ is thus a better estimate of H than H_M. Now if we have a way to evaluate $\sigma_{2D}(M_{\text{sub}})$ for any M_{sub}, the optimal subset M^*_{sub} of matches for estimating H is:

$$M^*_{\text{sub}} = \underset{M_{\text{sub}}\subset M}{\arg\min}\ \sigma_{2D}(M_{\text{sub}})/\sqrt{|M_{\text{sub}}|}. \tag{7.2}$$

$H_{M^*_{\text{sub}}}$ minimizes reprojection errors w.r.t. ground truth H.

To our knowledge, a similar result is not known for the fundamental matrix. The situation is more complex in this case as estimating F, with 7- or 8-point methods, relies on singular value decomposition (SVD) and/or requires solving complex polynomial systems.

7.2.2 Empirical Results

As a theoretical result is difficult to obtain, we study empirically the influence of $|M|$ and $\sigma_{2D}(M)$ on SfM accuracy. Using a collection of images with accurate ground-truth calibration, presenting various feature distributions, we measure the following:

- F_M is the fundamental matrix estimated from M using ORSA (a RANSAC variant) [57] and iterative re-weighted least squares (IRLS) (see Section 3.7.3).
- $e_F(M)$ is the root mean square error (RMSE) of the distance $e_F(M,m)$ of $\mathbf{x}'$ to the F_M-epipolar line of $\mathbf{x}$ in I', for all $m{=}(\mathbf{x},\mathbf{x}')\in M$.
- $e_R(M) = \angle R_{\text{gt}} R_M^{-1}$ is the angle between the ground-truth rotation R_{gt} and its estimate R_M based on M.
- $e_t(M) = \angle(t_{\text{gt}}, t_M)$ is the angle between the ground-truth translation direction t_{gt} and its estimate t_M.
- $e_{3D}(M,R,t)$ is the RMSE of the distance of the 3D point $\widehat{\mathbf{X}}$ triangulated from $\mathbf{x},\mathbf{x}'$ using a given rotation and translation R,t, to the ground-truth 3D point $\mathbf{X}$, for all $m{=}(\mathbf{x},\mathbf{x}')\in M$. We also define $e_{3D}(M) = e_{3D}(M,R_M,t_M)$.

7.2.3 Realistic, semi-synthetic dataset

Estimating SfM errors requires a ground truth for both calibration and matched points. While accurate camera calibrations can be determined using LiDAR data [73], it is difficult to construct a significant number of accurate ground-truth point matches. For this, we resort to semi-synthetic ground-truth datasets: the images, the camera poses and the distribution of matching points are real, but the actual point locations are adjusted to make sure they are error-free.

Concretely, given a pair of images I, I' with known calibration P_{gt}, P'_{gt}, we detect and match SIFT feature points in each image. We use a descriptor distance ratio to next best match at most 0.8, which is the standard setting [49]. As these matches may still contain mismatches, we first clean them using the K-VLD method(cf. Chapter 4, that eliminates many false matches, including near the epipolar lines, and then using ORSA, an adaptive state-of-the-art variant of RANSAC by Moisan and Stival [57], known for its robustness in practical SfM systems [61]. It results in an almost mismatch-free set of matches $\tilde{M}$. Treating them as inliers, for each match $\tilde{m}=(\tilde{\mathbf{x}},\tilde{\mathbf{x}}')\in\tilde{M}$, we construct an estimated 3D point $\mathbf{X}$ by triangulation using ground truth calibration P_{gt}, P'_{gt}, and reproject it onto images I, I' as new 2D points $(\mathbf{x},\mathbf{x}')=m$. The resulting set of matches M_{gt} yields a perfect ground truth that is realistic in terms of feature distribution in images and in space.(location and number)

We then add noise by randomly moving in image I' the matched points $\mathbf{x}'$, using an isotropic Gaussian distribution with given standard deviation σ_{2D}. This asymmetric setting reproduces the theoretical hypothesis mentioned in Section 7.2.1. (Adding noise to points in both images experimentally leads to almost identical results, scaled by a constant factor.) To also conform to this hypothesis, the noise is independent of the characteristics of the features that originated the synthetic points, such as scale. Moreover, we add variation to the number of matches by randomly selecting only a given ratio r. This defines new sets of matches $M=M(\sigma_{2D},r)$.

In our experiments, we use Strecha et al.'s dataset [73]. It consists of 6 groups of 8 to 30 images with both internal and external accurate ground-truth calibration. We consider all pairs of consecutive images in all image groups, in which we detect and match SIFT features. The number of matches typically varies between 300 and 6000. For each image pair, we consider discrete ratios of matches $r=k^2/100$ with $k=4,\dots,10$ (thus different variant of point configurations), and standard deviation $\sigma_{2D}=0.2+0.3k$ with $k=0,\dots,6$ (in pixels). For each combination of r and σ_{2D}, we sample 50 noisy variants of the data, estimate their SfM accuracy, and average the corresponding error measures by quadratic mean (RMS).

7.2.4 Analysis

Adding noise σ_{2D} and ratio r as $M=M(\sigma_{2D},r)$ has a direct impact on epipolar error $e_F(M)$ and on rotation and translation errors e_R, e_t. Two observations are important and guide the following work.

Error source decomposition

Noise σ_{2D} has an impact on the error e_{3D} of estimated 3D points in two ways. First, it directly introduces error due to noise in matches, which linearly increases with σ_{2D}. Second, camera motion estimation R_M, t_M from noisy matches generates a systematic indirect error due to calibration, see Figure 7.1.

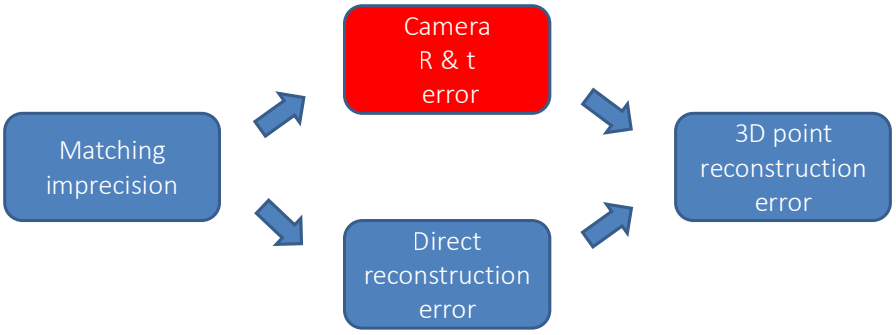


Figure 7.1: Reconstruction error composition due to σ_{2D}

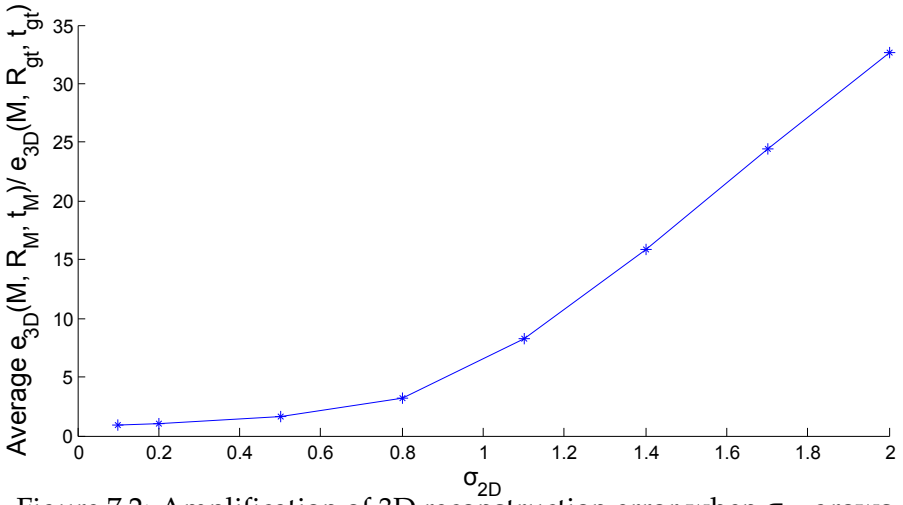


Figure 7.2: Amplification of 3D reconstruction error when σ_{2D} grows

Figure 7.2 illustrates the ratio between the combined reconstruction error (direct reconstruction error plus indirect camera R_M, t_M errors) and the pure direct reconstruction error (camera position set to ground truth value R_{gt}, t_{gt}). With the increase of σ_{2D}, the calibration errors quickly amplify the 3D reconstruction errors (average on image pairs). This observation shows that the camera calibration error is the main source of 3D reconstruction error, and reducing errors in R_M, t_M is a promising way to improve reconstruction accuracy.

Behavior of reconstruction errors e_{3D}, e_R and e_t

Experimentally, we observe in Figure 7.3 that e_{3D}, e_R and e_t all are highly correlated to $N = |M|$ and σ_{2D}: despite some variations, we notice that the average values of $\log e_{3D}$, $\log e_R$ and $\log e_t$ are more or less linear with respect to $\log N$ when σ_{2D} is fixed, with some slope α depending on the image pair, and more or less linear with $\log \sigma_{2D}$ when N is fixed (but not the configuration as when $N < |M|$, we draw N random matches among the $|M|$ matches.) with some slope $-\beta$ also depending on the image pair. It is confirmed by computing the regression correlation coefficient (RCC, see Section 7.11.1) of e_{3D}, e_R, e_t with $\sigma_{2D}^{\alpha}/N^{\beta}$, which is in general very close to 1, as can be seen in Figure 7.4 (bottom 3 curves, plotted on the same diagram).

Besides, we also found empirically that σ_{2D} is more or less proportional to $e_F(M)$, not only to the exact epipolar error (i.e., w.r.t. the ground-truth fundamental matrix).

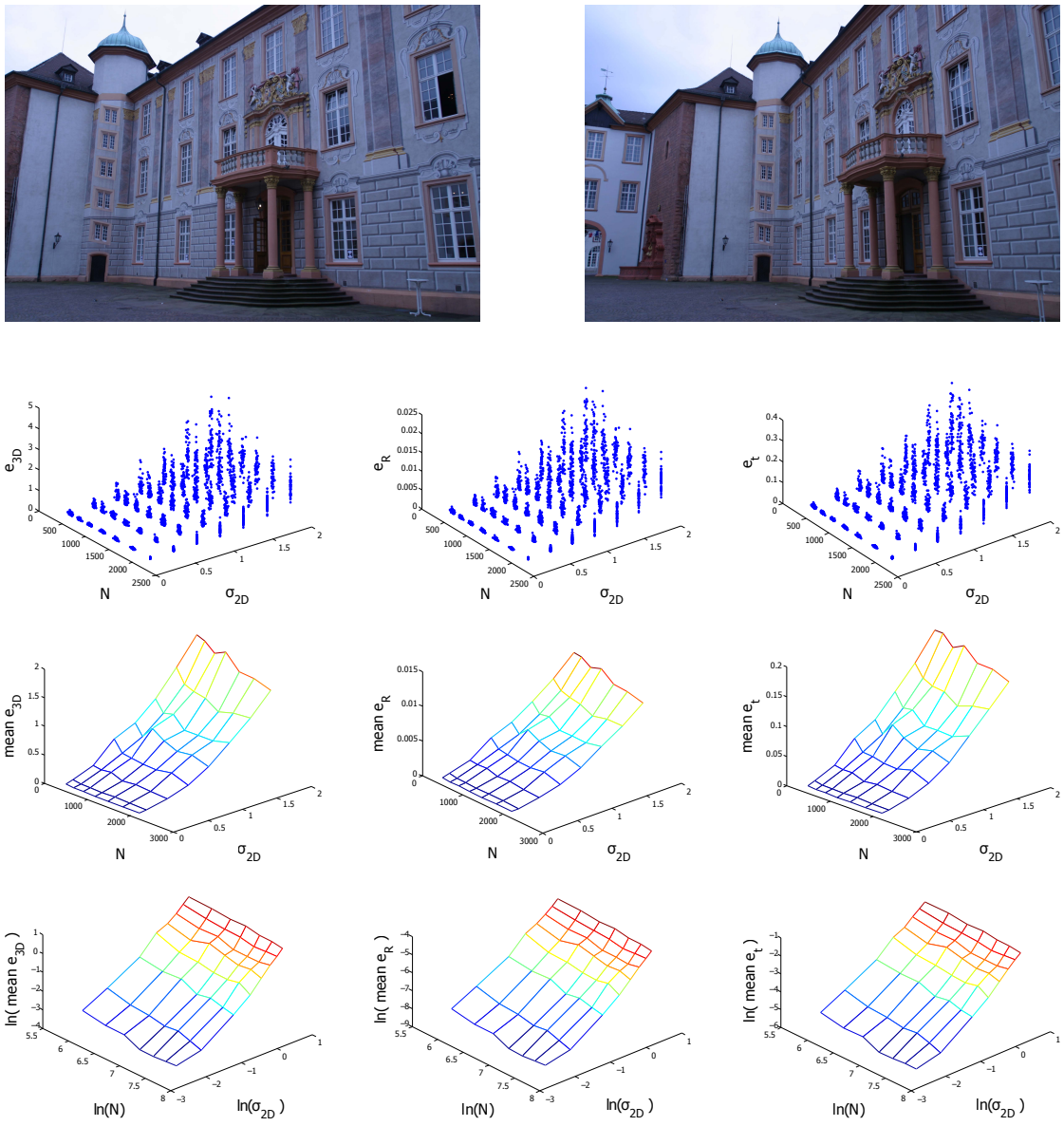


Figure 7.3: Behavior of e_{3D}, e_R and e_t in a pair of images.
Row 1: the pair of images.
Row 2: error distribution w.r.t. different N and σ_{2D}.
Row 3: average error w.r.t. different N and σ_{2D}.
Row 4: ln of average error w.r.t. different $\ln N$ and $\ln \sigma_{2D}$.
From left to right, the error respectively represents e_{3D}, e_R and e_t.

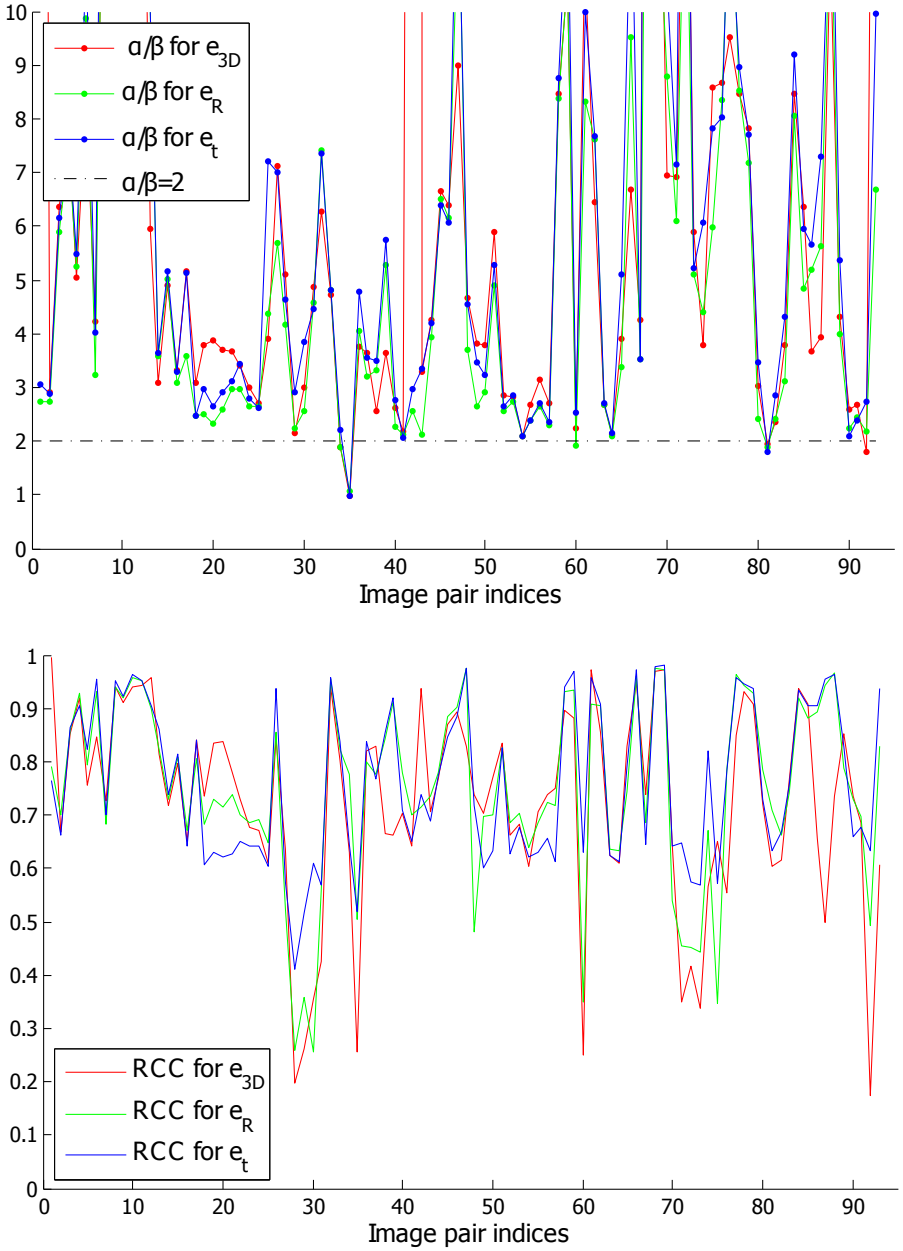


Figure 7.4: Top: curves with dots (bullets): estimated α/β for different image pairs (the order is irrelevant). Having a large α/β means that varying the match number has an increased influence on error.
Down: regression correlation coefficient between average e_R, e_t, or e_{3D}, and $\sigma_{2D}^{\alpha}/N^{\beta}$.
- mean RCC for e_{3D}: 0.72.
- mean RCC for e_R: 0.75.
- mean RCC for e_t: 0.76.

We thus hypothesize the relation:

$$e_R, e_t, e_{3D} \propto \frac{\sigma_{2D}^{\alpha}}{N^{\beta}} \propto \frac{e_F^{\alpha}}{N^{\beta}}. \tag{7.3}$$

With a fixed distribution of points, we should have $\alpha = 1$ for small errors. Still with a fixed configuration, but duplicating each match $N' = 2N$ (thus with the same distribution), the variance σ^2 of each estimated parameter is halved (cf. Section 3.8.1). We should thus have $N'^{\beta} = \sqrt{2}N^{\beta}$, hence $\beta = 0.5$. This is consistent with (7.1), however it does not hold when the point configuration varies. Experimentally, α and β can vary significantly depending on images pairs and match sampling. In our semi-synthetic dataset (cf. Section 7.2.3), β varies between 0.2 and 1.5. Yet, assuming relation (7.3), knowing α/β is sufficient to compare errors for a given image pair:

$$\frac{e_F^{\alpha}}{N^{\beta}} < \frac{e_F'^{\alpha}}{N'^{\beta}} \Leftrightarrow \frac{e_F^{\alpha/\beta}}{N} < \frac{e_F'^{\alpha/\beta}}{N'}. \tag{7.4}$$

The situation where all matched points are treated as inliers and contribute to estimating F amounts to preferring the largest N (smallest $1/N$) independently of e_F, i.e., corresponds to $\alpha/\beta = 0$. On the contrary, the larger α/β, the more aggressively low-accuracy features should be discarded. As can be seen in Figure 7.4,

$$\alpha/\beta \geq 2 \text{ almost consistently.} \tag{7.5}$$

7.3 Match Selection to Improve Accuracy

To improve accuracy, we estimate the SfM using a selected subset of good matches.

7.3.1 Cleaning up input matches

Although we use IRLS for estimating F, the level of accuracy we target may be sensitive to mismatches remaining after RANSAC. We thus try early to eliminate these mismatches from the set of input matches. We have to do it without introducing the bias of an early approximate calibration estimation, which would be the case if we were to first filter the matches using RANSAC. For this reason, we first clean up the matches using the K-VLD method (cf. Chapter 4). Based on semi-local geometric and photometric consistency, it eliminates many mismatches without any calibration assumption. Running ORSA afterwards for estimating F on the resulting set of matches M typically (on Strecha et al.'s dataset) only removes on the order of 10% of matches (instead of up to 90% without K-VLD) with an estimated threshold of less than 2-pixel error.

7.3.2 Comparing subsets of matches

The SfM errors we want to reduce are $e_R(M)$, $e_t(M)$, $e_{3D}(M)$, though only $e_F(M)$ can be easily measured given a pair of images and a set of matches M. However, as indicated by (7.3) and (7.4), $e_R(M)$, $e_t(M)$, $e_{3D}(M)$ vary monotonically with $e_F(M)^{\alpha/\beta}/|M|$. The basic idea of our match selection is to use only a subset of matches $M_{\text{sub}} \subset M$ as soon as:

$$\frac{e_F(M_{\text{sub}})^{\alpha/\beta}}{|M_{\text{sub}}|} < \frac{e_F(M)^{\alpha/\beta}}{|M|}. \tag{7.6}$$

However, α/β is a priori unknown for an arbitrary image pair. Moreover, we want to improve SfM without taking the risk to degrade it. What we need is a sufficient

condition that reducing the number of matches will probably improve accuracy but most certainly will not reduce it. For this, we look for a possible value $\gamma \geq 0$ such that, for any image pair, any set of matches M with corresponding α, β parameters, and any subset of matches $M_{\text{sub}} \subset M$,

$$\frac{e_F(M_{\text{sub}})^\gamma}{|M_{\text{sub}}|} < \frac{e_F(M)^\gamma}{|M|} \Rightarrow \frac{e_F(M_{\text{sub}})^{\alpha/\beta}}{|M_{\text{sub}}|} < \frac{e_F(M)^{\alpha/\beta}}{|M|} \tag{7.7}$$

Given such a γ we could then choose the following optimal subset of matches M^*_{sub} for estimating F:

$$M^*_{\text{sub}} = \underset{M_{\text{sub}} \subset M}{\arg\min} \frac{e_F(M_{\text{sub}})^\gamma}{|M_{\text{sub}}|} \tag{7.8}$$

The fundamental $F_{M^*_{\text{sub}}}$ minimizes reprojection errors w.r.t. ground truth F_{gt}.

Noting that $(e_F(M_{\text{sub}})/e_F(M))^\gamma < |M_{\text{sub}}|/|M| < 1$ and hypothesizing (7.5), we can choose $\gamma = 2$ because then $(e_F(M_{\text{sub}})/e_F(M))^{\alpha/\beta} < (e_F(M_{\text{sub}})/e_F(M))^\gamma$, ensuring condition (7.7). Note that parameter γ is chosen as a safe empirical lower bound, not as an average value, which is more robust. Still, a general method to treat a specific class of images would be to run experiments as in Section 7.2.2 and to pick a value $\gamma \leq \alpha/\beta$. Without loss of generally, we assume $\gamma = 2$ in the following.

7.3.3 Exploring subsets of matches

The difficulty to find M^*_{sub} is to explore $M_{\text{sub}} \subset M$, as there are too many such subsets ($2^{|M|}$). We propose to evaluate just a fraction of them, that has the most chances to lead to smaller ratios $e_F(M_{\text{sub}})^2/|M_{\text{sub}}|$. For this, we rank the matches in M and use this ordering to explore only subsets of top-rank matches. More precisely, we look for a ranking function $\phi : M \to \mathbb{R}$ to order the matches into a sequence $(m_i)_{1 \leq i \leq |M|}$ such that $i < j \Rightarrow \phi(m_i) \leq \phi(m_j)$, and consider $M_{\text{sub}}(N) = \{m_i \mid 1 \leq i \leq N\}$. If the ranking function ϕ is highly correlated to the reprojection errors $e_{2D}(M, m)$, and hence to the epipolar errors $e_F(M, m)$, then

$$\begin{aligned} \min_{M_{\text{sub}} \subset M} \frac{e_F(M_{\text{sub}})^2}{|M_{\text{sub}}|} &= \min_{N \leq |M|} \frac{1}{N} \min_{\substack{M_{\text{sub}} \subset M \\ |M_{\text{sub}}| = N}} e_F(M_{\text{sub}})^2 \\ &\approx \min_{N \leq |M|} \frac{1}{N} e_F(M_{\text{sub}}(N))^2 \end{aligned} \tag{7.9}$$

We may thus resort to:

$$N^* = \underset{N \leq |M|}{\arg\min} \frac{e_F(M_{\text{sub}}(N))^2}{N} \tag{7.10}$$

$$M^* = M_{\text{sub}}(N^*). \tag{7.11}$$

The number of subsets to explore is then reduced from $2^{|M|}$ to $|M|$, which is still a lot given that M may contain several thousands of matches. Note that $e_F(M_{\text{sub}}(N))^2/N$ is not necessarily a convex function of N. However, it is in practice "smooth" enough for a reduced exploration of $8 \leq N \leq |M|$ to make sense. In our various experiments, we found it both robust and accurate enough to preserve in M_{sub} a minimum of 40% of matches in M and to explore fractions of M with a 5% step, i.e., to consider $N = r|M|$ with ratio $r = 0.4 + 0.05k$ and $k = 0, \ldots, 12$.

7.4 Ranking matches

The choice of a ranking function ϕ varies with the kind of feature. For SIFT, it seems natural to consider the distance between feature descriptors $d(desc(\mathbf{x}), desc(\mathbf{x}'))$ as an indicator of feature accuracy. Besides, Tang [75, 76] showed that SIFT subsampling amplifies location error by the feature scale factor $scale(\mathbf{x})$. This leads us to define the following ranking function:

$$\phi(\mathbf{x}, \mathbf{x}') = \max(scale(\mathbf{x}), scale(\mathbf{x}'))\, d(desc(\mathbf{x}), desc(\mathbf{x}')). \tag{7.12}$$

Large features thus tend to be ordered last, unless their descriptors match well. Still, although they have a poor accuracy, they are often useful for robustness, which could be an issue if too many of them are discarded. But our use of K-VLD provides enough (if not better) robustness improvement to compensate.

Note that the definition of ϕ relies only on detection scale and on the SIFT *descriptor*, not on the detector. It can thus be used, e.g., for all detectors of Mikolajczyk et al. [54], including SURF, Harris-Affine and MSER. Transposition to other descriptors is direct, but the correlation coefficient should be checked.

7.4.1 SIFT ranking function

For SIFT, we have studied the behavior of matching position accuracy based on several parameters such as the difference of descriptor vectors, the scale, the salience of detection.

Dataset

We analyze these parameters using both real and synthetic images (actually transformations of real images). The images and transformation matrices are from Mikolajczyk's dataset [53]. We take the scenes "Bark", "Boat", "Graf" and "Wall" for analysis. For the real image set, we note the first image of each scene as the image I and image of index 2, 3, 4 as I' in each pair. For the synthetic image set, we note the first image of each scene as the image I and interpolate the image I' via the transformation matrix of image of index 2, 3, 4. Thus we prepared 12 real images pairs (not shown here) and 12 synthetic image pairs (Figure 7.5).

Match preparation

For each image pair, SIFT features are extracted and matched according to the nearest neighbor strategy with the Lowe threshold equal to 0.8. Matches with less than 2 pixels reprojection error are taken into consideration.

Correlation result

The correlation for real and synthetic image pairs is illustrated in Table 7.1. Note that the correlation value varies with the upper bound of the position accuracy for matches. We consider here accurate matches with reprojection error less than 2 pixels, which explains why the figures are slightly different from that of our published work [48].

We found a correlation coefficient of 0.34 between ϕ and σ_{2D} on real data (respectively 0.62 for synthetic data), which proves the relevance of ϕ for ordering M. It outperforms other indicators, such as feature saliency that has a correlation score of -0.05.

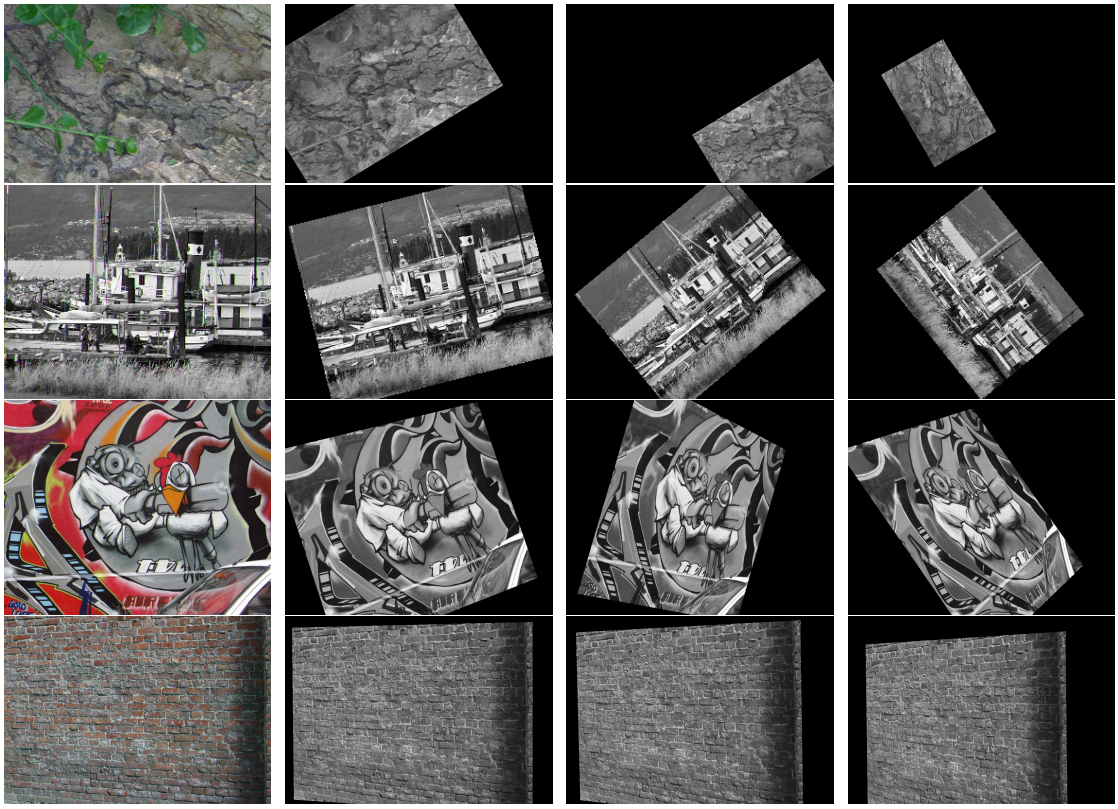


Figure 7.5: Images used to study the ranking function of SIFT features and matches. Apart from the first column, images are synthetic.

The Lowe score (ratio of descriptor distance to next best match) has an individual correlation of 0.18 for real image pairs (0.28 for synthetic image pairs), but it does not improve the global correlation when combined with ϕ. Figure 7.6 illustrates the fitting result as a function of ϕ. Notice that the maximum error has a tendency to increase with ϕ, but since most of the matches are in the region $\phi \leq 0.4$, we can rely on ϕ as a good indicator overall.

Parameter	Match accuracy (real)	Match accuracy (synt.)
SIFT scale	0.28	0.52
Descriptor distance	0.07	0.11
Salience	-0.05	-0.11
Lowe score	0.18	0.28
SIFT scale $\times$ desc. dist. ($= \phi$)	0.34	0.62

Table 7.1: Correlation coefficient between match position accuracy and various parameters for real (middle column) and synthetic (right column) image pairs.

7.5 Algorithm

7.5.1 Match selection alone (without match refinement)

Our match selection algorithm is summarized on Figure 7.7. After feature detection and matching, matches M are cleaned up using K-VLD and ordered using the ranking

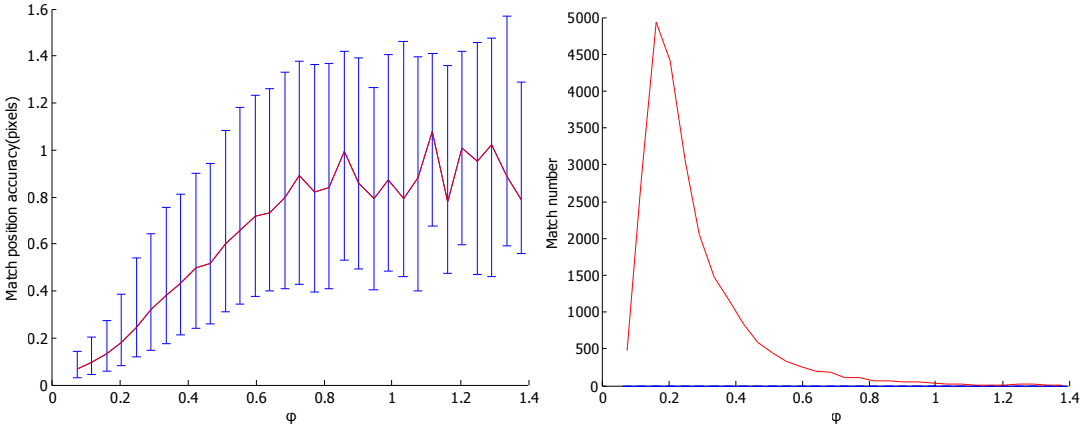


Figure 7.6: Left: Match position accuracy as a function of ϕ. Right: Match number histogram as a function of ϕ.

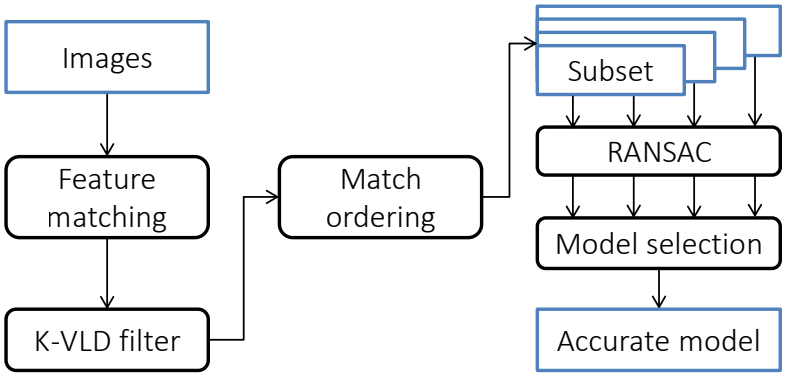


Figure 7.7: A global view of the match selection algorithm

function ϕ. Subsets M_{sub} of sorted matches are explored to minimize $e_F(M_{\text{sub}})^\gamma/|M_{\text{sub}}|$ and the subset with the lowest value is used to construct the accurate model.

7.5.2 Match selection with match refinement

Match selection (this chapter) and match refinement (Chapter 6) are independent improvements that can be combined, match refinement coming first (see Figure 7.8). However, match refinement changes the correlation between the match errors and the indicators of feature accuracy. When the two methods are combined, the match ranking to create subset candidates (see Section 7.3) has to be changed.

Based on experiments with the same semi-synthetic data as in Section 7.3, we found that, after match refinement, the dissimilarity measure η using the focused grid correlates with the actual feature localization error, with a score of 0.27. Besides, intuitively, the scaling and shearing of the image, as defined by the affinity estimate A, also has an impact on the quality of matching (see Figure 7.9). Given orthogonal unit vectors (u,v) in I, we consider the value $\max_{u,v} \frac{|u^T J^T J v|}{|Ju||Jv|}$ with $J = \begin{pmatrix} A_{00} & A_{01} \\ A_{10} & A_{11} \end{pmatrix}$, which is the cosine of the maximum crushing after transformation. It can be shown to be simply expressed as $\chi = \frac{|\lambda_1 - \lambda_2|}{\lambda_1 + \lambda_2}$, where $\lambda_1, \lambda_2 > 0$ are the eigenvalues of $J^T J$ (the proof is in Section 7.11.2). It has a correlation score of 0.12 with the localization error. By a linear regression over the same semi-synthetic data, we empirically define $\phi(m) = 0.19\eta + 0.97\chi$, which has a

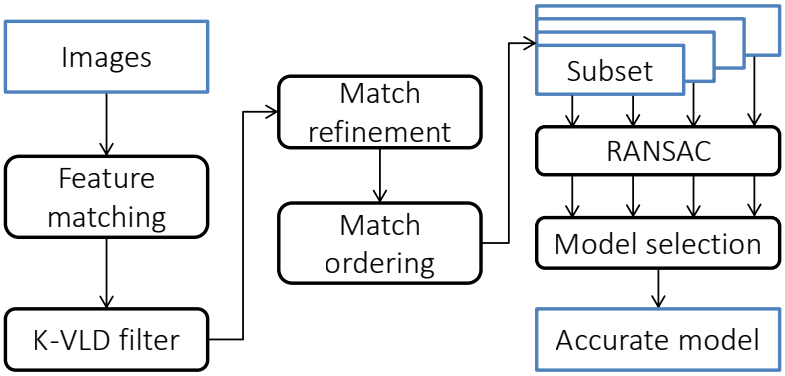


Figure 7.8: A global view of the combined algorithm

correlation of 0.49 with the location error on synthetic dataset and 0.18 on real dataset.[1] Note that the feature scale no longer correlates with the location error (correlation is only 0.01) and is thus discarded from the ranking function.

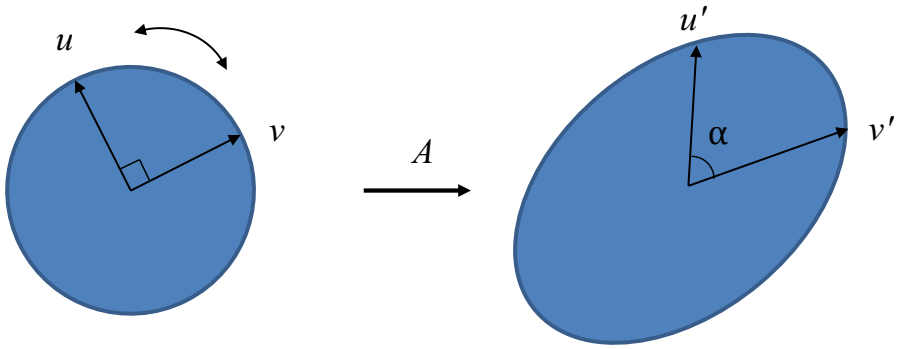


Figure 7.9: We look for $\max_{u,v}|\cos(\alpha)|$ with (u,v) orthogonal unit vectors passing through the feature center in image I.

7.5.3 Comparison to related methods

The PROSAC variant of RANSAC also constructs a series of match subsets and iterates first on better ones [20]. However, the target is not accuracy but fast convergence; robustness and precision are similar to RANSAC. Note that our method is not an alternative to RANSAC nor a fundamental matrix estimator, but a complement: the choice of a RANSAC variant as well as a fundamental matrix estimator is still required to compute the calibration and the corresponding epipolar error e_F for the different M_{sub} subsets considered. As a matter of fact, Section 7.6 shows that our method, combined with different variants of RANSAC, consistently provides much better results than using the RANSAC variant alone.

7.6 Experiments

To evaluate our method, we consider some RANSAC variants [19, 78] among those that are considered the most suited for accuracy (as opposed, e.g., to robustness or speed):

[1] These value are different from our published work [48], for two reasons. First, as mentioned above, we are more interested in accurate matches (with reprojection < 2 pixels) instead of taking all filtered matches. Second, we have slightly modified the refinement method, which leads to different correlation relations.

Dataset	Strecha et al. [73]					DTU robot [1]				
e_R (deg $\times 10^{-2}$)	raw	MS	MR	MR+MS	gain	raw	MS	MR	MR+MS	gain
RANSAC	15.7	9.59	10.9	9.78	1.9	27.4	22.8	21.5	21.2	1.3
MSAC	13.1	9.53	9.55	8.73	1.7	22.5	22.3	21.2	21.2	1.1
LO-RANSAC	16.0	9.71	10.86	9.6	1.8	27.2	22.7	21.6	21.2	1.3
MLESAC	18.0	7.58	6.97	6.95	2.0	23.2	23.2	21.6	21.2	1.1
ORSA	11.5	6.74	5.86	5.90	1.9	23.1	22.1	20.7	21.3	1.1
e_t (deg)	raw	MS	MR	MR+MS	gain	raw	MS	MR	MR+MS	gain
RANSAC	1.85	1.08	1.25	1.10	1.7	3.71	1.42	1.42	0.81	4.5
MSAC	1.41	1.03	1.08	0.96	1.5	1.22	0.99	0.84	0.61	2.0
LO-RANSAC	1.75	1.10	1.24	1.08	1.7	3.70	1.42	1.42	0.82	4.5
MLESAC	1.84	0.74	0.70	0.67	2.7	1.90	1.09	1.11	0.71	2.7
ORSA	1.32	0.70	0.59	0.55	2.4	1.20	0.88	0.65	0.72	1.7

Table 7.2: Average rotation and translation errors: RANSAC alone (raw), + match selection (MS), + match refinement (MR), + both (MR+MS), and gain raw/(MR+MS)

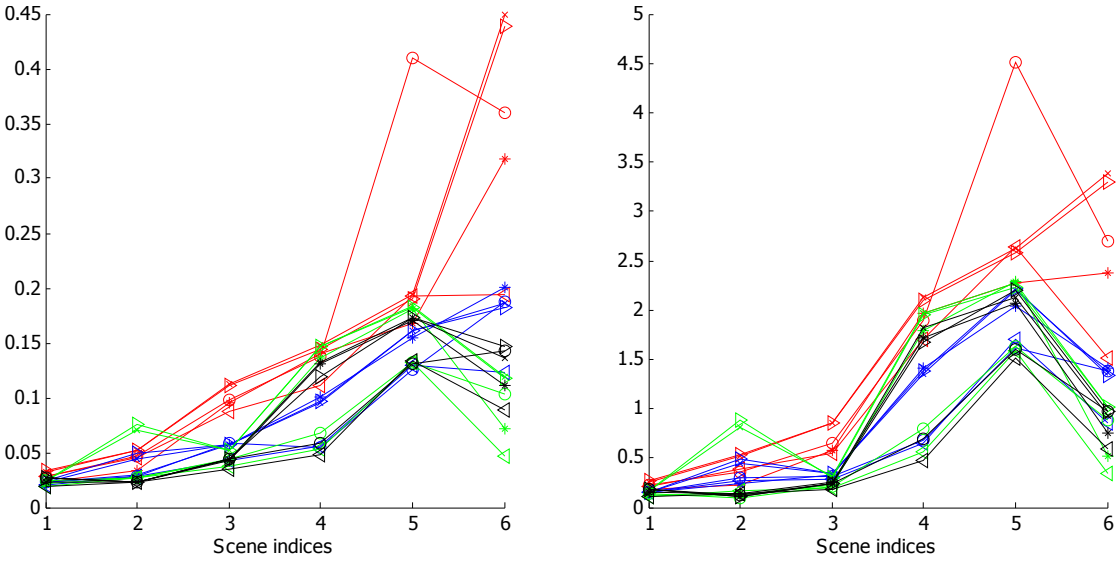
RANSAC with iterative re-weighted least squares (IRLS) for final model estimation [78, method S1], RANSAC with M-estimator (MSAC), LO-RANSAC [21], MLESAC [77], and ORSA with IRLS [57]. IRLS tries iteratively to minimize the sum of squares of geometric error between points in the right image and the epipolar line of corresponding points in left image. For each of these variants, we compare 4 settings: RANSAC alone, RANSAC preceded by match selection (MS), RANSAC preceded by match refinement (MR) using LSFM, and RANSAC preceded both by match refinement and match selection (MR+MS). A uniform threshold of 3 pixels (distance to epipolar line) is used in the RANSAC variants for outlier rejection, apart from ORSA that chooses the threshold automatically. All the results we provide are averaged over 20 runs.

Only datasets with highly accurate ground-truth calibration can be used for validation. We experimented with the full dataset of Strecha et al. [73], a de facto standard in camera calibration: 6 groups of 8 to 30 images totaling 95 pairs of successive images. For each pair, SIFT feature points are detected and matched with the usual setting [49] (no tweaking as in Sect. 7.2.3), i.e., a descriptor distance ratio to next best match at most 0.8. We ran the same experiment with the DTU robot dataset [1]. However, as it is huge (about 0.5 To), we only considered 9 of the 60 groups of images, covering various themes (scenes 1, 2, 4, 9, 10, 12, 21, 28, 52), in the reduced format (fewer images, yielding 12 images pairs: 1-12, 12-24, 24-25, 25-26, 26-37, 37-49, 50-57, 57-64, 57-65, 57-94, 64-95, 64-119), with identical illumination condition (number 08 for all tests), but full-size images.

7.6.1 Match selection and refinement

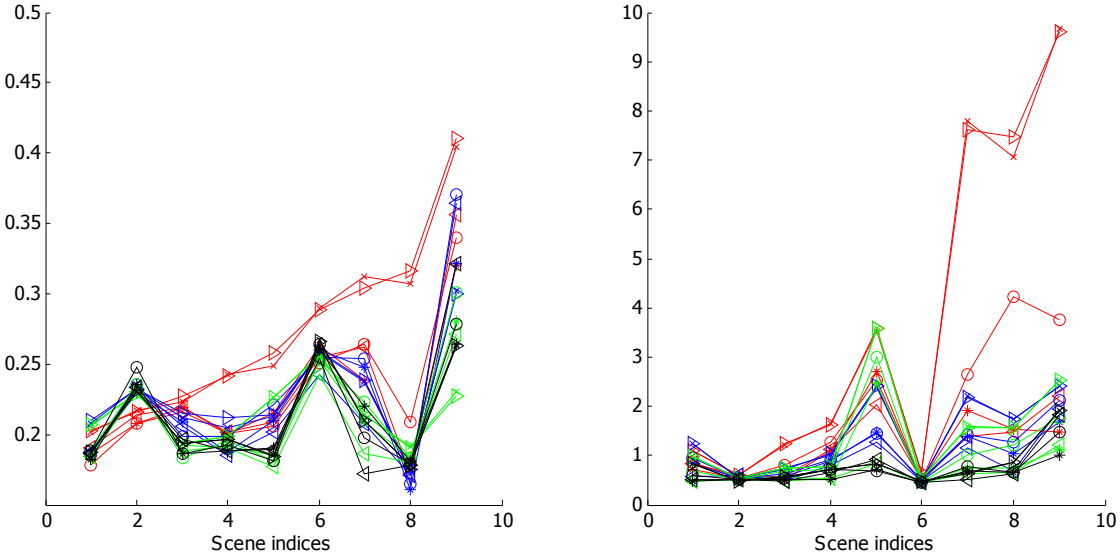
Figure 7.10 shows the average rotation and translation errors e_R, e_t for each scene of each dataset. Table 7.2 shows the average results[2], illustrating both the separate and combined benefits of MS and MR. Gain factors attain 2.0 for rotation and 4.5 for translation. Note that most parameters are learned on independent and widely different images [54]; only the lower bound $\gamma = 2$ is defined from the feature distribution of [73] and nothing else. Our results on datasets [1, 73] suggest that these parameters make sense for a wide range of images.

[2]For the same reasons as mentioned just before, our figures here are slightly different (actually better) than those in [48].



Strecha et al. (6 scenes) [73]

Figure 7.10: Average results on Strecha et al.'s datasets. Left: rotation error e_R in degree. Right: translation error e_t in degree (the angular error of the camera translation vector). Color red: raw RANSAC; blue: with match selection (MS); green: with match refinement using LSFM (MR); **black**: with both match selection and match refinement (MR+MS). Line symbol -▷-: RANSAC with IRLS; -*-: MSAC; -x-: LO-RANSAC; -o-: MLESAC; -◁-: ORSA. Scenes are reordered by increasing rotation error of RANSAC.



DTU robot (10 scenes) [1]

Figure 7.11: Average results on the DTU robot datasets. See Figure 7.10 for notations.

7.7 Visual illustration of 3D reconstruction accuracy

We now illustrate the accuracy of our method regarding 3D reconstruction, i.e., structure. The problem is that a 3D ground truth is not available for the considered datasets.

To get around this problem, we construct a *pseudo ground truth* based on exact rotation and translation, but approximate point matches: for each match $m = (\mathbf{x}, \mathbf{x}')$, in images I, I' with ground-truth camera centers C, C', we construct a 3D point $X_{\mathbf{x}}(\mathbf{x}')$ as

the point on line $\overline{C\mathbf{x}}$ that is the closest to line $\overline{C'\mathbf{x}'}$.

Note that we do not resort to ordinary triangulation here, e.g., mid-point of lines $\overline{C\mathbf{x}}$ and $\overline{C'\mathbf{x}'}$, gold-standard algorithm, etc. [34]. The reason is that a 3D point $\mathbf{X}_{(\mathbf{x},\mathbf{x}')}$ originating from ordinary triangulation provides a kind of middle ground between views $\mathbf{x}$ and $\mathbf{x}'$, where $(\mathbf{x},\mathbf{x}')$ does not try to aim at a *specific* 3D point. As a result, it does not make sense with respect to match refinement. The fact is match refinement is asymmetric; it only moves points in image I'. It yields a new putative match $(\mathbf{x},\mathbf{x}'')$ that tries to better locate $\mathbf{x}$ in 3D, which is different from $\mathbf{X}_{(\mathbf{x},\mathbf{x}')}$. On the contrary, if we consider 3D points $X_{\mathbf{x}}(\mathbf{x}')$ as indicated above, match refinement makes sense: we then try to get closer to the 3D ground truth location of $\mathbf{x}$ both before or after refinement.

A drawback, though, is that the error of the pseudo ground truth with respect to the unknown actual ground truth might be doubled compared to the ordinary triangulation case. We accept that and consider the measure as relative but fair in the sense that we evaluate all SfM methods with exactly the same 3D reconstruction principle.

Given that we only have a pseudo ground truth for point cloud (true camera positions with estimated 3D point position), it does not make sense to provide quantitative results for e_{3D}. Instead, we provide a qualitative visualization on two representative image pairs.

Figures 7.12 and 7.13 show how our approach compares to RANSAC-only: reconstructed 3D points are much closer to the pseudo ground truth with our method. Note that points on the top left and top right parts of the views are not outliers; they correspond to points on the roof. Figures 7.14 and 7.15 provide a similar example.

Figure 7.12: An image pair in Strecha et al.'s dataset.

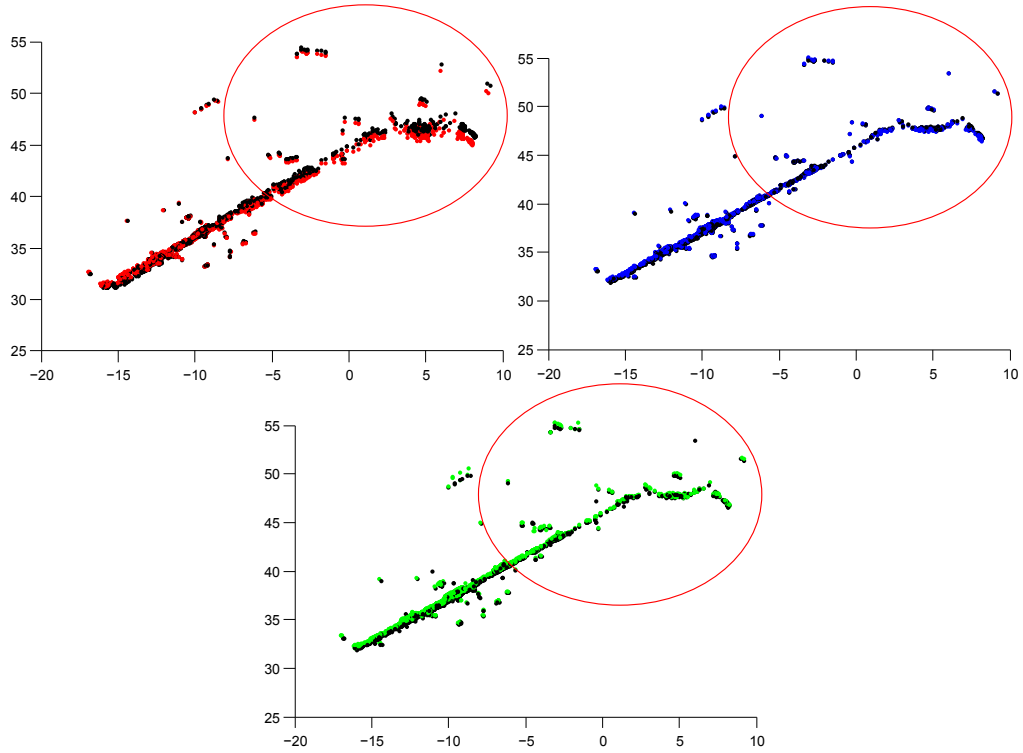


Figure 7.13: View from above of the 3D points reconstructed from the image pair in Figure 7.12. The colors are as follows:
– **black**: pseudo ground truth,
– **red**: using ORSA alone,
– **blue**: using match selection (MS) before ORSA,
– **green**: using match refinement followed by match selection(MR+MS) before ORSA.
The difference is more noticeable in the circled area.

Figure 7.14: Another image pair in Strecha et al.'s dataset.

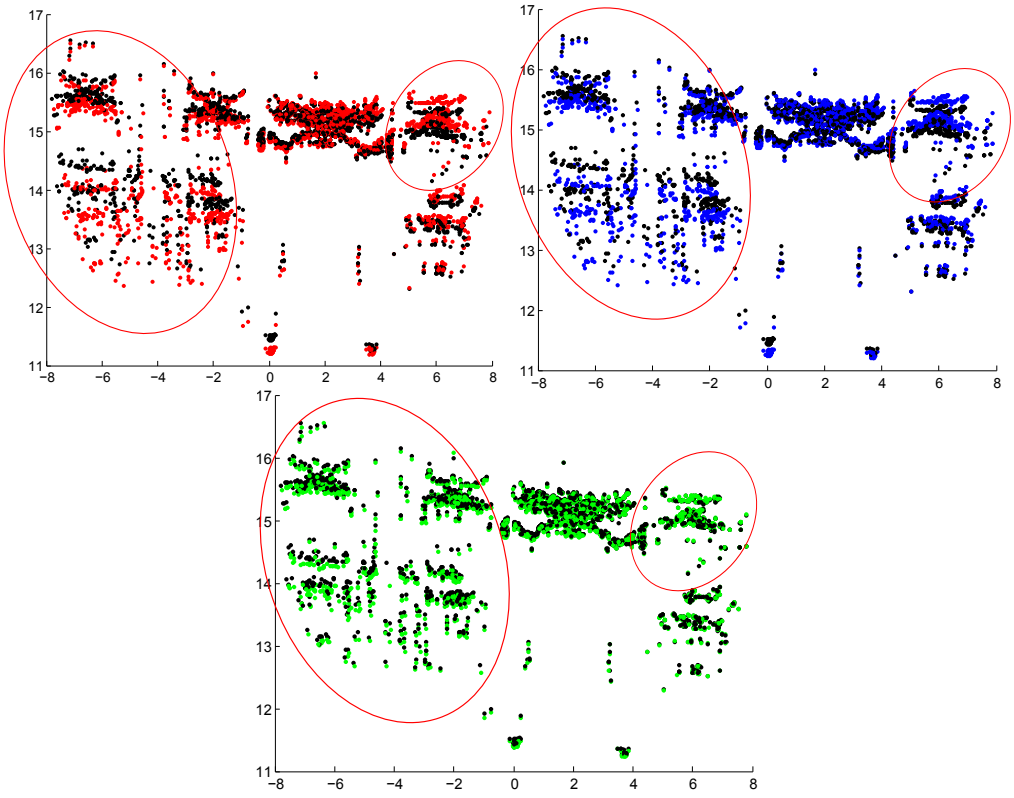


Figure 7.15: Front view of the 3D point cloud reconstructed from the image pair shown in Figure 7.14. The colors are as follows:
– **black**: pseudo ground truth,
– **red**: using ORSA alone,
– **blue**: using match selection (MS) before ORSA,
– **green**: using match refinement followed by match selection(MR+MS) before ORSA.
The difference is more noticeable in the circled areas.

7.8 Number of matches kept by match selection

Match selection removes matches when they are likely to degrade accuracy. Experiments of Section 7.6 show that the remaining matches reduce the rotation and translation errors with respect to actual ground truth. It is interesting to look at the number or proportion of matches that are discarded.

This is illustrated in Figure 7.16. Match selection alone (MS) keeps 61% of the matches on average. But preceded by match refinement (MR), match selection (MR+MS) keeps on average 78% of the matches, as they are more reliable. Note that the number of used matches may slightly increase after match refinement because some matches that were previously discarded by the final RANSAC stage (to compute motion) are now considered as inliers. Note also that the ratio of used matched N rarely goes lower than 40%, which justifies our heuristic for exploring only discrete fractions of $M_{\text{sub}}(N)$ starting from ratio $r = 0.4$ up (see Section 7.3.3).

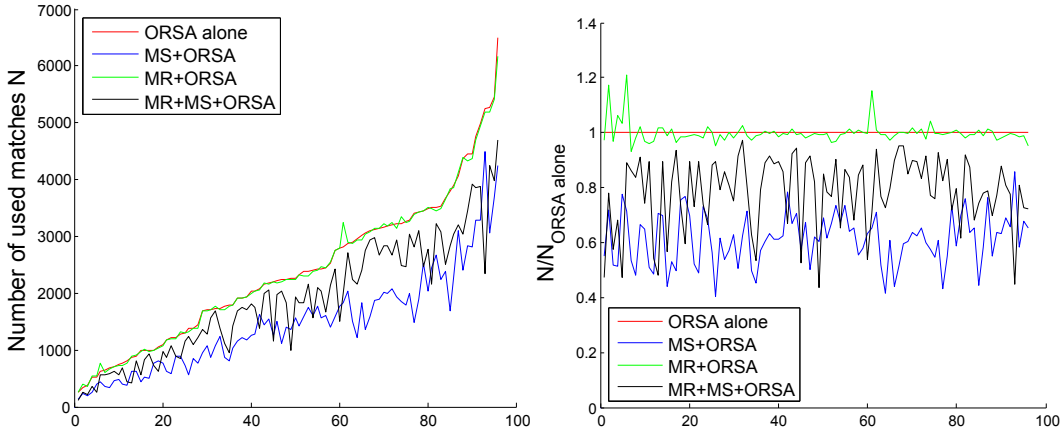


Figure 7.16: Left: number of matches selected to compute motion for image pairs in Strecha et al.'s dataset. Right: proportion of selected matches. (The ratio can be greater than 1 with MR-based methods as match refinement can turn outliers that are near inliers into actual inliers.) Image pairs are ordered by increasing number of matches for ORSA alone.

7.9 Various possible bias in alternative algorithms

7.9.1 Cleaning up matches with RANSAC before selection is biased

As mentioned in Section 7.3, the preliminary step of cleaning up matches before actual match selection consists in eliminating likely mismatches. It is crucial *not* to introduce any bias at this stage. We compare here some bad alternative of cleaning up choices for match selection due to bias.

There would be a bias if we were to filter the matches using RANSAC and an estimated epipolar geometry. This is illustrated on Figure 7.17 ("ORSA before MS"), on the 6 scenes of Strecha et al.'s dataset [73]: an increase in both rotation and translation errors can be observed if match selection (MS) is preceded by ORSA [57] to first clean up input matches.

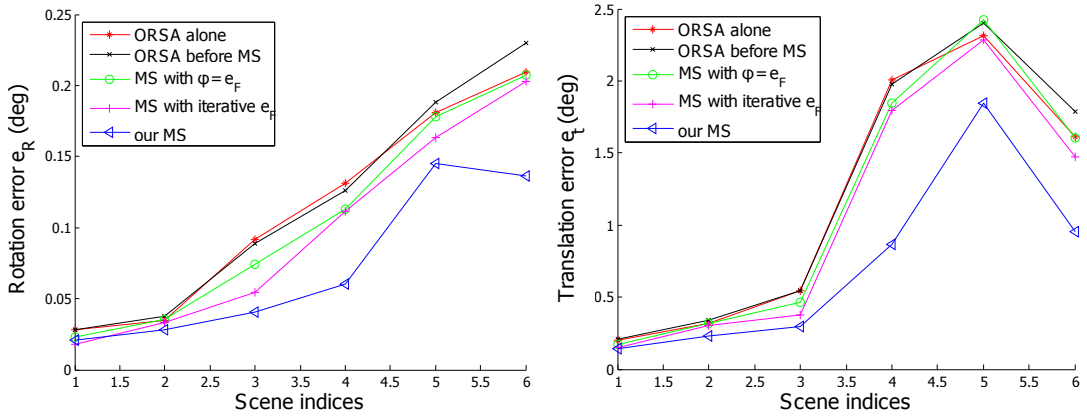


Figure 7.17: Possible bias with inappropriate match selection. Left: rotation error e_R on Strecha et al.'s dataset. Right: translation error e_t. Lines are defined as follows:
-+-: ordinary ORSA alone (an a-contrario variant of RANSAC),
-×-: MS preceded by ORSA to first clean up input matches,
-○-: MS using distance to epipolar line as ranking function ϕ,
-+-: MS using iterated distance to epipolar line and $r_{\min} = 0.4$,
-◁-: our MS method.
Scenes are ordered by increasing rotation error for ORSA alone.

7.9.2 Distance to the epipolar line is biased for ranking matches

Match selection relies on a ranking function ϕ to order the matches (cf. Section 7.4). However, using geometrical information in function ϕ introduces a bias. In particular, it is not appropriate to use the distance to the estimated epipolar line to rank the matches, e.g., to define $\phi(m) = e_F(M,m)$. This is illustrated on Figure 7.17 ("MS with $\phi = e_F$"), also on the 6 scenes of Strecha et al.'s dataset: results are not as good as with our unbiased ranking function.

This estimate can be slightly improved, although still with a bias. After estimating a fundamental matrix $F_{M'}$ for a given subset of matches $M' \subset M$, and considering another subset of matches $M_{\text{sub}} \subset M$, we can compute $e_F(M', M_{\text{sub}})$, the root mean square error of the distance of matches in M_{sub} to the $F_{M'}$-epipolar lines. The matches $m \in M$ can then be ordered by increasing distance $e_F(M', m)$ as a sequence $(m_i)_{1 \le i \le |M|}$ such that $i < j \Rightarrow e_F(M', m_i) \le e_F(M', m_j)$. Noting $M'_{|n} = \{m_i \mid 1 \le n\}$ the first n matches in M' and setting a minimum number of matches $N_{\min}$ to retain, we can easily find the exact optimal subset $M'^* \subset M$ with respect to $F_{M'}$:

$$\begin{aligned} M'^* &= \underset{\substack{M_{\text{sub}} \subset M \\ N_{\min} \le |M_{\text{sub}}|}}{\arg\min} \frac{e_F(M', M_{\text{sub}})^2}{|M_{\text{sub}}|} \\ &= \underset{\substack{M_{\text{sub}} = M'_{|n} \\ N_{\min} \le n \le |M|}}{\arg\min} \frac{e_F(M', M_{\text{sub}})^2}{|M_{\text{sub}}|} \\ &= M'_{|n^*}, \text{ with } n^* = \underset{N_{\min} \le n \le |M|}{\arg\min} \frac{e_F(M', M'_{|n})^2}{n} \end{aligned}$$

A linear exploration of n in $\{N_{\min}, \ldots, |M|\}$ is enough to compute n^*, and then $M'^* = M'_{|n^*}$. Starting with $M'_0 = M$, defining $M'_{k+1} = M'^*_k$, and stopping when $M'^*_k = M'_k$, we can itera-

tively get a good estimate for $M^*_{\text{sub}} \subset M$ defined as:

$$M^*_{\text{sub}} = \underset{M_{\text{sub}} \subset M}{\arg\min} \frac{e_F(M_{\text{sub}}, M_{\text{sub}})^2}{|M_{\text{sub}}|}. \tag{7.13}$$

As shown of Figure 7.17 ("MS with iterative e_F"), results with this estimate for minimum ratio of kept points $r_{\min} = N_{\min}/|M'| = 40\%$ are slightly better on average than with $\phi(m) = e_F(M, m)$, but still not as good as with ϕ as defined in Equation 7.12 Moreover, experiments show that this algorithm tends to lead to values of $|M'^*_k|$ that are close to $N_{\min}$, which means it is not well behaved.

7.10 Conclusion

In this chapter we have studied, in the two-view case, the "quality vs. quantity" balance of point matches for structure from motion — a poorly addressed issue in the literature. We have found a correlation between SfM errors and a function of the number of matches and their epipolar errors. Using this relation, we have presented a new method for selecting relevant subsets of points to improve SfM accuracy. Using extensive experiments involving real data with ground-truth calibration, we have shown that match selection and match refinement independently lead to a major reduction of SfM errors over the best methods targeted at accuracy. Combining both methods, the error is reduced by factors up to 2.0 for rotations and 3.8 for translations, a huge improvement.

Our work is valuable for stereovision. Extending it to the multi-view case is not trivial because of track consistency. First, removing one match does not necessarily remove the associated points from the track and leads to a substantially different bundle adjustment problem. Second, the location of points in a track would need to be optimized simultaneously in all associated images. We actually want *track selection* (or *reduction*) as well as *track refinement*. Besides, a good term to minimize to assess the benefit of match reduction is likely to be linked to the total reprojection error with respect to all 3D points *after* bundle adjustment. A study similar to that of Section 7.2 thus has to be carried out.

Still, a lower bound of the possible improvement can be obtained by applying match selection (MS) on each image pair in an SfM pipeline, before actual processing by the system. A preliminary experiment on Strecha et al.'s dataset using OpenMVG [61], a competitor to Bundler, shows improvements up to 15% on the average camera location error, in particular on scenes with wider viewpoint changes and fewer images (HerzJesu-P8 vs -P25, Castle-P19 vs -P30). Conversely, it may be the case that bundle adjustment is efficient at averaging on long tracks, compensating for the inaccuracy of point location. Track selection and track refinement are thus likely to be more profitable on difficult scenes.

Finally, most of our results are constructed on empirical studies. We however believe the "quality vs. quantity" issue deserves a better theoretical treatment, including a study of the influence of the configuration of points in images.

7.11 Annex

7.11.1 Regression correlation coefficient

This measures the strength and the direction of a linear relationship between two variables. Suppose we have n observation of the two variables noted as x_i and y_i, $i \in [1 \dots n]$:

$$RCC = \frac{n\sum_{i=1}^{n} x_i y_i - (\sum_{i=1}^{n} x_i)(\sum_{i=1}^{n} y_i)}{\sqrt{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}\sqrt{n\sum_{i=1}^{n} y_i^2 - (\sum_{i=1}^{n} y_i)^2}}. \tag{7.14}$$

7.11.2 Maximum crushing expression derivation

Given orthogonal unit vectors (u,v) in I and an affine transformation matrix A, we consider the value $\max_{u,v} \frac{|u^T J^T Jv|}{|Ju||Jv|}$ with $J = \begin{pmatrix} A_{00} & A_{01} \\ A_{10} & A_{11} \end{pmatrix}$, which is the cosine of the maximum crushing after transformation. We prove that it can be simply expressed as $\chi = \frac{|\lambda_1 - \lambda_2|}{\lambda_1 + \lambda_2}$, where $\lambda_1, \lambda_2 > 0$ are the eigenvalues of $J^T J$. Thus $\max_{u,v} |\cos(\alpha)| = \frac{|\lambda_1 - \lambda_2|}{\lambda_1 + \lambda_2}$, with α illustrated in Figure 7.9.

Let $m = (\mathbf{x}, \mathbf{x}')$ be a refined match with estimated affine transformation A. The transformed vectors (u', v') are equal to (Ju, Jv) with $J = \begin{pmatrix} A_{00} & A_{01} \\ A_{10} & A_{11} \end{pmatrix}$. We get $|\cos(\alpha)| = \frac{|u^T J^T Jv|}{|Ju||Jv|}$. We note (V_1, V_2) the unit eigen-vectors of $J^T J$ with non-negative eigenvalues λ_1, λ_2 and decompose (u,v) under (V_1, V_2) as $u = \cos(\theta)V_1 + \sin(\theta)V_2$ and $v = -\sin(\theta)V_1 + \cos(\theta)V_2$. We express $\cos^2(\alpha)$ as:

$$\begin{aligned} \cos^2(\alpha) &= \frac{\sin^2(\theta)\cos^2(\theta)(\lambda_1 - \lambda_2)^2}{[\lambda_1 \cos^2(\theta) + \lambda_2 \sin^2(\theta)][\lambda_1 \sin^2(\theta) + \lambda_2 \cos^2(\theta)]} \\ &= \frac{\sin^2(\theta)\cos^2(\theta)(\lambda_1 - \lambda_2)^2}{\sin^2(\theta)\cos^2(\theta)(\lambda_2 + \lambda_1)^2 + (\sin^2(\theta) - \cos^2(\theta))^2 \lambda_1 \lambda_2} \end{aligned} \tag{7.15}$$

Since the term in the denominator $(\sin^2(\theta) - \cos^2(\theta))^2 \lambda_1 \lambda_2 \geq 0$, we get:

$$\cos^2(\alpha) \leq \frac{\sin^2(\theta)\cos^2(\theta)(\lambda_1 - \lambda_2)^2}{\sin^2(\theta)\cos^2(\theta)(\lambda_1 + \lambda_2)^2} = \frac{(\lambda_1 - \lambda_2)^2}{(\lambda_1 + \lambda_2)^2}. \tag{7.16}$$

Thus we have

$$\max_{u,v} |\cos(\alpha)| = \frac{|\lambda_1 - \lambda_2|}{|\lambda_1 + \lambda_2|} \text{ reached when } |\sin(\theta)| = |\cos(\theta)| \Leftrightarrow \theta = \frac{\pi}{4} \mod \frac{\pi}{2}. \tag{7.17}$$

Chapter 8

Conclusion and perspectives

8.1 Conclusion

The work presented in this manuscript essentially aims at improving the SfM camera calibration and hence the 3D reconstruction accuracy in the two-view case. We have proposed different solutions to improve matching and camera pose refinement in order to increase the robustness and accuracy in the SfM process.

Robustness and accuracy of matching

We have analyzed the feature matching processes, with graph-matching methods and with RANSAC-like methods. We have found that the graph-matching methods lack a robust match constraint based on photometric comparison and the RANSAC-like methods are limited in trying to eliminate false matches *away* from the epipolar lines. A middle-level match validation process based on photometric consistence is needed to improve the feature matching using a less local method. For that, our solution is the following:

- We have defined a *virtual line descriptor* (VLD) that encodes photometric information between points by a chain of SIFT-like descriptors. It offers the possibility to compute a photometric consistency of a pair of matches by comparing their connecting VLDs in the two images.
- We have proposed K-VLD, a semi-local matching method using the VLD constraint. This method verifies the photometric consistency of every match at a larger scale than feature descriptors. K-VLD is a kind of light photometric-based graph-matching method, which can be used as a filter before RANSAC methods. It is robust both to rigid and non-rigid deformations. With a few modifications, it can also be used to define an efficient segment matching algorithm.

The semi-local property allows the K-VLD method to apply to a large number of matches in a reasonable time. In our experiments, the practical complexity of our algorithm is quasi linear in the number of matches. On standard desktop machines, it takes about one minute to process 10,000 matches and just a few seconds to process 1,000.

Accuracy of camera pose estimation

In order to make better camera pose estimation, we have analyzed the behavior of camera rotation and translation errors under different conditions (varying image pair,

number of matches and match error distribution). We have found that using all inliers to estimate the fundamental matrix is not the optimal choice to get a better accuracy.

- A study of reconstruction accuracy in the two-view SfM case has been carried out with theoretical support and experimental observations while varying the number of matches and the accuracy of matches. We obtain a "quantity vs. quality" trade-off, which is safe for reconstruction accuracy.
- The "quantity vs. quality" study shows the possibility of using fewer matches with higher location accuracy to get a better resulting accuracy in camera pose estimation. This differs widely from traditional approaches, which use all inliers to estimate the camera pose. Our match selection approach considers a series of match subsets of different sizes and generate estimation candidates. The subset and estimation with the optimal "quantity vs. quality" trade-off is kept.
- We have also applied the match-selection approach to features refined by our least square matching extension and demonstrated that the match selection method works even better.

8.2 Perspectives

K-VLD level up

As an initial work in semi-local matching method, K-VLD can be improved in several aspects.

Speed and performance. The neighbor searching process is quadratic in the number of features. Given the fact that K-VLD looks only for approximately close matches without order, with a high-enough number of neighbors, some more efficient neighbor searching methods can be applied. For instance, the kD-tree structure used in [64] would be well suited to our needs.

Besides, the VLD descriptor is strongly inspired by SIFT. It could be transposed to a number of other existing descriptors to reduce the memory size of a VLD or to accelerate its computation and comparison time.

To add invariance (at least in certain directions) against perspective transformation, it also seems possible to use a non-uniform region according to the scale ratio between features on the extremities of VLD.

Memory requirement. K-VLD requires a $N \times N$ matrix to store pairwise match consistency information, where N is the number of matches. But the algorithm tests only a very small portion of the $N \times N$ comparisons. Every match compares only with its neighbors, and the comparison stops before testing all neighbors when there are enough supporting neighbors. Thus, a sparse structure would be more adequate than an complete matrix.

Extension of match refinement and match selection

Presently, our match refinement and match selection are only applied to two-view SfM and with a ranking function based on a simple combination of correlated parameters. There is promising future work:

Machine learning. First, the defined ranking function is quite simple. A deeper study in match localization accuracy is needed to produce a better ranking function. It is likely that using a learning process over complex parameters would further improve the result.

Accuracy modelization. We have studied the impact of match quality and match quantity over the reconstruction results. Other factors such as the match configuration (i.e., distribution over images) have not been taken into account, except as dataset variations to average or bound. A analysis of specific match configurations should provide a better model to estimate accuracy.

N-view case. Our methods has only been tested over two-view structure from motion. A naive application in using our method as an input filter did not show remarkable improvements in accuracy for N-view SfM. A better and more natural extension to the N-view case would consist in studying track refinement and track selection or reduction. For track refinement, the difficulty is that the number of refinement parameters increases with the length of the track, making the algorithm hard to converge to a global minimum. In terms of track selection/reduction, a reliable ranking function is needed, which we have not discovered yet. However, if we "cheat" by using as ranking function for tracks the average re-projection error calculated with ground-truth camera positions, we can reduce by 40% the translation error using a global SfM method [62], which indicates the remaining potential of progress.

Bibliography

[1] HENRIK AANÆS, ANDERS LINDBJERG DAHL, and KIM STEENSTRUP PEDERSEN. Interesting interest points. In: *International Journal of Computer Vision*, **97**:1 (2012), pp. 18–35 (see pp. 110, 111, 129, 130)

[2] CUNEYT AKINLAR and CIHAN TOPAL. Edlines: Real-time line segment detection by Edge Drawing (ed). In: *Image Processing (ICIP), 2011 18th IEEE International Conference on*. IEEE. 2011, pp. 2837–2840 (see p. 84)

[3] A. ALBARELLI, E. RODOLA, and A. TORSELLO. Robust game-theoretic inlier selection for bundle adjustment. In: *3DPVT*. 2010 (see pp. 42, 50, 52, 60, 91)

[4] PABLO FERNÁNDEZ ALCANTARILLA, ADRIEN BARTOLI, and ANDREW J DAVISON. "KAZE features". In: *Computer Vision–ECCV 2012*. Springer, 2012, pp. 214–227 (see p. 35)

[5] CÉDRIC ALLENE, J-P PONS, and RENAUD KERIVEN. Seamless image-based texture atlases using multi-band blending. In: *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*. IEEE. 2008, pp. 1–4 (see p. 77)

[6] GEORGES BAATZ, KEVIN KÖSER, DAVID CHEN, RADEK GRZESZCZUK, and MARC POLLEFEYS. Leveraging 3D city models for rotation invariant place-of-interest recognition. In: *International journal of computer vision*, **96**:3 (2012), pp. 315–334 (see p. 102)

[7] SIMON BAKER, DANIEL SCHARSTEIN, JP LEWIS, STEFAN ROTH, MICHAEL J BLACK, and RICHARD SZELISKI. A database and evaluation methodology for optical flow. In: *IJCV*, **92**:1 (2011), pp. 1–31 (see p. 102)

[8] H. BAY, A. ESS, T. TUYTELAARS, and L. VAN GOOL. Speeded-Up Robust Features (SURF). In: *Computer Vision and Image Understanding*, **110**:3 (2008), pp. 346–359 (see p. 49)

[9] ALEXANDER C. BERG, TAMARA L. BERG, and JITENDRA MALIK. Shape matching and object recognition using low distortion correspondence. In: *CVPR*. IEEE. 2005, pp. 26–33 (see pp. 42, 50)

[10] MICHAEL J BROOKS, WOJCIECH CHOJNACKI, DARREN GAWLEY, and ANTON VAN DEN HENGEL. What value covariance information in estimating vision parameters? In: *Proceedings of the 8th IEEE International Conference on Computer Vision (ICCV)*. Vol. 1. IEEE. 2001, pp. 302–308 (see p. 105)

[11] MATTHEW BROWN and DAVID G LOWE. Automatic panoramic image stitching using invariant features. In: *International journal of computer vision*, **74**:1 (2007), pp. 59–73 (see p. 77)

[12] MATTHEW BROWN and DAVID G LOWE. Invariant features from interest point groups. In: *BMVC*. s 1. 2002 (see p. 35)

[13] CMP, CZECH TECHNICAL UNIVERSITY, PRAGUE. *Dětenice fountain dataset*. http://cmp.felk.cvut.cz/projects/is3d/Data.html. 2008 (see p. 60)

[14] MICHAEL CALONDER, VINCENT LEPETIT, MUSTAFA OZUYSAL, TOMASZ TRZCINSKI, CHRISTOPH STRECHA, and PASCAL FUA. BRIEF: Computing a local binary descriptor very fast. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **34**:7 (2012), pp. 1281–1298 (see p. 36)

[15] YANPENG CAO and JOHN MCDONALD. Viewpoint invariant features from single images using 3D geometry. In: *Applications of Computer Vision (WACV), 2009 Workshop on*. IEEE. 2009, pp. 1–6 (see p. 102)

[16] M. CHERTOK and Y. KELLER. Efficient high order matching. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, **32**:12 (2010), pp. 2205–2215 (see pp. 43, 50)

[17] MINSU CHO and JUNGMIN LEE. Feature correspondence and deformable object matching via agglomerative correspondence clustering. In: *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE. 2009, pp. 1280–1287 (see p. 50)

[18] MINSU CHO, JUNGMIN LEE, and KYOUNG MU LEE. Reweighted random walks for graph matching. In: *European Conference on Computer Vision (ECCV)*. 2010, pp. 492–505 (see pp. 42, 50)

[19] SUNGLOK CHOI, TAEMIN KIM, and WONPIL YU. Performance evaluation of RANSAC family. In: *BMVC*. 2009, pp. 1–12 (see pp. 41, 128)

[20] ONDREJ CHUM and JIRI MATAS. Matching with PROSAC – progressive sample consensus. In: *CVPR*. 2005 (see pp. 49, 117, 128)

[21] ONDREJ CHUM, JIRI MATAS, and STEPAN OBDRZALEK. Enhancing RANSAC by generalized model optimization. In: *ACCV*. 2004 (see pp. 42, 117, 129)

[22] D. CONTE, P. FOGGIA, C. SANSONE, and M. VENTO. Thirty years of graph matching in pattern recognition. In: *IJPRAI*, **18**: (2004), pp. 265–298 (see pp. 42, 50)

[23] O. DUCHENNE, F. BACH, I. KWEON, and J. PONCE. A tensor-based algorithm for high-order graph matching. In: *Conference on Computer Vision and Pattern Recognition*. IEEE. 2009, pp. 1980–1987 (see p. 50)

[24] OLIVIER DUCHENNE, FRANCIS BACH, IN-SO KWEON, and JEAN PONCE. A tensor-based algorithm for high-order graph matching. In: *PAMI*, **33**:12 (Dec. 2011), pp. 2383–2395 (see pp. 43, 50, 60)

[25] BIN FAN, FUCHAO WU, and ZHANYI HU. Line matching leveraged by point correspondences. In: *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE. 2010, pp. 390–397 (see pp. 80, 86)

[26] VITTORIO FERRARI, TINNE TUYTELAARS, and LUC VAN GOOL. "Simultaneous object recognition and segmentation by image exploration". In: *Computer Vision-ECCV 2004*. Springer, 2004, pp. 40–54 (see p. 50)

[27] MARTIN A FISCHLER and ROBERT C BOLLES. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. In: *CACM*, **24**:6 (1981) (see pp. 39, 49, 117)

[28] P.E. FORSSEN and D.G. LOWE. Shape descriptors for maximally stable extremal regions. In: *ICCV 2007. IEEE 11th International Conference on Computer Vision, 2007*. IEEE. 2007, pp. 1–8 (see pp. 35, 49)

[29] ARMIN GRUEN. Adaptive least squares correlation: a powerful image matching technique. In: *S. Afr. J. of Photogrammetry, Remote Sensing and Cartography*, **14**:3 (1985) (see pp. 102, 104)

[30] S. GU, Y. ZHENG, and C. TOMASI. Critical nets and beta-stable features for image matching. In: *European Conference on Computer Vision (ECCV)*. 2010, pp. 663–676 (see p. 50)

[31] YOAV HACOHEN, ELI SHECHTMAN, DAN B GOLDMAN, and DANI LISCHINSKI. Non-rigid dense correspondence with applications for image enhancement. In: *ACM Transactions on Graphics (TOG)*. Vol. 30. 4. ACM. 2011, p. 70 (see pp. 76, 78)

[32] YOAV HACOHEN, ELI SHECHTMAN, DAN B GOLDMAN, and DANI LISCHINSKI. Optimizing color consistency in photo collections. In: *ACM Transactions on Graphics (TOG)*, **32**:4 (2013), p. 38 (see p. 76)

[33] CHRIS HARRIS and MIKE STEPHENS. A combined corner and edge detector. In: *Alvey vision conference*. Vol. 15. Manchester, UK. 1988, p. 50 (see p. 34)

[34] R. I. HARTLEY and A. ZISSERMAN. *Multiple view geometry in computer vision*. Cambridge University Press, 2004. ISBN: 0521540518 (see pp. 27, 31, 49, 81, 111, 118, 131)

[35] M. HOFER, A. WENDEL, and H. BISCHOF. Incremental line-based 3D reconstruction using geometric constraints. In: *BMVC13*. 2013 (see p. 80)

[36] KENICHI KANATANI and YASUYUKI SUGAYA. "Compact fundamental matrix computation". In: *Advances in Image and Video Technology*. Springer, 2009, pp. 179–190 (see p. 43)

[37] YASUSHI KANAZAWA and KENICHI KANATANI. Do we really have to consider covariance matrices for image features? In: *Proceedings of the 8th IEEE International Conference on Computer Vision (ICCV)*. Vol. 2. IEEE. 2001, pp. 301–306 (see p. 105)

[38] W. KAREL, M. DONEUS, C. BRIESE, G. VERHOEVEN, and N. PFEIFER. Investigation on the automatic geo-referencing of archaeological UAV photographs by correlation with pre-existing ortho-photos. In: *ISPRS Technical Commission V Symposium*. 2014, pp. 307–312 (see p. 97)

[39] WILFRIED KAREL, MICHAEL DONEUS, GEERT VERHOEVEN, CHRISTIAN BRIESE, CAMILLO RESSL, and NORBERT PFEIFER. Oriental: automatic geo-referencing and ortho-rectification of archaeological aerial photographs. In: *XXIV International CIPA Symposium*. Vol. 2. 2013, pp. 175–180 (see p. 97)

[40] KEVIN KÖSER and REINHARD KOCH. Exploiting uncertainty propagation in gradient-based image registration. In: *BMVC*. 2008 (see p. 108)

[41] KEVIN KÖSER and REINHARD KOCH. Perspectively invariant normal features. In: *ICCV*. 2007 (see p. 102)

[42] SUMEDHA KSHIRSAGAR, STEPHANE GARCHERY, and NADIA MAGNENAT-THALMANN. "Feature point based mesh deformation applied to mpeg-4 facial animation". In: *Deformable Avatars*. Springer, 2001, pp. 24–34 (see p. 50)

[43] J. LEE, M. CHO, and K.M. LEE. Hyper-graph matching via reweighted random walks. In: *IEEE Conf. Computer Vision & Pattern Recognition (CVPR)*. 2011, pp. 1633–1640 (see pp. 43, 50, 60)

[44] M. LEORDEANU and M. HEBERT. A spectral technique for correspondence problems using pairwise constraints. In: *10th ICCV*. Vol. 2. IEEE. 2005, pp. 1482–1489 (see pp. 42, 50, 52, 60)

[45] M. LEORDEANU, R. SUKTHANKAR, and M. HEBERT. Unsupervised learning for graph matching. In: *International Journal of Computer Vision (IJCV)*, **96**:1 (2012), pp. 28–45 (see pp. 60, 66, 91)

[46] T. LINDEBERG. Feature detection with automatic scale selection. In: *International Journal of Computer Vision*. Vol. 30(2). 1998, pp. 79–116 (see p. 35)

[47] CE LIU, JENNY YUEN, and ANTONIO TORRALBA. Sift flow: Dense correspondence across scenes and its applications. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **33**:5 (2011), pp. 978–994 (see p. 76)

[48] ZHE LIU, MONASSE PASCAL, and RENAUD MARLET. Match selection and refinement for highly accurate two-view structure from motion. In: *European Conference on Computer Vision*. 2014 (see pp. 125, 128, 129)

[49] D.G. LOWE. Distinctive image features from scale-invariant keypoints. In: *IJCV*, **60**:2 (2004), pp. 91–110 (see pp. 49, 52, 53, 64, 66, 70, 72, 102, 106, 110, 119, 129)

[50] DAVID G LOWE. Object recognition from local scale-invariant features. In: *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*. Vol. 2. Ieee. 1999, pp. 1150–1157 (see pp. 35, 36, 38)

[51] ANDRÁS L MAJDIK, YVES ALBERS-SCHOENBERG, and DAVIDE SCARAMUZZA. MAV urban localization from Google street view data. In: *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE. 2013, pp. 3979–3986 (see pp. 18, 97, 99)

[52] J. MATAS, O. CHUM, M. URBAN, and T. PAJDLA. Robust wide-baseline stereo from maximally stable extremal regions. In: *Image and Vision Computing*, **22**:10 (2004), pp. 761–767 (see pp. 35, 42, 49, 102)

[53] K. MIKOLAJCZYK and C. SCHMID. A performance evaluation of local descriptors. In: *IEEE Trans. Pattern Anal. Mach. Intell. (PAMI)*, **27**:10 (Oct. 2005), pp. 1615–1630 (see pp. 36, 60–62, 66, 72, 125)

[54] K. MIKOLAJCZYK and C. SCHMID. Scale & affine invariant interest point detectors. In: *IJCV*, **60**:1 (2004), pp. 63–86 (see pp. 49, 102, 106, 125, 129)

[55] KRYSTIAN MIKOLAJCZYK and CORDELIA SCHMID. An affine invariant interest point detector. In: (2002), pp. 128–142 (see p. 35)

[56] KRYSTIAN MIKOLAJCZYK and CORDELIA SCHMID. Indexing based on scale invariant interest points. In: *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*. Vol. 1. IEEE. 2001, pp. 525–531 (see p. 35)

[57] L. MOISAN and B. STIVAL. A probabilistic criterion to detect rigid point matches between two images and estimate the fundamental matrix. In: *IJCV*, **57**:3 (2004), pp. 201–218 (see pp. 42, 49, 60, 64, 66, 99, 110, 117–119, 129, 134)

[58] L. MOISAN, P. MOULON, and P. MONASSE. Automatic homographic registration of a pair of images, with a contrario elimination of outliers. In: *Image Processing On Line (IPOL)*, (2012). http://www.ipol.im/pub/algo/mmm_orsa_homography (see p. 60)

[59] JEAN-MICHEL MOREL and GUOSHEN YU. ASIFT: A new framework for fully affine invariant image comparison. In: *SIAM Journal on Imaging Sciences*, **2**:2 (2009), pp. 438–469 (see pp. 35, 64, 102)

[60] PIERRE MOULON. *Syntethic images used for testing ASIFT*. http://www.ipol.im/pub/demo/my_affine_sift/archive?key=0DA014C43786492B0A854FAD5DE5EACD. 2011 (see p. 64)

[61] PIERRE MOULON, PASCAL MONASSE, and RENAUD MARLET. Adaptive Structure from Motion with a contrario model estimation. In: *ACCV*. 2012 (see pp. 119, 136)

[62] PIERRE MOULON, PASCAL MONASSE, and RENAUD MARLET. Global fusion of relative motions for robust, accurate and scalable structure from motion. In: *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE. 2013, pp. 3248–3255 (see pp. 46, 68, 69, 141)

[63] PIERRE MOULON, BRUNO DUISIT, and PASCAL MONASSE. Global multiple-view color consistency. In: *Proceedings of Conference on Visual Media Production*. 2013 (see pp. 77, 78)

[64] MARIUS MUJA and DAVID G LOWE. Fast approximate nearest neighbors with automatic algorithm configuration. In: *VISAPP (1)*. 2009, pp. 331–340 (see pp. 38, 140)

[65] DAVID OK, RENAUD MARLET, and JEAN-YVES AUDIBERT. "Efficient and scalable 4th-order match propagation". In: *Computer Vision–ACCV 2012*. Springer, 2013, pp. 460–473 (see p. 50)

[66] MARKÉTA POTŮČKOVÁ. *Image matching and its applications in photogrammetry*. PhD thesis. Aalborg Universitet, 2004 (see p. 103)

[67] RAHUL RAGURAM, ONDREJ CHUM, MARC POLLEFEYS, JIRI MATAS, and J FRAHM. USAC: a universal framework for random sample consensus. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **35**:8 (2013), pp. 2022–2038 (see p. 41)

[68] P. REN, R. C. WILSON, and E. R. HANCOCK. High order structural matching using dominant cluster analysis. In: *Image Analysis and Processing (ICIAP)*. 2011 (see pp. 43, 50)

[69] EDWARD ROSTEN and TOM DRUMMOND. "Machine learning for high-speed corner detection". In: *Computer Vision–ECCV 2006*. Springer, 2006, pp. 430–443 (see p. 35)

[70] TONI F SCHENK. *Digital photogrammetry: backgrounds, fundamentals, automatic orientation procedures*. TerraScience, 1999 (see p. 103)

[71] CORDELIA SCHMID and ANDREW ZISSERMAN. Automatic line matching across views. In: *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*. IEEE. 1997, pp. 666–671 (see p. 80)

[72] NOAH SNAVELY, STEVEN M. SEITZ, and RICHARD SZELISKI. Photo tourism: exploring photo collections in 3D. In: *SIGGRAPH Conference Proceedings*. New York, NY, USA: ACM Press, 2006, pp. 835–846. ISBN: 1-59593-364-6 (see pp. 46, 68)

[73] C. STRECHA, W. VON HANSEN, L. VAN GOOL, P. FUA, and U. THOENNESSEN. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In: *CVPR*. 2008 (see pp. 64, 65, 68, 69, 71, 77, 110–112, 119, 129, 130, 134)

[74] FRÉDÉRIC SUR, NICOLAS NOURY, and MARIE-ODILE BERGER. Computing the uncertainty of the 8 point algorithm for fundamental matrix estimation. In: *Proceedings of 19th British Machine Vision Conference (BMVC)*. 2008, pp. 96.1–96.10 (see pp. 35, 105)

[75] ZHONGWEI TANG. *High precision in camera calibration*. PhD thesis. ENS Cachan, 2012 (see p. 125)

[76] ZHONGWEI TANG, PASCAL MONASSE, and JEAN-MICHEL MOREL. On the matching precision of SIFT. In: *International Conference on Image Processing*. IEEE. 2014 (see p. 125)

[77] P. H. S. TORR and A. ZISSERMAN. MLESAC: a new robust estimator with application to estimating image geometry. In: *CVIU*, **78**:1 (2000), pp. 138–156 (see pp. 42, 49, 117, 129)

[78] PHILIP HS TORR and DAVID W MURRAY. The development and comparison of robust methods for estimating the fundamental matrix. In: *IJCV*, **24**:3 (1997), pp. 271–300 (see pp. 43, 44, 128, 129)

[79] BILL TRIGGS, PHILIP F MCLAUCHLAN, RICHARD I HARTLEY, and ANDREW W FITZGIBBON. Bundle adjustment-a modern synthesis. In: *Vision algorithms: theory and practice*. Springer, 2000, pp. 298–372 (see p. 45)

[80] A. VEDALDI and B. FULKERSON. VLFeat: An open and portable library of computer vision algorithms. In: *Int'l Conference on Multimedia*. ACM. 2010, pp. 1469–1472 (see p. 60)

[81] R GROMPONE VON GIOI, JEREMIE JAKUBOWICZ, JEAN-MICHEL MOREL, and GREGORY RANDALL. LSD: A fast line segment detector with a false detection control. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32**:4 (2010), pp. 722–732 (see p. 84)

[82] LU WANG, ULRICH NEUMANN, and SUYA YOU. Wide-baseline image matching using line signatures. In: *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE. 2009, pp. 1311–1318 (see pp. 80, 86)

[83] ZHIHENG WANG, FUCHAO WU, and ZHANYI HU. MSLD: A robust descriptor for line matching. In: *Pattern Recognition*, **42**:5 (2009), pp. 941–953 (see pp. 80, 81, 86)

[84] CHANGCHANG WU, BRIAN CLIPP, XIAOWEI LI, J-M FRAHM, and MARC POLLEFEYS. 3D model matching with Viewpoint-Invariant Patches (VIP). In: *CVPR*. 2008 (see p. 102)

[85] LUN WU, ARVIND GANESH, BOXIN SHI, YASUYUKI MATSUSHITA, YONGTIAN WANG, and YI MA. "Robust photometric stereo via low-rank matrix completion and recovery". In: *Computer Vision–ACCV 2010*. Springer, 2011, pp. 703–717 (see p. 16)

[86] WEI XU and JANE MULLIGAN. Performance evaluation of color correction approaches for automatic multi-view image and video stitching. In: *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE. 2010, pp. 263–270 (see p. 76)

[87] GUOSHEN YU and JEAN-MICHEL MOREL. ASIFT: An algorithm for fully affine invariant comparison. In: *Image Processing On Line*, (2011). DOI: http://dx.doi.org/10.5201/ipol.2011.my-asift (see p. 64)

[88] R. ZASS and A. SHASHUA. Probabilistic graph and hypergraph matching. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2008 (see pp. 50, 52, 60, 91)

[89] BERNHARD ZEISL, PIERRE FITE GEORGEL, FLORIAN SCHWEIGER, ECKEHARD G STEINBACH, NASSIR NAVAB, and GER MUNICH. Estimation of location uncertainty for scale invariant features points. In: *Proceedings of 20th British Machine Vision Conference (BMVC)*. 2009, pp. 1–12 (see p. 105)

[90] LILIAN ZHANG and REINHARD KOCH. An efficient and robust line segment matching approach based on LBD descriptor and pairwise geometric consistency. In: *J. Visual Communication and Image Representation*, **24**:7 (2013), pp. 794–805 (see pp. 80, 81, 86)