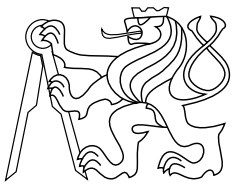
CENTER FOR
MACHINE PERCEPTION



CZECH TECHNICAL
UNIVERSITY IN PRAGUE

DIPLOMA THESIS

Image Matching for Dynamic Scenes

Filip Šrajer

filip@srajer.eu

May 26, 2016

Available at
http://cmp.felk.cvut.cz/~srajefil/theses/filip-srajer-diploma-thesis.pdf

Thesis Advisor: Ing. Tomáš Pajdla, Ph.D.

This work has been supported by RVO13000 – Conceptual development of research organization.

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

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics

DIPLOMA THESIS ASSIGNMENT

Student: Bc. Filip Šrajer

Study programme: Open Informatics

Specialisation: Computer Vision and Image Processing

Title of Diploma Thesis: Image Matching for Dynamic Scenes

Guidelines:

1. Review the state of the art in image matching for dynamic scenes related to stereo matching and to structure from motion [1-13].
2. Design a method for image matching suitable for dynamic scenes with multiple moving objects.
3. Implement the method and demonstrate it as a part of a structure from motion pipeline on relevant data sets from [1-13].

Bibliography/Sources:

[1] R. Vidal and R. Hartley: Three-View Multibody Structure from Motion. IEEE Transactions on Pattern Analysis and Machine Intelligence, volume 30, number 2, pages 214 - 227, 2008.
[2] S. Rao, R. Tron, R. Vidal, and Y. Ma: Motion Segmentation in the Presence of Outlying, Incomplete, or Corrupted Trajectories. IEEE Transactions on Pattern Analysis and Machine Intelligence, 32(10):1832-1845, 2010.
[3] http://www.vision.jhu.edu/motion.php#results, http://www.vision.jhu.edu/data/hopkins155/
[4] O.Chum and J. Matas: Homography Estimation from Correspondences of Local Elliptical Features, ICPR 2012.
[5] J. Pritts, O. Chum, and J. Matas: Detection, Rectification and Segmentation of Coplanar Repeated Patterns, CVPR 2014.
[6] J. Pritts, O. Chum, and J. Matas: Approximate Models for Fast and Accurate Epipolar Geometry Estimation, IVCNZ 2013.
[7] D. Mishkin, M. Perdoch, J. Matas: Two-View Matching with View Synthesis Revisited. http://arxiv.org/abs/1306.3855
[8] D. Mishkin, J. Matas, M. Perdoch, K. Lenc: WxBS: Wide Baseline Stereo Generalizations. http://arxiv.org/pdf/1504.06603.pdf
[9] D. Mishkin, J. Matas, M. Perdoch: MODS: Fast and Robust Method for Two-View Matching. http://arxiv.org/abs/1503.02619
[10] D. Mishkin, M. Perdoch, J. Matas: Place Recognition with WxBS Retrieval. CVPR 2015 Workshop on Visual Place Recognition in Changing. Environments. http://cmp.felk.cvut.cz/~mishkdmy/papers/mishkin-place-recognition.pdf
[11] B. Zeisl, T. Sattler, M. Pollefeys: Camera Pose Voting for Large-Scale Image-Based Localization. ICCV 2015.
[12] C. Sweeney, T. Sattler, T. Höllerer, M. Turk, M. Pollefeys: Optimizing the Viewing Graph for Structure-from-Motion. ICCV 2015.
[13] A. Cohen, T. Sattler, M. Pollefeys: Merging the Unmatchable: Stitching Visually Disconnected SfM Models. ICCV 2015.

Diploma Thesis Supervisor: Ing. Tomáš Pajdla, Ph.D.

Valid until: the end of the summer semester of academic year 2016/2017

L.S.

prof. Dr. Ing. Jan Kybic
Head of Department

prof. Ing. Pavel Ripka, CSc.
Dean

Prague, December 18, 2015

Acknowledgements

I would like to express my gratitude to my thesis adviser Tomáš Pajdla for his valuable guidance and advice during my study. My thanks also goes to my family for all their support.

Author statement

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Prohlášení autora práce

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne ....

Podpis autora práce

Abstract

Image matching is an important intermediate step for computer vision tasks such as 3D reconstruction, image retrieval, and image stitching. We argue that it is important to consider dynamic scenes with different motions because the real world is dynamic. We propose a fast greedy approach for detection of multiple homographies and their subsequent grouping into motion groups. For this purpose, we utilize two properties valid for two homographies that move together; their composition is a planar homology and epipolar geometry can be fitted well to their inliers. We show that our approach performs just as well or better than sequential application of RANSAC. Furthermore, we propose to fuse the groupings of matches available for every image pair into global grouping of n-view matches when more than two views are available. Next, we supply the groups of n-view matches to an incremental structure-from-motion pipeline to compute sparse 3D reconstructions independently. The pipeline is implemented using our library which we have designed to be reliable, easy to extend, and efficient. The approach for reconstruction of dynamic scenes is evaluated on a dataset with three moving objects.

Abstrakt

Hledání korespondencí mezi obrázky je důležitý krok v zaměřeních počítačového vidění jako jsou 3D rekonstrukce, hledání podobných obrázků a slepování obrázků. Vyzdvihujeme, že je důležité brát v potaz dynamické scény s různými pohyby, protože skutečný svět je dynamický. Navrhujeme rychlou hladovou metodu pro detekování několika homografií a jejich následné slučování do skupin podle pohybů. K tomuto účelu využíváme dvě vlastnosti dvou spolu se pohybujících homografií: jejich složení je planární homologie a z jejich korespondencí lze dobře vypočítat epipolární geometrii. Ukazujeme, že naše metoda funguje stejně dobře nebo lépe než sekvenční aplikování metody RANSAC. Dále navrhujeme sloučit znalost o skupinách korespondencí, které máme pro páry obrázků, do globálních skupin korespondencí, pokud máme k dispozici více než dva snímky. Tyto globání skupiny dále vložíme do klasického systému na rekonstrukci poloh kamer a bodů, abychom získali řídké rekonstrukce. Rekonstrukční systém je implementován pomocí knihovny, kterou jsme navrhli, aby byla spolehlivá, snadno rozšiřitelná a rychlá. Metodu na rekonstrukci ověřujeme na snímcích zachycující tři pohybující se objekty.

Contents

1 Introduction

In computer vision, image matching is an important task which is employed as an intermediate step in many approaches. To name a few, it is used in 3D reconstruction, image retrieval or image stitching. Image matching basically finds relations between images.

A typical approach to image matching works in three steps. First, features (also called keypoints or regions of interest) such as SIFT [45] and MSER [49] are detected in every image separately. Features can be any points or regions but they should be repeatably detectable, *i.e.* it should be possible to detect them under changing conditions such as changing illumination, rotation, scale, *etc.* Second, features of pairs of images are matched in order to find corresponding ones. This step typically utilizes nearest neighbor search [55] based on local appearance of features which can result in a set of matches corrupted by a large number of outliers. Therefore, the third step aims to geometrically verify the matches in order to filter out outliers. The geometric verification is often done by estimating a homography or an epipolar geometry with the largest amount of inliers and discarding all outliers.

Accepting a hypothesis with the largest amount of inliers, however, models only the largest motion in the scene. That is valid and works very well for static scenes. Nevertheless, the real world is dynamic and objects often move with different motions. Consider a city with moving cars, buses, and people, for instance.

In the image retrieval task, it is common to query similar images using the features of the query image and do query expansion step to get better results. The query expansion retrieves additional features related to the query image by matching features of the query image to the features of similar images and again queries for other similar images. Assuming a static scene in a dynamic one can result in ignoring motions that have minority of feature matches.

In the 3D reconstruction task, feature matching is applied in order to find camera parameters. If the goal is to reconstruct a single object, it could prove problematic when another differently moving object or background would be present in the images. The object that is not of interest could generate many features and become the major motion. It could also generate about the same number of features as the object of interest. Then the two objects would compete for being dominant in different image pairs. Furthermore, consider a moving robot which needs to understand the 3D scene in order to navigate. As the real world is dynamic, it needs to avoid differently moving obstacles such as people or cars and hence has to detect them.

1.1 Contributions

This thesis presents four main contributions.

First, we design a fast greedy approach for detection of multiple homographies in an image pair. It can be used for detecting different motions which are modellable by a homography or for detecting multiple planes of one motion, for example. We use a classical approach for computing tentative matches and focus on homography estimation. We propose to grow homographies from similarity transformations induced

Figure 1.1 Feature matches as detected by our approach from Sec. 4.1.2. Different colors correspond to different motions. The images are from the Hopkins155dataset [80].

by feature matches similarly to [81, 63]. Our novelty is twofold. We utilize the algorithm for the detection of multiple homographies. Furthermore, we speed it up by not trying all possible hypotheses but robustly refining the best found hypothesis instead, which gives results of the same quality.

Second, the above described approach is applied to scenes with full rigid motions which extracts multiple homograhpies where some may belong to the same motion. We compute a score for every pair of homograhies in order to determine which ones should be merged into the same group. The score is based on two ideas. First, a composition of two homographies belonging to the same motion should be a planar homology [31]. Second, fundamental matrix computed from inliers of two homograhies should model all these inliers and possibly some other matches if the two homograhies arose from a single motion. The main novelty is the application of the planar-homology property for dynamic scenes.

Third, we have implemented a new computer vision C++ library intended to be used for implementing structure-from-motion systems. It is designed to be reliable, easy to extend, and fast. It is based on the linear algebra library Eigen [28] so that writing matrix expressions would be easy. Our library implements feature matching, unified RANSAC framework, several camera pose solvers, two-view and n-view matches handling, 3D points triangulation, current state saving and loading, image similarity computation, and various auxiliary functions. It indirectly supports also feature detection and non-linear optimization through external libraries.

Four, we use our library to implement an incremental structure-from-motion pipeline utilizing our geometric verification approach. The contribution is the approach for fusing the information from two-view matching phase which provides grouping of matches per image pair. We propose to connect two-view matches into n-view matches and group them based on the two-view groups. Then, we run a classical structure from motion on the individual n-view match groups independently.

1.2 Thesis structure

Chapter 2 first establishes the notation and basic concepts used in the thesis. Chapter 3 then reviews related work in image matching, motion segmentation and 3D reconstruction. Next, Chapter 4 presets the proposed approaches for feature matching and dynamic scene reconstruction. Chapter 5 follows with the description of the implemented library. Furthermore, Chapter 6 evaluates the performance of the proposed approaches and Chapter 7 identifies their limitations. Finally, Chapter 8 summarizes the thesis.

2 Notation and Concepts

2.1 Notation

$a, b, \ldots, \alpha, \beta, \ldots$	scalars
$\boldsymbol{a}, \boldsymbol{b}, \ldots$	column vectors
$\mathtt{A}, \mathtt{B}, \ldots$	matrices
$a, b, \ldots, A, B, \ldots$	functions
$\mathcal{A}, \mathcal{B}, \ldots$	sets
v	function vectorizing a matrix in row-wise order
$\boldsymbol{x} \leftrightarrow \boldsymbol{x}'$	$\boldsymbol{x}$ corresponds to $\boldsymbol{x}'$
$\lvert \cdot \rvert$	number of elements in a set or absolute value of a scalar
$\lvert\lvert \cdot \rvert\rvert$	Euclidean norm
$\mathtt{I}$	identity matrix

2.2 Homography

This section briefly introduces homography and some of its specializations in order to make the reader acquainted with the concept. A homography [31] is a mapping represented by a full-rank matrix $\mathtt{H} \in \mathbb{R}^{3\times3}$. Note that its dimensions could be different but this thesis needs to transform points of the real projective plane only. A homography is defined up to scale and therefore all matrices $\gamma\mathtt{H}$ for $\gamma \in \mathbb{R} \setminus \{0\}$ represent the same homography. A point $\boldsymbol{x} \in \mathbb{R}^3$ is mapped to $\boldsymbol{x}' \in \mathbb{R}^3$ as

$$\mathtt{H}\boldsymbol{x} = \lambda\boldsymbol{x}'. \tag{2.1}$$

In case it is required to map image points $\boldsymbol{u}, \boldsymbol{u}' \in \mathbb{R}^2$, it is possible to make use of their homogeoneus representation, *i.e.* set the third coordinate to one. A homography mapping image points between two images can represent two situations. First, all the image points are modellable by the homography in case that the two images correspond to the same camera center. Second, if the images correspond to different camera centers, then the homography maps projections of a single world plane in the first image to corresponding projections in the second image.

Having a homography $\mathtt{H}$ and matching points $\boldsymbol{u}, \boldsymbol{u}'$, we define an error function which we use for distinguishing inliers from outliers by thresholding.

$$e_{\mathtt{H}}(\boldsymbol{u}, \boldsymbol{u}', \mathtt{H}) = \left|\left| \boldsymbol{u}' - \frac{(\mathtt{H}\bar{\boldsymbol{u}})_{1..2}}{(\mathtt{H}\bar{\boldsymbol{u}})_3} \right|\right| \quad \bar{\boldsymbol{u}} = \begin{bmatrix} \boldsymbol{u} \\ 1 \end{bmatrix}. \tag{2.2}$$

Notice that this error computes the pixel distance of a point projected by a homography from its corresponding point.

Next, consider a reversed scenario, where image-point matches are known and homography unknown. Having a match

$$\begin{bmatrix} u \\ v \end{bmatrix} \leftrightarrow \begin{bmatrix} u' \\ v' \end{bmatrix}, \tag{2.3}$$

Eq. (2.1) can be rewritten as

$$\begin{bmatrix} u & v & 1 & 0 & 0 & 0 & -u'u & -u'v & -u' \\ 0 & 0 & 0 & u & v & 1 & -v'u & -v'v & -v' \end{bmatrix} v(\mathtt{H}) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{2.4}$$

where v is a vectorizing function. That makes two constraints for the eight degrees of freedom (nine numbers in $\mathtt{H}$ and minus one for scale). It is hence possible to stack equations for four points to get an exact solution. More than four would make an overdetermined system of equations. Both cases can be solved using SVD.

2.2.1 Affinity

An affinity, also called an affine transformation, is a homography specialization. It represents a non-singular linear transformation followed by a translation and its matrix form looks like

$$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ 0 & 0 & 1 \end{bmatrix} \quad h_{ij} \in \mathbb{R} \quad i \in \{1,2\}, j \in \{1,2,3\}. \tag{2.5}$$

Similarly to full homography, Eq. (2.1) can be rewritten as

$$\begin{bmatrix} u & v & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & u & v & 1 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \end{bmatrix} = \begin{bmatrix} u' \\ v' \end{bmatrix} \tag{2.6}$$

which has the form $\mathtt{A}\boldsymbol{x} = \boldsymbol{b}$ and can be solved using least squares. Three matches are needed to compute an exact transformation (six degrees of freedom for six unknowns) and more than three to create an overdetermined system.

2.2.2 Similarity

A similarity is a further affinity specialization. It is a combination of translation, rotation and scaling. Hence, it has the following form

$$\begin{bmatrix} s\cos o & -s\sin o & t_1 \\ s\sin o & s\cos o & t_2 \\ 0 & 0 & 1 \end{bmatrix} \quad s, o, t_1, t_2 \in \mathbb{R} \tag{2.7}$$

where s represents scale, o orientation and $\boldsymbol{t}$ translation. We do not need to compute a similarity from image matches and thus we do not derive an estimation procedure.

2.2.3 Planar homology

We define the planar homology as in Hartley and Zisserman [31]. It is a homography that has a line of fixed points (called the *axis*) and a fixed point not on the line (called the *vertex*). A fixed point is a point that is not changed by a transformation. Algebraically, a planar homology has two equal and one distinct eigenvalues. The axis is the line through the two eigenvectors corresponding to the same eigenvalues and the vertex is

the eigenvector corresponding to the distinct eigenvalue. Assuming that the eigenvalues are up to a common scale factor $\{\mu, 1, 1\}$, then a homology can be parameterized as

$$\mathtt{I} + (\mu - 1)\frac{\boldsymbol{v}\boldsymbol{a}^\top}{\boldsymbol{v}^\top\boldsymbol{a}} \tag{2.8}$$

where $\boldsymbol{v}$ is the vertex and $\boldsymbol{a}$ is the axis.

2.3 Epipolar geometry

This section introduces epipolar constraint and fundamental matrix [31] in a necessary depth for understanding this thesis only. The epipolar constraint is valid for all image points that belong to one motion. Let us define fundamental matrix in order to express the epipolar constraint. A valid fundamental matrix is a rank-two matrix $\mathtt{F} \in \mathbb{R}^{3\times3}$, which relates matching points $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^3$ as

$$\boldsymbol{x}'^\top \mathtt{F}\boldsymbol{x} = 0. \tag{2.9}$$

The geometric interpretation is that $\mathtt{F}$ maps the point $\boldsymbol{x}$ from the first image to a line in the second image going through the correspoinding point $\boldsymbol{x}'$. Notice that multiplying $\mathtt{F}$ by a real number does not change the validity of the constraint. Therefore, fundamental matrix is, similarly to homography, defined up to scale.

To examine $\mathtt{F}$ more deeply, consider the basis vectors of the left null space $\boldsymbol{e}'$ and the right null space $\boldsymbol{e}$. Points $\boldsymbol{e}$ and $\boldsymbol{e}'$ are called *epipoles*. They turn out to be projections of the camera centers in the other images, *i.e.* $\boldsymbol{e}$ is the projection of the second camera center in the first image. The epipole $\boldsymbol{e}'$ has an interesting property; every line mapped by $\mathtt{F}$ goes through it. Additionally, it can be verified that if $\mathtt{F}$ represents the relation of the first and second images, then $\mathtt{F}^\top$ relates the second and the first image, *i.e.* in the reverse order.

Having $\mathtt{F}$, we use the Sampson distance [31] throughout the thesis as an error measure of matches. The Sampson distance is first-order geometric error approximation and is defined as

$$e_F(\boldsymbol{x}, \boldsymbol{x}', \mathtt{F}) = \frac{\boldsymbol{x}'^\top \mathtt{F}\boldsymbol{x}}{\sqrt{(\mathtt{F}\boldsymbol{x})_1^2 + (\mathtt{F}\boldsymbol{x})_2^2 + (\mathtt{F}^\top\boldsymbol{x}')_1^2 + (\mathtt{F}^\top\boldsymbol{x}')_2^2}}. \tag{2.10}$$

The advatage of the Sampson distance is that it includes only parameters of $\mathtt{F}$. Note that the geometric error would need the 3D point coordinates to be known.

Moreover, we describe an approach for fundamental matrix estimation. Assuming that matching image points are known

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} \leftrightarrow \begin{bmatrix} u' \\ v' \\ w' \end{bmatrix}, \tag{2.11}$$

the epipolar constraint can be rewritten as

$$\begin{bmatrix} uu' & vu' & wu' & uv' & vv' & wv' & uw' & vw' & ww' \end{bmatrix} v(\mathtt{F}) = 0 \tag{2.12}$$

which gives one constraint on the fundamental matrix. Since fundamental matrix has seven degrees of freedom (nine numbers in $\mathtt{F}$, minus one for scale, and minus one for rank two), seven matches are needed to compute an exact solution.

Having seven matches, Eq. (2.12) for every match can be stacked on top of each other and two dimensional null space computed using SVD. Let us denote the null space as $\mathtt{F}_1$ and $\mathtt{F}_2$. Finding a solution to

$$\det\left(\alpha\mathtt{F}_1 + (1-\alpha)\mathtt{F}_2\right) = 0, \tag{2.13}$$

gives up to three real fundamental matrices which have to be further verified by computing their inlier sets. The described estimation process is called the seven-point algorithm.

Furthermore, we describe the eight-point algorithm for estimating a fundamental matrix from more than seven matches [31]. It is needed by our matches verification approach as it needs to fit a fundamental matrix to many matches. The idea of the eight-point algorithm is stacking Eq. (2.12) for more than seven matches, using SVD to find a rank-three matrix and then using SVD again but on the found matrix to find the closest rank-two matrix. In order to make the algorithm numerically more stable, the points in individual images are first normalized by similarities transforming them so that the means would be in the origin and the standard deviations one in both coordinate directions. Hence, the fundamental matrix is estimated from the normalized points and then multiplied by the similarities so that it would relate the original points and not the normalized ones. Additionally, we refine the resulting matrix using non-linear optimization which takes advantage of a non-minimal parameterization of the fundamental matrix

$$\mathtt{F} = [\boldsymbol{v}]_\times \mathtt{H} \tag{2.14}$$

where $\mathtt{H} \in \mathbb{R}^{3\times3}$ is of rank tree and $\boldsymbol{v} \in \mathbb{R}^3$. ‘

3 State of the Art

This chapter first reviews approaches for feature matching including those assuming a static scene. Secondly, an overview of state of the art in motion segmentation is given. Finally, a survey of well-known and recent structure-from-motion works follows.

3.1 Feature matching

Most of the existing works, including our feature matching approaches, follow a similar scheme. Features, such as SIFT [45], MSER [49] or Hessian-Affine [50], are firstly detected in images. Then, given a pair of images, the nearest neighbors to every feature in the first image are found in the second image in the descriptor space (see Sec. 4.1). To filter the worst outliers, Lowe ratio [45], *i.e.* the ratio of the distance to the nearest neighbor over the distance to the second nearest neighbor, is computed and thresholded. Only the distance between descriptors would not be discriminative enough. Nevertheless, matches are not guaranteed to be correct even after this filter. They have to be further verified geometrically, which typically consists of fitting a homography for planar scenes or epipolar geometry for 3D scenes to the matches and keeping only the inliers. In addition, approaches assuming a dynamic scene fit multiple homographies or epipolar geometries in order to model all motions.

3.1.1 Tentative matches

Some works aim at improving and enriching tentative matches, *i.e.* matches not yet geometrically verified. Note that we focus on geometric verification and all the approaches in this section could thus be naturally applied to improve our results. RootSIFT [4] is basically a normalization of a descriptor which ensures that computing Euclidean distance between two descriptors actually computes Hellinger distance. Fragoso and Turk [25] employ an alternative score to Lowe ratio.

Recently, [53, 61, 44, 51] presented matching schemes for enriching features. They generate multiple synthetic views of an input image, which simulate different viewpoints, and detect features on them. This approach is able to cope with significant perspective effects on image patches. Mishkin *et al.* [51], in addition, introduce a new ratio score, which takes the distance of the first geometrically inconsistent nearest neighbor instead of the second one, to deal with the increased number of similar image features located in the same place.

3.1.2 RANSAC

A lot of geometric verification works is from the RANSAC family. The core idea of using RANSAC [23] for geometric verification is the computation of homography or epipolar geometry from random minimal samples and keeping the hypothesis that has the largest support. This ensures robust estimation.

Raguram *et al.* [66] give a comprehensive overview of the most interesting variations to the 'vanilla' RANSAC [23] and themselves propose USAC: Universal Framework for RANSAC. Their USAC implementation leverages various state-of-the-art algorithms for

the individual modules of the framework. To name a few, USAC, like PROSAC [15], takes advantage of Lowe ratio to score the quality of every match and prefers to use the better ones first. Next, it avoids degenerate epipolar geometries following the approach of DEGENSAC [17]. Moreover, it chooses the best hypothesis using SPRT test [48] to speed up inlier counting. In addition, it refines the hypotheses computed from minimal samples using all their inliers as proposed in LO-PROSAC [16].

There are further indirect adaptations to RANSAC. [62, 77] sample non-minimal samples to reduce the amount of noise in a sample. Chin *et al.* [13] reorder the matches based on the residual which enforces minimal samples to be from the same model. Wong *et al.* [88] accumulate the residual to different hypotheses to dynamically identify outliers and bias the sampling towards the discovery of multiple models in the data. Raguram and Frahm [67] also use residuals but to design a method which does not need error threshold. BEEM [27], a RANSAC resembling algorithm, focuses on managing matches with low percentage of inliers and avoiding degenerate epipolar geometries. Unlike RANSAC, it exploits available prior knowledge and changes the sampling strategy to arrive to better hypotheses. Nonetheless, Brahmachari and Sarkar [9] propose a Monte Carlo approach for estimating epipolar geometry and they claim to be better than BEEM.

Transformation from one match

Another group of works related to the RANSAC specializes on generating a hypothesis from one feature match. Consider that every feature has a location and a shape, an oriented disc (SIFT) or an ellipse (MSER), for instance. It is possible to compute a transformation normalizing the image patch to a unit circle in the origin; a similarity for SIFT and an affine transformation for MSER. Having a matching pair of features, it is possible to combine the transformations of both features to get a transformation from one image to the other. Note that the set of all possible samples is very small since the minimal-sample size is one. This enables to remove the randomness by trying to generate a hypothesis from every sample.

Philbin *et al.* [63] compute a hypothesis from a single match, find its inliers, refine it into an affine transformation using the inliers, and again find inliers. They do this this for every match and keep the largest set of inliers. Vedaldi and Zisserman [81] take a similar approach but refine the transformation into a full homography. We build on [81] and significantly speed up the approach while keeping the same quality of results. Additionally, we apply the approach on non-planar scenes and propose how to merge homographies into epipolar geometries. [3, 65] use two pairs of matching elliptical regions to compute an affine fundamental matrix. Pritts *et al.* [65], in addition, do a local optimization step in every step to refine the affine fundamental matrix into a full fundamental matrix.

3.1.3 Hough transform

In contrast to RANSAC, Hough transform [7] based approaches use all matches to vote for transformations that are consistent with them and finally select the one with the highest support. This enables handling higher outlier contamination. Nonetheless, it has high memory requirements as all possible transformations have to be represented, which is done by discretizing space of parameters. Hough transform was utilized already by Lowe [45] to verify SIFT matches. Hough Pyramid Matching [6] uses Hough-like scheme to recursively group matches into motion groups. Chen *et al.* [11] first group

matches to objects using co-segmentation to subsequently compute per-object homographies using Hough transform.

3.1.4 Multiple models

All the previous works focus on extracting a hypothesis with the largest support. The following ones consider that there can be multiple models in a scene. That could be multiple planes or multiple motions. [8, 76] use RANSAC to greedily extract multiple hypotheses, *i.e.* find the best hypothesis, remove its inliers from the set of all matches and repeat until there is enough matches. In addition, Bhat *et al.* [8] handle degenerate situations by estimating both homography and fundamental matrix and keeping the homography if it explains most of the fundamental matrix inliers. On the contrary, Zuliani *et al.* [96] formulate the MultiRANSAC algorithm which estimates multiple transformations in every round.

Isack and Boykov [33] argue that greedily extracting a hypothesis with the most inliers is not the best strategy and design an energy-based optimal labeling approach called PEARL. J-linkage [78], an agglomerative clustering method, adresses fitting multiple models to noisy data with outliers. It does not need the number of clusters to be known and reports better results than greedy sequential RANSAC application. Fouhey *et al.* [24] detect multiple planes in a scene using J-linkage and subsequently jointly refines the estimated models. T-linkage [46] is an improvement of J-linkage that handles outliers and noise by replacing binary analysis by continuous generalization.

Similarly to [8, 76], our approach for detecting multiple homographies is also greedy. Nevertheless, to the best of our knowledge, no other work detects multiple homographies in order to merge them into motion groups. Importantly, note that our approach also addresses degeneracies since a motion group can be defined either by a homography or an epipolar geometry.

3.2 Motion segmentation

This section gives an overview of recent motion segmentation approaches. Motion segmentation is a discipline of grouping matches into motion groups. Even though it is not called directly image matching, it is related to it. Notice the similarity to fitting multiple models in Sec. 3.1.4. Nevertheless, motion segmentation methods are different since they often assume multiple views and segment n-view matches, also called tracks, (see Sec. 5.3.3). It is typically also assumed that the number of motions is known, there are no outliers or even that there are no missing data (*i.e.* all the matches span over all frames). The following subsections roughly group the motion segmentation methods based on what approach they rely on, *i.e.* geometry, matrix factorization, optical flow, and subspace clustering.

3.2.1 Geometric

Works based on geometry significantly differ in the number of frames they consider. In general, multi-view approaches are more accurate than two-view ones. However, their accuracy rapidly decreases as the number of frames gets too low.

The following methods follow different ideas but they all need only two views of a scene. Vidal *et al.* [83] generalize epipolar constraint into multibody epipolar constraint, which they estimate using Generalized Principal Component Analysis (GPCA). Rao *et al.* [69] propose to join homography and epipolar constraints into a hybrid perspective

constraint to handle planar and 3D structures. Given the number of motions K and feature matches in two-views, they fit a set of $2K$-degree polynomials to partition matches into motion groups. Poling and Lerman [64] present a concept of global dimension, which they minimize with their fast projected gradient algorithm. Jung *et al.* [38] introduce a randomized voting scheme, where the grouping is iteratively refined in voting and labeling steps. The core of the voting step is a process of estimating fundamental matrices from samples and then giving bigger votes to matches with smaller residuals. Rubino *et al.* [71] make use of an object detector to guide the motion segmentation. Our proposed geometric verification approach compares to the two-view methods as it needs two views. Importantly, it does not need the number of motions to be known. Also, it handles both planar and 3D structures.

Vidal and Hartley [82] take one more view of the scene and propose multibody trilinear constraint, which is a natural generalization of the trilinear constraint. They estimate the associated multibody trifocal tensor using GPCA to subsequently run an iterative refinement algorithm alternating between trifocal tensor computation and segmentation of matches.

Motion segmenters capable of utilizing multiple views include [43, 95]. Li *et al.* [43] utilize the two-view epipolar constraint and then combine point correspondence information across multiple views. Next, they adopt an over-segment and merge approach, which does not need the number of motions as input. Similarly, we fuse two-view matches into n-view matches. Nevertheless, we are capable of handling also planar motions as we do not use only epipolar constraints but homographies too. Zografos and Nordberg [95] design a motion segmentation method, which, in its core, makes use of the theory of linear combination of views. In contract, Jacquet *et al.* [34] analyze known relative motions between two parts in order to identify the type of articulated motion.

3.2.2 Matrix factorization

Matrix factorization approaches typically construct a large matrix from n-view matches and decompose it into a product of two matrices. One of those matrices then reveals grouping of the data.

Cheriyadat and Radke [12] decompose the data matrix into different motion components and their non-negative weights which can be used to segment the data. They do not require the matches to span all frames. Mo and Draper [52] propose to deal with missing data using a fast method based on Semi-Nonnegative Matrix Factorization. Favaro *et al.* [21] can cope with noise but not with missing data. They pose the problem as a rank minimization task and decompose the data matrix into a clean low-rank dictionary and a matrix of noise/outliers. Vidal *et al.* [84] project the point trajectories onto a 5-dimensional subspace using PowerFactorization and fit multiple linear subspaces representing different motions using GPCA.

3.2.3 Optical flow

Methods based on optical flow can be used for videos captured by a moving camera. Klappstein *et al.* [39] estimate optical flow by analyzing motion patterns. They formulate the problem as an energy minimization and solve it via a min-cut-max-flow algorithm. Ochs *et al.* [58] exploit long term point trajectories and focus on areas in the image, where optical flow estimation works best to get reliable tracks. Narayana *et al.* [56] address the problem that the motion can produce different optical flow in differ-

ent depths. They propose to use the optical flow orientations, which are independent of depth.

3.2.4 Subspace clustering

Consider that n-view matches belonging to different motions lie on different subspaces. Motion segmentation approaches based on subspace clustering group together n-view matches that belong to similar subspaces.

Elhamifar and Vidal [20] propose a method based on sparse representation to cluster data drawn from multiple low-dimensional subspaces embedded in a high-dimensional space. They aim to find the sparsest representation of every n-view match by using L1-norm optimization. Rao *et al.* [68] find the segmentation of the n-view matches via minimization of coding length that is needed to represent the data by a mixture of Gaussians. To solve this problem efficiently, they first find a suboptimal solution by considering every match as a group. Then, they merge the groups to minimize the coding lenght. Thus, the number of motions does not have to be known. Zhang *et al.* [94] introduce the application of correntropy to robustify subspace clustering in order to take into account real-life corruptions of the data. To solve the proposed robust subspace clustering, they employ half-quadratic minimization.

3.3 3D reconstruction

3D reconstruction is an active research area and this section provides only a brief survey. The main body of work assumes that the scene is static. Such approaches would extract the motion with the highest support and discard the others in dynamic scenes. We first briefly review the state of the art of approaches for static scenes and then give an overview of 3D reconstruction for dynamic scenes.

3.3.1 Static

A typical structure-from-motion pipeline would process in two main steps. Firstly, it would detect n-view matches (see Sec. 3.1 and Sec. 5.3.3), where one n-view match represents a projection of one 3D point into different images. Next, the pipeline recovers the coordinates of the 3D points corresponding to the n-view matches and camera parameters corresponding to the images.

The existing works can be divided in two groups based on the scheme of recovering the camera parameters and 3D points. There are so called global [18, 5, 10, 37, 54, 87, 75, 19] and incremental [74, 1, 26, 42, 91, 32] methods.

Incremental The core idea of this technique is the initialization of a 3D model from a few cameras (typically a pair of cameras) and subsequent iterative addition of other cameras. A typical scheme would start by using RANSAC to estimate relative pose of a calibrated pair of images with many matches in common to initialize the reconstruction. Then the camera seeing the most n-view matches that have been already reconstructed into 3D points would be added. The adding process finds camera parameters via absolute pose solver fed by image-to-scene matches and reconstructs all new 3D points visible by the new camera and another already recovered camera. Bundle adjustment [79], non-linear optimization of all camera parameters and 3D points, is ran after every camera addition.

We propose to use a classical incremental structure from motion on a dynamic scene by pre-grouping n-view matches and feeding the n-view match groups to the pipeline independently.

Global Global methods consider all available relative poses between pairs of images to estimate all camera poses in a single step. The cameras are considered to be calibrated and hence it is possible to estimate their global orientations [10, 29, 30, 47] and positions [5, 37, 54, 87] simultaneously by averaging. To refine the model, a final bundle adjustment [79] step is ran on the whole scene. This contrasts incremental approaches as they run bundle adjustment of the whole scene in every iteration. Additionally, it is also rewarding to globally optimize all the relative poses in the beginning [75].

3.3.2 Dynamic

Despite the advances in static 3D reconstruction and motion segmentation, there is not many 3D reconstruction systems taking into account a dynamic scene. Most of the researchers in this area consider availability of a video sequence [59, 22, 93, 70, 72]. On the other hand, our approach can handle unordered images. Ozden *et al.* [59] address important issues of dynamic reconstruction mainly theoretically and then propose an exemplary framework. For instance, they tackle the cases where the number of independent motions changes as some objects can leave the field of view or can stop moving (*e.g.* a car is parking). Jacquet *et al.* [35] improve the reconstruction of multiple rigid bodies by making use of non-intersection constraints which force the rigid bodies not to intersect at any point in time.

[22, 93, 72, 70] present systems for joint estimation of motion segmentation and 3D reconstruction. They apply a hill climbing, EM-like, approach and alternate between fixing and updating certain groups of parameters (*e.g.* 3D points, group labelling and motion models [72, 22]). Note that some of the works do not assume a rigid motion. Fayad *et al.* [22] put focus on articulated motion and demonstrate how their system works on full human body motion sequences. Russell *et al.* [72] go a step further and make no assumption on the motion, *i.e.* the motion can be non-rigid. They show experiments with videos downloaded from the internet.

In contrast to the aforementioned, Wang *et al.* [86] propose a method, which requires only two views of the scene. They use SIFT features boosted by synthetic views (see Sec. 3.1.1) and match them via classic nearest neighbor search. Next, they overestimate the number of motions by greedily fitting fundamental matrices with RANSAC. Finally, they run an optimization of the whole problem enforcing neighboring features to belong to the same motion group and preferring less motion groups.

4 The Proposed Approach

This chapter firstly introduces our feature matching approach for dynamic scenes. Then, it presents our incremental structure-from-motion pipeline which utilizes the matching approach in order to handle dynamic scenes.

4.1 Image matching for dynamic scenes

First, our image matching approach, which assumes that motions can be modelled by homographies, is detailed in Sec. 4.1.1. It is possible to make the assumption for planar objects or planar motions. The assumption can be made for matching subsequent frames of videos if the frame rate is sufficient to make a motion seem planar, for example. Secondly, we remove the assumption and extend the method for rigid motions that cannot be modelled by homographies. This extension is described in Sec. 4.1.2.

4.1.1 Growing homographies

Consider the following task we want to solve in this subsection. We are given two images I, I' as input with an unknown number of motions $\bar{p}$ which are all modellable by a homography. Hence, we are looking for homographies

$$\mathtt{H}_i \in \mathbb{R}^{3\times 3} \quad i \in \{1, 2, \ldots, \bar{p}\} \tag{4.1}$$

transforming points from I to I'.

To give an overview, our approach detects features, finds tentative matches of features between images and geometrically verifies them. The geometric verification greedily grows multiple homographies from individual matches.

Features

We propose to utilize SIFT features [45] which are widely used (*e.g.* by [8, 53, 76, 86, 1, 74, 5, 18]). A SIFT feature is made of coordinates $\boldsymbol{x} \in \mathbb{R}^2$, scale $s \in \mathbb{R}$ and orientation $o \in \mathbb{R}$. Fig. 4.1 visualizes a few SIFT features to enable easier understanding. For convenience of notation, we write

$$\boldsymbol{f} = \begin{bmatrix} \boldsymbol{x} \\ s \\ o \end{bmatrix} \in \mathbb{R}^4. \tag{4.2}$$

and denote a set of features by $\mathcal{F}$.

Tentative matches

Assume that we have detected n features $\mathcal{F}$ in I and m features $\mathcal{F}'$ in I'. Next, we compute SIFT descriptors [45] $\mathcal{D}$ and $\mathcal{D}'$ corresponding to the features. A SIFT descriptor $\boldsymbol{d} \in \mathbb{R}^{128}$ is a vector encoding local appearance of an image path. Hence, descriptors can be used to find similar-looking features. We use this property to find

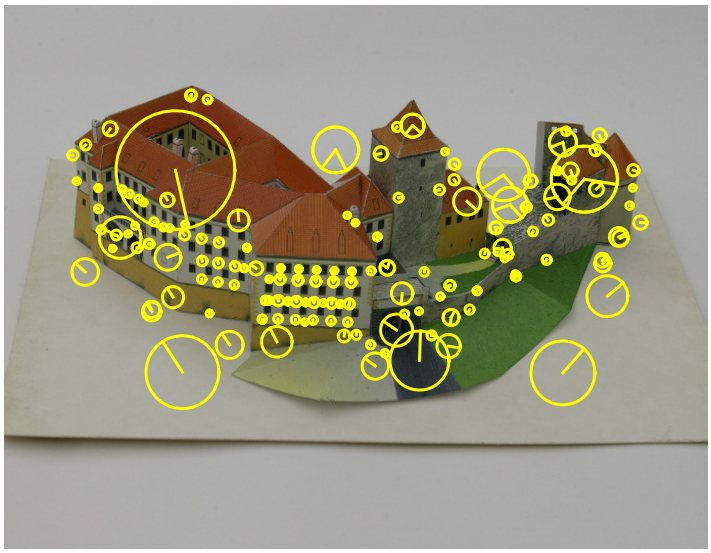


Figure 4.1 A visualization of SIFT features [45]. The yellow circles represent individual features. The circle centers correspond to feature coordinates, the circle radii to scales and the lines going from the circle centers to orientations. This example shows only features of a scale higher than five to enable a nice visualization.

matching features between images by searching nearest neighbor in $\mathcal{D}'$ in descriptor space for every descriptor in $\mathcal{D}$. More formally, a matching $\mathcal{M}_0$ is

$$\mathcal{M}_0 = \left\{ (i,j) \;\middle|\; i \in \{1,2,\ldots,n\} \;\wedge\; j = \underset{j' \in \{1,2,\ldots,m\}}{\operatorname{argmin}} \left|\left| \boldsymbol{d}_i - \boldsymbol{d}'_{j'} \right|\right| \right\} \tag{4.3}$$

where $\boldsymbol{d}_i \in \mathcal{D}$ and $\boldsymbol{d}'_j \in \mathcal{D}'$. We call the set $\mathcal{M}_0$ *tentative initial matches*.

Unfortunately, $\mathcal{M}_0$ often contains mismatches as the nearest neighbor in descriptor space does not have to be a true correspondence and some features in I just do not have a true corresponding feature in I'. To filter out some mismatches, we compute Lowe ratio [45] for every match and keep only those with smaller ratio than a threshold. Lowe ratio is defined as the distance to the nearest neighbor over the distance to the second nearest neighbor, *i.e.* given match (i,j)

$$r(i,j) = \frac{\left|\left| \boldsymbol{d}_i - \boldsymbol{d}'_j \right|\right|}{\min\limits_{j' \in \{1,2,\ldots,m\};\; j' \neq j} \left|\left| \boldsymbol{d}_i - \boldsymbol{d}'_{j'} \right|\right|}. \tag{4.4}$$

The ratio basically measures how unique is the nearest neighbor. See Fig. 4.2 for precision-recall curve to compare discriminability of the distance to the nearest neighbor and Lowe ratio. Finally, we filter the matches

$$\mathcal{M} \leftarrow \{(i,j) \mid (i,j) \in \mathcal{M}_0 \;\wedge\; r(i,j) < t\} \tag{4.5}$$

where $t \in \mathbb{R}$ is a threshold.

We employ FLANN library [55] for all the nearest neighbor searches. In more detail, we utilize structure called randomized kd-tree forest. This structure starts by building four kd-trees over the target set, *i.e.* $\mathcal{D}'$ in our case. kd-tree is a binary tree splitting space in every node in one dimension. The splitting dimension is chosen randomly

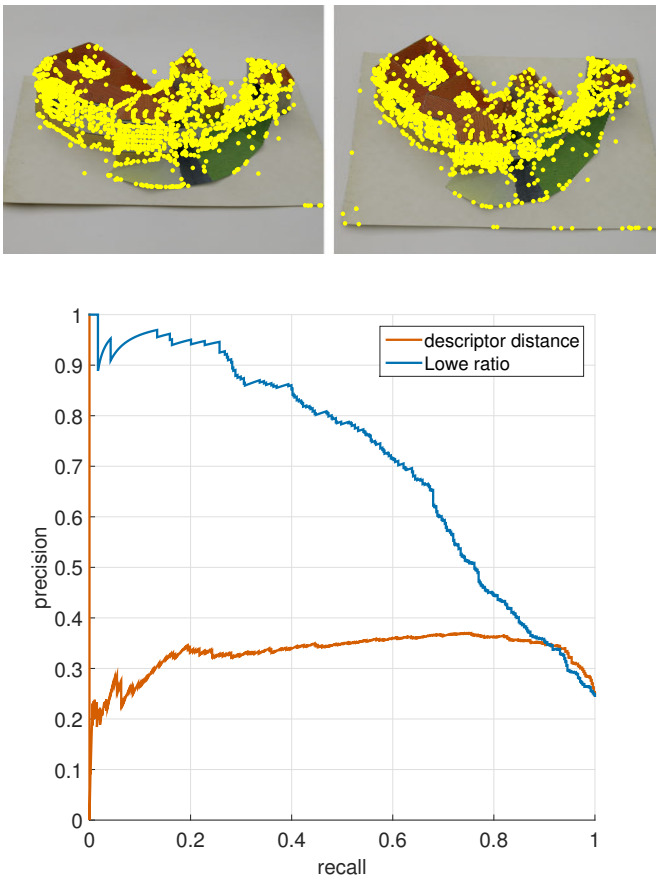


Figure 4.2 The top part of the figure shows an image pair with features detected in both images as yellow dots. 1953 features were detected in the left image and 1911 features were detected in the right one. The bottom figure gives precision-recall curves for feature matches of the image pair. We can see that Lowe ratio is a better score than simple descriptor distance. The area under curve, which can be used to summarize the quality of a score, is 0.33 for descriptor distance and 0.71 for Lowe ratio. The ground truth was obtained by manually labelling the matches.

from top five with highest variance. Thus, every tree in the forest is different. The data points, *i.e.* descriptors, are stored in leaves which means that a search procedure has to traverse a tree to the bottom. All trees are searched simultaneously by a procedure preferring better branches. Finally, the search is sped up by stopping the search after checking 32 data points. Note that this makes the algorithm approximate.

Finally, we apply one more filter. Typically, some features from the first image match the same feature from the second image. We discard all such matches as inconsistent and keep only *unique matches*. Hence, the matching becomes

$$\mathcal{M} \leftarrow \{(i,j) \mid (i,j) \in \mathcal{M} \ \wedge \ |\{(q,j) \mid (q,j) \in \mathcal{M}\}| = 1\}\,. \tag{4.6}$$

We call the filtered $\mathcal{M}$ *tentative matches*.

Similar matching schemes are also employed by [74, 1, 18, 8], for instance. However, note that tentative matches are still contaminated by mismatches and need to be further verified geometrically.

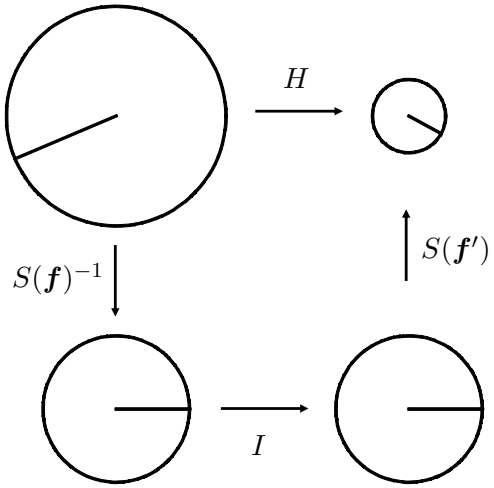


Figure 4.3 The illustration of an acquisition of a transformation from one image to the other using one match of SIFT features. The top circles represent two different features $\boldsymbol{f} \in \mathcal{F}$ and $\boldsymbol{f}' \in \mathcal{F}'$. The bottom ones are unit-scale circles with zero orientation. Note that the illustration does not take into account translation which is also part of the transformations (the centers of both unit circles are in the origin).

Growing a homography from one match

We adopt the strategy of Vedaldi and Zisserman [81] for growing a homography from one match. Given a pair of matching features $\boldsymbol{f} \in \mathcal{F}$ and $\boldsymbol{f}' \in \mathcal{F}'$, we want to find a homography $\mathtt{H} \in \mathbb{R}^{3\times3}$ with as many inliers as possible. Briefly, the strategy to reach this goal is to initialize the homography from the feature match alone and then iterate between finding inliers and using them to refine the homography.

Consider that a SIFT feature $\boldsymbol{f} = \begin{bmatrix} u & v & s & o \end{bmatrix}^\top$ can be used to compute a similarity transforming a unit circle to the feature image patch. Such a transformation is defined as

$$S(\boldsymbol{f}) = \begin{bmatrix} s\cos o & -s\sin o & u \\ s\sin o & s\cos o & v \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.7}$$

Therefore, from a pair of matching features, we can initialize the wanted homography with

$$\mathtt{H} = S(\boldsymbol{f}')S(\boldsymbol{f})^{-1}. \tag{4.8}$$

See Fig. 4.3 for a visualization of this composition.

Having $\mathtt{H}$ initialized as a similarity transformation, we are able to perform iterative refinement by alternating between inlier identification and hypothesis update using inliers. Following [81], we do four iterations of refining $\mathtt{H}$ into an affine transformation and then three iterations of refining $\mathtt{H}$ into a full homography. The hypothesis updates are detailed in Sec. 2.2. An affine transformation is estimated using least squares and a full homography via SVD.

See Alg. 1 for a pseudocode of the algorithm for growing one homography from one match. Note that different thresholds are used for different types of transformations. For similarity, affine transformation and full homography, it is 20, 10 and 5 pixels, respectively. This is justified as the transformations get less and less restrictive.

Algorithm 1: Growing a homography from one match

Input: $\mathcal{F}, \mathcal{F}', \mathcal{M}, (i,j) \in \mathcal{M}$
Output: $\mathtt{H}, \mathcal{M}_I$

for $iter \in \{1, 2, \ldots, 8\}$ **do**
  if $iter = 1$ **then**
    $\mathtt{H} \leftarrow S(\boldsymbol{f}'_j)S(\boldsymbol{f}_i)^{-1}$
  else if $iter \leq 5$ **then**
    $\mathtt{H} \leftarrow$ fit_affinity$(\mathcal{M}_I, \mathcal{F}, \mathcal{F}')$
  else
    $\mathtt{H} \leftarrow$ fit_homography$(\mathcal{M}_I, \mathcal{F}, \mathcal{F}')$
  end
  $\mathcal{M}_I \leftarrow$ find_inliers$(\mathtt{H}, \mathcal{M}, \mathcal{F}, \mathcal{F}')$
  if $|\mathcal{M}_I| < 4$ **then** `// min matches to fit a homography`
    break
  end
end

Growing multiple homographies

Finally, given tentative matches $\mathcal{M}$ and features $\mathcal{F}, \mathcal{F}'$, we geometrically verify $\mathcal{M}$ and simultaneously detect motion groups present in $\mathcal{M}$. For this purpose, we propose to extract homography with the largest support, remove its inliers from $\mathcal{M}$ and repeat while there are some reliable homographies. The individual homographies then represent motions. Our method modifies the method of Vedaldi and Zisserman [81] and extends it to dynamic scenes. They focus on finding only one homography with the largest support.

[81] use an algorithm related to RANSAC [23] and LO-RANSAC [16]. Similarly, they use a minimal sample to generate a hypothesis, refine it using inliers and return the hypothesis with the largest support. The difference is that the size of a minimal sample is fixed to be one which allows to remove randomness by using all matches as minimal samples. The hypothesis generation and refinement process, growing a homography from one match, is described above.

Such an algorithm has, however, quadratic time complexity in the number of tentative matches $|\mathcal{M}|$. We introduce three modifications to speed up the algorithm and keep the same precision and recall. First, we argue that it is not necessary to grow from every match since matches from the same true homography group tend to grow into homographies with identical or similar inlier sets. Therefore, we propose to use a set of so called visited matches $\mathcal{M}_v$. It is initialized as an empty set. As the algorithm proceeds, *i.e.* grows homographies and finds their inliers, every match that becomes inlier is added to $\mathcal{M}_v$. The algorithm then does not use matches that are in $\mathcal{M}_v$ for growing a homography. Even though this modification alone significantly speeds up the algorithm, it delivers inferior results in terms of recall compared to [81].

Second, consider that the first modification makes the algorithm dependent on the order of matches. Hence, another modification improves recall by ordering the matches. Basically, any score could be beneficial compared to random ordering. We suggest to use Lowe ratio which is readily available from the tentative-matching phase. Importantly, the sorting does not incur any significant slowdown.

Third, we propose to further improve recall by applying robust non-linear optimiza-

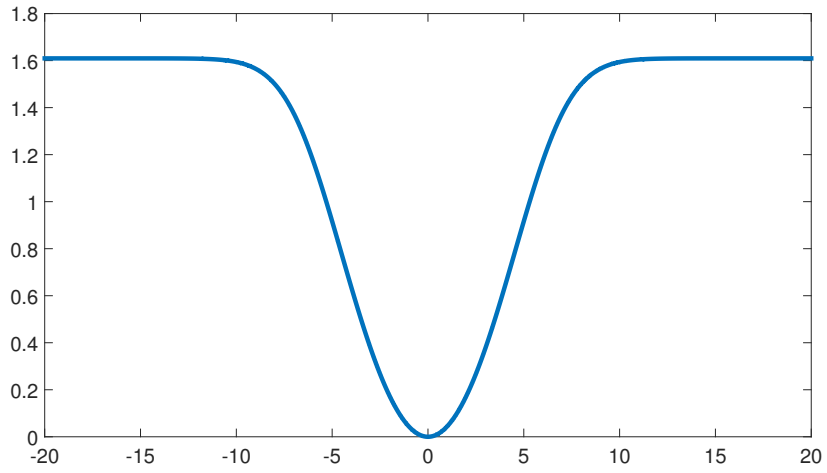


Figure 4.4 The figure displays error robustifier function, *i.e.* Eq. (4.9).

tion on all matches after the selection of the best homography by the above-described algorithm. More concretely, we initialize the optimization with the best homography, run the optimization and then identify the final inliers. The optimization is carried out by Ceres Solver [2]. It is ran with default settings except the maximum number of iterations which is set to ten in order to keep the algorithm fast. The used minimizer is trust region and linear solver sparse cholesky. The error function we use is the standard pixel distance of projected and matching point (see Sec. 2.2). We robustify the error by a function inspired by the simplified robust error function from [73], *i.e.*

$$b(err) = -\log\left(e^{-\frac{err^2}{2\sigma^2}} + q\right) + \log(q+1) \tag{4.9}$$

where q and σ are hyperparameters. We set them according to [73], *i.e.*

$$q = 0.25 \qquad \sigma = \frac{t}{\sqrt{-\log q^2}} \tag{4.10}$$

where t is inlier/outlier threshold which, as mentioned above, is five pixels. Fig. 4.4 visualizes the shape of b. Such a robustifier gives almost constant penalty to points with too large error. Hence, the optimization effectively takes into account points with error close to the threshold and does not care about high-error ones. We compute all the derivatives using inbuilt automatic differentiation tool of Ceres Solver [2] in order to obtain exact derivatives.

See a pseudocode of the full geometric verification method in Alg. 2.

4.1.2 Extension for non-planar motions

This section details how the proposed approach for detecting planar motions, described in Sec. 4.1.1, can be extended for rigid motions. Let us assume that we have detected homographies

$$\mathtt{H}_i \in \mathbb{R}^{3\times 3} \quad i \in \{1, 2, \ldots, p\}\,. \tag{4.11}$$

We propose to group the homographies based on pairwise scores, *i.e.* we take pairs of homographies and decide if they belong to one rigid motion. We first introduce a score

Algorithm 2: Growing multiple homographies

Input: $\mathcal{F}, \mathcal{F}', \mathcal{M}$
Output: $\mathcal{H}, \mathcal{G}$

$\mathcal{H}, \mathcal{G} \leftarrow \varnothing, \varnothing$
for $iH \in \{1, 2, \ldots, 7\}$ **do** `// find max 7 homographies`
  $\mathcal{M}_v \leftarrow \varnothing$
  $m \leftarrow 0$
  for $(i, j) \in \mathcal{M}$ **do** `// in the order based on Lowe ratio`
    if $(i, j) \in \mathcal{M}_v$ **then**
      continue
    end
    $\mathtt{H}_{ij}, \mathcal{M}_I \leftarrow$ grow_homography$(\mathcal{F}, \mathcal{F}', \mathcal{M}, (i, j))$ `// Alg. 1`
    $\mathcal{M}_v \leftarrow \mathcal{M}_v \cup \mathcal{M}_I$
    if $|\mathcal{M}_I| > m$ **then**
      $m \leftarrow |\mathcal{M}_I|$
      $\mathtt{H} \leftarrow \mathtt{H}_{ij}$
    end
  end
  $\mathtt{H} \leftarrow$ robust_refine$(\mathtt{H}, \mathcal{M}, \mathcal{F}, \mathcal{F}')$
  $\mathcal{M}_I \leftarrow$ find_inliers$(\mathtt{H}, \mathcal{M}, \mathcal{F}, \mathcal{F}')$
  $\mathcal{M} \leftarrow \mathcal{M} \setminus \mathcal{M}_I$
  $\mathcal{H} \leftarrow \mathcal{H} \cup \{\mathtt{H}\}$
  $\mathcal{G} \leftarrow \mathcal{G} \cup \{\mathcal{M}_I\}$
end

for determining if a pair belongs to one motion. Next, we examine degenerate situations and finally present a procedure for merging multiple homographies using the pairwise scores.

Homography-pair score

Let us assume that we have homographies $\mathtt{H}_1$ and $\mathtt{H}_2$ and we are to decide if they belong to the same motion. Consider the parameterization [60] of a general homography $\mathtt{H}$ induced by a plane in the scene

$$\lambda \mathtt{H} = \mathtt{A}' \mathtt{A}^{-1} \left(\mathtt{I} + \boldsymbol{c}' \frac{1}{z} \boldsymbol{n}^\top \right) \tag{4.12}$$

where λ is an unknown scale factor, first and second cameras are respectively

$$\mathtt{P} = \mathtt{A} \begin{bmatrix} \mathtt{I} & \begin{bmatrix} -x \\ -y \\ -z \end{bmatrix} \end{bmatrix} \quad \mathtt{P}' = \mathtt{A}' \begin{bmatrix} \mathtt{I} & \begin{bmatrix} -x' \\ -y' \\ -z' \end{bmatrix} \end{bmatrix}, \tag{4.13}$$

$\boldsymbol{c}'$ is the projection of the second camera center by the first camera and $\boldsymbol{n}$ is the normal of the plane in the first camera coordinate system. Importantly, the basis of the world coordinate system is chosen so that the first two basis vectors span the plane and third is the plane normal. Therefore, z and z' are orthogonal distances of the camera centers to the plane in the world.

We use this parameterization to prove the following theorem.

Theorem 1. *Assuming that homographies* $\mathtt{H}_1$ *and* $\mathtt{H}_2$ *are induced by two planes which move together, then* $\mathtt{H} = \mathtt{H}_2^{-1}\mathtt{H}_1$ *is a planar homology,* i.e. *has a line of fixed points and a fixed point not on the line. See Sec. 2.2.3 for planar homology.*

Proof. The homographies can be expressed in the above mentioned parameterization as

$$\lambda_1 \mathtt{H}_1 = \mathtt{A}'_1 \mathtt{A}_1^{-1} \left(\mathtt{I} + \boldsymbol{c}' \frac{1}{z_1} \boldsymbol{n}_1^\top \right) \tag{4.14}$$

$$\lambda_2 \mathtt{H}_2 = \mathtt{A}'_2 \mathtt{A}_2^{-1} \left(\mathtt{I} + \boldsymbol{c}' \frac{1}{z_2} \boldsymbol{n}_2^\top \right) \tag{4.15}$$

Note that the world coordinate system is different for both planes. Nevertheless, the calibration matrices $\mathtt{K}$ and $\mathtt{K}'$ are shared

$$\mathtt{A}_1 = \mathtt{K}\mathtt{R}_1 \quad \mathtt{A}_2 = \mathtt{K}\mathtt{R}_2 \quad \mathtt{A}'_1 = \mathtt{K}'\mathtt{R}'_1 \quad \mathtt{A}'_2 = \mathtt{K}'\mathtt{R}'_2. \tag{4.16}$$

The composition of the homographies becomes

$$\frac{\lambda_1}{\lambda_2} \mathtt{H}_2^{-1} \mathtt{H}_1 = \left(\mathtt{I} + \boldsymbol{c}' \frac{1}{z_2} \boldsymbol{n}_2^\top \right)^{-1} \mathtt{K}\mathtt{R}_2 \mathtt{R}_2'^\top \mathtt{K}'^{-1} \mathtt{K}' \mathtt{R}'_1 \mathtt{R}_1^\top \mathtt{K}^{-1} \left(\mathtt{I} + \boldsymbol{c}' \frac{1}{z_1} \boldsymbol{n}_1^\top \right). \tag{4.17}$$

Next, consider that changing the world coordinate system does not change relative rotations and thus

$$\mathtt{R}'_1 \mathtt{R}_1^\top = \mathtt{R}'_2 \mathtt{R}_2^\top \tag{4.18}$$

which simplifies the homographies composition

$$\frac{\lambda_1}{\lambda_2} \mathtt{H}_2^{-1} \mathtt{H}_1 = \left(\mathtt{I} + \boldsymbol{c}' \frac{1}{z_2} \boldsymbol{n}_2^\top \right)^{-1} \left(\mathtt{I} + \boldsymbol{c}' \frac{1}{z_1} \boldsymbol{n}_1^\top \right). \tag{4.19}$$

This can be further rewritten using

$$\boldsymbol{v}_1 = \frac{1}{z_1} \boldsymbol{n}_1 \quad \boldsymbol{v}_2 = \frac{1}{z_2} \boldsymbol{n}_2 \tag{4.20}$$

as

$$\frac{\lambda_1}{\lambda_2} \mathtt{H}_2^{-1} \mathtt{H}_1 = \left(\mathtt{I} + \boldsymbol{c}' \boldsymbol{v}_2^\top \right)^{-1} \left(\mathtt{I} + \boldsymbol{c}' \boldsymbol{v}_1^\top \right) \tag{4.21}$$

$$= \left(\mathtt{I} - \frac{\boldsymbol{c}' \boldsymbol{v}_2^\top}{1 + \boldsymbol{v}_2^\top \boldsymbol{c}'} \right) \left(\mathtt{I} + \boldsymbol{c}' \boldsymbol{v}_1^\top \right) \tag{4.22}$$

$$= \mathtt{I} + \boldsymbol{c}' \boldsymbol{v}_1^\top - \frac{\boldsymbol{c}' \boldsymbol{v}_2^\top + \boldsymbol{c}' \left(\boldsymbol{v}_2^\top \boldsymbol{c}' \right) \boldsymbol{v}_1^\top}{1 + \boldsymbol{v}_2^\top \boldsymbol{c}'} \tag{4.23}$$

$$= \mathtt{I} + \frac{\left(1 + \boldsymbol{v}_2^\top \boldsymbol{c}' \right) \boldsymbol{c}' \boldsymbol{v}_1^\top - \boldsymbol{c}' \boldsymbol{v}_2^\top - \left(\boldsymbol{v}_2^\top \boldsymbol{c}' \right) \boldsymbol{c}' \boldsymbol{v}_1^\top}{1 + \boldsymbol{v}_2^\top \boldsymbol{c}'} \tag{4.24}$$

$$= \mathtt{I} + \frac{\boldsymbol{c}' \left(\left(1 + \boldsymbol{v}_2^\top \boldsymbol{c}' \right) \boldsymbol{v}_1^\top - \boldsymbol{v}_2^\top - \left(\boldsymbol{v}_2^\top \boldsymbol{c}' \right) \boldsymbol{v}_1^\top \right)}{1 + \boldsymbol{v}_2^\top \boldsymbol{c}'} \tag{4.25}$$

$$= \mathtt{I} + \gamma \boldsymbol{c}' \left(\boldsymbol{v}_1^\top - \boldsymbol{v}_2^\top \right) \qquad / \ \gamma = \frac{1}{1 + \boldsymbol{v}_2^\top \boldsymbol{c}'}. \tag{4.26}$$

Such a matrix has a fixed point $\boldsymbol{c}'$ which is proved as

$$\mathtt{H}_2^{-1}\mathtt{H}_1\boldsymbol{c}' = \frac{\lambda_2}{\lambda_1}\left(\boldsymbol{c}' + \gamma\boldsymbol{c}'\left(\boldsymbol{v}_1^\top - \boldsymbol{v}_2^\top\right)\boldsymbol{c}'\right) \tag{4.27}$$

$$= \frac{\lambda_2}{\lambda_1}\left(1 + \gamma\left(\boldsymbol{v}_1^\top - \boldsymbol{v}_2^\top\right)\boldsymbol{c}'\right)\boldsymbol{c}' \tag{4.28}$$

$$= s\boldsymbol{c}' \tag{4.29}$$

where s is a scalar. Furthermore, the composition matrix has a line of fixed points $\left(\boldsymbol{v}_1^\top - \boldsymbol{v}_2^\top\right)$. Having a point $\boldsymbol{x}$ on the line, *i.e.* $\left(\boldsymbol{v}_1^\top - \boldsymbol{v}_2^\top\right)\boldsymbol{x} = 0$, we see that

$$\mathtt{H}_2^{-1}\mathtt{H}_1\boldsymbol{x} = \frac{\lambda_2}{\lambda_1}\left(\boldsymbol{x} + \gamma\boldsymbol{c}'\left(\boldsymbol{v}_1^\top - \boldsymbol{v}_2^\top\right)\boldsymbol{x}\right) \tag{4.30}$$

$$= \frac{\lambda_2}{\lambda_1}\boldsymbol{x}. \tag{4.31}$$

□

Corollary 1. *The matrix* $\mathtt{H} = \mathtt{H}_2^{-1}\mathtt{H}_1$ *has two equal and one distinct eigenvalues.*

Proof. Follows directly from Theorem 1 and Sec. 2.2.3. □

Corollary 2. *The eigenvector of the matrix* $\mathtt{H} = \mathtt{H}_2^{-1}\mathtt{H}_1$ *corresponding to the distinct eigenvalue is the projection of the second camera center to the first image,* i.e. *the epipole.*

Proof. Follows directly from Theorem 1, its proof and definition of eigenvectors. □

Observation 1. *The line of fixed points under* $\mathtt{H} = \mathtt{H}_2^{-1}\mathtt{H}_1$ *is the intersection of the planes.*

Proof. Intuitively follows from geometric interpretation of the problem. The points on the intersection are transformed to the same points by both homographies. □

Assuming that two eigenvalues should be equal, we propose to compute eigenvalues of $\mathtt{H} = \mathtt{H}_2^{-1}\mathtt{H}_1$ and measure how far they are from that situation. Let the eigenvalues of $\mathtt{H}$ be $\boldsymbol{\lambda} \in \mathbb{C}^3$. We distinguish two cases that can happen. Either all eigenvalues are real or there are two complex ones (a complex conjugate pair). In the first case, we compute how close are to each other all pairs to determine the pair of eigenvalues that should be equal. The closeness of the pair (λ_i, λ_j) is measured by normalizing the pair close to one and computing the difference from one, *i.e.*

$$\left|1 - \frac{\lambda_i}{\frac{\lambda_i+\lambda_j}{2}}\right| + \left|1 - \frac{\lambda_j}{\frac{\lambda_i+\lambda_j}{2}}\right| \tag{4.32}$$

which can be rewritten for all pairs into

$$e_1(\boldsymbol{\lambda}) = 2\min\left\{\left|\frac{\lambda_1-\lambda_2}{\lambda_1+\lambda_2}\right|, \left|\frac{\lambda_1-\lambda_3}{\lambda_1+\lambda_3}\right|, \left|\frac{\lambda_2-\lambda_3}{\lambda_2+\lambda_3}\right|\right\}. \tag{4.33}$$

As for the second case, assume that

$$\lambda_i = a + bi \quad \lambda_j = a - bi \quad \lambda_k = c + 0i. \tag{4.34}$$

We argue that λ_i and λ_j correspond to the same eigenvalues if the homographies originated from the same motion and they have non-zero imaginary part only because the homographies were not detected exactly due to noise. In this case, the imaginary part should be small relative to the real part. We propose to compute

$$e_2(\boldsymbol{\lambda}) = 2\left|\frac{b}{a}\right| \tag{4.35}$$

where we multiply by 2 to take into account both of the eigenvalues that should be equal. The final scoring function is then a composition of the two previously defined as

$$e(\boldsymbol{\lambda}) = \begin{cases} e_1(\boldsymbol{\lambda}) & \text{if } \boldsymbol{\lambda} \in \mathbb{R}^3, \\ e_2(\boldsymbol{\lambda}) & \text{otherwise.} \end{cases} \tag{4.36}$$

Degenerate situations for the homography-pair score

Having established Theorem 1, we need to examine what happens in images of dynamic scenes. We first state which situations are degenerate in the following theorem and then go from the general case of dynamic scenes to the degenerate situations.

Theorem 2. *Assuming that homographies* $\mathtt{H}_1$ *and* $\mathtt{H}_2$ *are induced by two parallel planes which rotate equally, then* $\mathtt{H} = \mathtt{H}_2^{-1}\mathtt{H}_1$ *is a planar homology,* i.e. *has a line of fixed points and a fixed point not on the line. See Sec. 2.2.3 for planar homology.*

Proof. We start from the general case of dynamic scenes in order to find degeneracies. The two homographies are now induced by different motions, which can be thought of as if we had two second cameras and expressed as

$$\lambda_1\mathtt{H}_1 = \mathtt{A}_1'\mathtt{A}_1^{-1}\left(\mathtt{I} + \boldsymbol{c}'\frac{1}{z_1}\boldsymbol{n}_1^\top\right) \tag{4.37}$$

$$\lambda_2\mathtt{H}_2 = \mathtt{A}_2''\mathtt{A}_2^{-1}\left(\mathtt{I} + \boldsymbol{c}''\frac{1}{z_2}\boldsymbol{n}_2^\top\right) \tag{4.38}$$

The world coordinate system is again different for both planes and the calibration matrices $\mathtt{K}$ and $\mathtt{K}'$ are shared

$$\mathtt{A}_1 = \mathtt{K}\mathtt{R}_1 \quad \mathtt{A}_2 = \mathtt{K}\mathtt{R}_2 \quad \mathtt{A}_1' = \mathtt{K}'\mathtt{R}_1' \quad \mathtt{A}_2'' = \mathtt{K}'\mathtt{R}_2''. \tag{4.39}$$

The composition of the homographies becomes

$$\frac{\lambda_1}{\lambda_2}\mathtt{H}_2^{-1}\mathtt{H}_1 = \left(\mathtt{I} + \boldsymbol{c}''\frac{1}{z_2}\boldsymbol{n}_2^\top\right)^{-1}\mathtt{K}\mathtt{R}_2\mathtt{R}_2''^\top\mathtt{K}'^{-1}\mathtt{K}'\mathtt{R}_1'\mathtt{R}_1^\top\mathtt{K}^{-1}\left(\mathtt{I} + \boldsymbol{c}'\frac{1}{z_1}\boldsymbol{n}_1^\top\right) \tag{4.40}$$

$$= \left(\mathtt{I} + \boldsymbol{c}''\frac{1}{z_2}\boldsymbol{n}_2^\top\right)^{-1}\mathtt{K}\mathtt{R}_2\mathtt{R}_2''^\top\mathtt{R}_1'\mathtt{R}_1^\top\mathtt{K}^{-1}\left(\mathtt{I} + \boldsymbol{c}'\frac{1}{z_1}\boldsymbol{n}_1^\top\right). \tag{4.41}$$

Such a matrix is generally not a planar homology.

Next, lets assume that relative rotations of the motions are the same, *i.e.*

$$\mathtt{R}_1'\mathtt{R}_1^\top = \mathtt{R}_2''\mathtt{R}_2^\top \tag{4.42}$$

which simplifies the homographies composition

$$\frac{\lambda_1}{\lambda_2}\mathtt{H}_2^{-1}\mathtt{H}_1 = \left(\mathtt{I} + \boldsymbol{c}''\frac{1}{z_2}\boldsymbol{n}_2^\top\right)^{-1}\left(\mathtt{I} + \boldsymbol{c}'\frac{1}{z_1}\boldsymbol{n}_1^\top\right). \tag{4.43}$$

This can be further rewritten using

$$\boldsymbol{v}_1 = \frac{1}{z_1}\boldsymbol{n}_1 \quad \boldsymbol{v}_2 = \frac{1}{z_2}\boldsymbol{n}_2 \tag{4.44}$$

as

$$\begin{aligned}
\frac{\lambda_1}{\lambda_2}\mathtt{H}_2^{-1}\mathtt{H}_1 &= \left(\mathtt{I} + \boldsymbol{c}''\boldsymbol{v}_2^\top\right)^{-1}\left(\mathtt{I} + \boldsymbol{c}'\boldsymbol{v}_1^\top\right) && (4.45)\\
&= \left(\mathtt{I} - \frac{\boldsymbol{c}''\boldsymbol{v}_2^\top}{1 + \boldsymbol{v}_2^\top\boldsymbol{c}''}\right)\left(\mathtt{I} + \boldsymbol{c}'\boldsymbol{v}_1^\top\right) && (4.46)\\
&= \mathtt{I} + \boldsymbol{c}'\boldsymbol{v}_1^\top - \gamma\left(\boldsymbol{c}''\boldsymbol{v}_2^\top + \boldsymbol{c}''\left(\boldsymbol{v}_2^\top\boldsymbol{c}'\right)\boldsymbol{v}_1^\top\right) \quad / \ \gamma = \frac{1}{1 + \boldsymbol{v}_2^\top\boldsymbol{c}''} && (4.47)\\
&= \mathtt{I} + \gamma\left(\frac{1}{\gamma}\boldsymbol{c}'\boldsymbol{v}_1^\top - \boldsymbol{c}''\boldsymbol{v}_2^\top - \left(\boldsymbol{v}_2^\top\boldsymbol{c}'\right)\boldsymbol{c}''\boldsymbol{v}_1^\top\right). && (4.48)
\end{aligned}$$

Such a matrix is still generally not a planar homology.

Assume further that the normals of the planes are the same, *i.e.* $\boldsymbol{n} = \boldsymbol{n}_1 = \boldsymbol{n}_2$. That simplifies the form of the composition matrix

$$\begin{aligned}
\frac{\lambda_1}{\lambda_2}\mathtt{H}_2^{-1}\mathtt{H}_1 &= \mathtt{I} + \gamma\left(\frac{1}{\gamma}\boldsymbol{c}'\frac{1}{z_1}\boldsymbol{n}^\top - \boldsymbol{c}''\frac{1}{z_2}\boldsymbol{n}^\top - \left(\boldsymbol{v}_2^\top\boldsymbol{c}'\right)\boldsymbol{c}''\frac{1}{z_1}\boldsymbol{n}^\top\right) && (4.49)\\
&= \mathtt{I} + \gamma\left(\frac{1}{\gamma z_1}\boldsymbol{c}' - \frac{1}{z_2}\boldsymbol{c}'' - \left(\boldsymbol{v}_2^\top\boldsymbol{c}'\right)\frac{1}{z_1}\boldsymbol{c}''\right)\boldsymbol{n}^\top && (4.50)\\
&= \mathtt{I} + \gamma\left(\frac{1}{\gamma z_1}\boldsymbol{c}' - \left(\frac{1}{z_2} + \left(\boldsymbol{v}_2^\top\boldsymbol{c}'\right)\frac{1}{z_1}\right)\boldsymbol{c}''\right)\boldsymbol{n}^\top && (4.51)\\
&= \mathtt{I} + (\eta'\boldsymbol{c}' - \eta''\boldsymbol{c}'')\boldsymbol{n}^\top && (4.52)
\end{aligned}$$

where $\eta' = \frac{1}{z_1}$ and $\eta'' = \gamma\left(\frac{1}{z_2} + \frac{1}{z_1 z_2}(\boldsymbol{n}^\top\boldsymbol{c}')\right)$. In this case, the composition matrix becomes a planar homology. □

Considering that our homography-pair score is vulnerable to the above described degeneracies, we propose to change the scoring function e. We argue that fitting an epipolar geometry to inliers of the homographies should result in an epipolar geometry valid for all the homography inliers and possibly some other matches. Assume that we have available tentative matches $\mathcal{M}$ and groups $\mathcal{G}_1, \mathcal{G}_2 \subseteq \mathcal{M}$ which correspond to $\mathtt{H}_1$ and $\mathtt{H}_2$ respectively. We take matches of both groups as if they were all members of one motion and use them to estimate a fundamental matrix. We use least squares to compute rank three matrix, SVD to make it rank two and non-linearly refine it. The algorithm is detailed in Sec. 2.3. The optimization is ran with 10 iterations at maximum to keep the algortihm fast. Then, we find inliers $\mathcal{M}_I$ amongst all matches, *i.e.* $\mathcal{M}_I \subseteq \mathcal{M}$. A match is considered to be an inlier if its Sampson distance is less than three pixels. The inliers are used to compute an alternative score to the first one

$$f(\mathcal{G}_1, \mathcal{G}_2, \mathcal{M}_I) = \frac{|\mathcal{M}_I|}{|\mathcal{G}_1| + |\mathcal{G}_2|}. \tag{4.53}$$

In short, function f computes relative number of inliers. In ideal case, it should be always at least one as all the matches belonging to both groups should be inliers. In addition, actual 3D objects often have matches which were not covered by the two homographies.

Finally, we present a score combining the eigenvalue function e and epipolar geometry function f

$$s(\boldsymbol{\lambda}, \mathcal{G}_1, \mathcal{G}_2, \mathcal{M}_I) = \frac{f(\mathcal{G}_1, \mathcal{G}_2, \mathcal{M}_I)}{e(\boldsymbol{\lambda})}. \tag{4.54}$$

This final score outperforms the eigenvalue score e not only in the degenerate situations. We have empirically found that noise and inaccurate homographies can corrupt the eigenvalue score. Such corruptions are effectively compensated by epipolar geometry score f. See experiments for the evaluation.

Merging multiple homographies

Let us return to the situation, where we have p homography groups and want to merge them to $\bar{p}$ true rigid motion groups. We propose to build a graph where nodes are the homography groups. Having the score s readily available for thresholding, we utilize it to create edges between nodes. A pair of homography groups has an edge between them if their score s falls below a certain threshold. In our experiments, we have empirically found a threshold of 5.5 to work well.

We define the new rigid motion groups as the connected components of the graph described above. We further use the rigid motion groups that arose from more than one homography to estimate epipolar geometry (see Sec. 2.3) and enrich the group with new inliers that were not used before in any of the existing groups. The new inliers have to have the Sampson distance smaller than three pixels. The optimization refining the epipolar geometry is ran with 30 iterations at maximum.

4.2 Structure from motion

This section presents a classical incremental structure-from-motion pipeline which takes advantage of our feature matching approach in order to handle dynamic scenes. We are inspired by popular incremental structure-from-motion systems Bundler [74] and VisualSFM [91, 92]. The input to the pipeline are only images and the output are camera parameters and sparse 3D reconstruction, *i.e.* a point cloud.

We proceed in several steps. First, we run feature matching for all pairs of images, construct n-view matches, and group the n-view matches into motion groups. Second, we select one image pair to initialize the reconstruction for every motion. Finally, we run an iterative procedure adding cameras and points to the reconstruction and refining all the parameters for every motion independently.

4.2.1 Matches

In order to be able to reconstruct 3D points, we need to know their projections in images. A set of projections of a single point in multiple images is called a n-view match (or a track in some works). We first detect two-view matches between all pairs of images in order to be able to connect them to n-view matches. We propose to apply the method detailed in Sec. 4.1 for the detection of two-view matches.

Next, we connect two-view matches across images to create n-view matches. We discard all n-view matches having more than one projection in a single image as they are deemed to be inconsistent.

Furthermore, we group the n-view matches into individual motions. Consider that k-th n-view match arose from multiple two-view detections. Let us denote the set of all the two-view detections as $t(k)$. Elements of the set $t(k)$ are pairs of indices of

images. Let us assume that $(i, j) \in t(k)$. Let us denote a function assigning an index of a group detected by two-view matching to a n-view match and a pair of image indices as $g((i, j), k)$.

We propose to merge n-view matches to motion groups starting by the most similar ones. We define the similarity as the number of pairs in which the matches are detected together. This problem can be formulated as agglomerative clustering. We initialize the algorithm with a cluster for every n-view match with t and g. Let us use the same notation for clusters as for n-view matches, *i.e.* k-th cluster has corresponding $t(k)$ and $g((i, j), k)$. We start by merging all clusters k, k' such that

$$t(k) = t(k') \ \wedge \ \forall (i, j) \in t(k) : g((i, j), k) = g((i, j), k') \tag{4.55}$$

which merges all n-view matches which are detected exactly in the same pairs of images and always move with the same motion. Next, we agglomeratively merge the remaining clusters. The distance of clusters k and k' is measured by

$$|t(k) \cap t(k')| \tag{4.56}$$

and the clusters can be merged only if

$$(i, j) \in t(k) \cap t(k') : g((i, j), k) = g((i, j), k'). \tag{4.57}$$

This approach merges groups of the most similar n-view matches until there are only unmergeable clusters. It is able to distinguish objects which move together in some image pairs and independently in others. As a result, the final groups will contain n-view matches which consistently move together in all observations.

Next, we apply a classical incremental reconstruction pipeline for every n-view match group independently.

4.2.2 Camera model

We use the camera model from Bundler [74]. Camera parameters include rotation in the angle-axis representation $\boldsymbol{r} \in \mathbb{R}^3$, camera center $\boldsymbol{c} \in \mathbb{R}^3$, focal lenght $f \in \mathbb{R}$, and radial distortion $\boldsymbol{\kappa} \in \mathbb{R}^2$. Let us denote all the camera parameters as $\mathcal{C}$. A 3D point $\boldsymbol{x} \in \mathbb{R}^3$ is projected as

$$\boldsymbol{y} = \boldsymbol{x} - \boldsymbol{c} \tag{4.58}$$

$$\theta = ||\boldsymbol{r}|| \tag{4.59}$$

$$\boldsymbol{a} = \frac{\boldsymbol{r}}{||\boldsymbol{r}||} \tag{4.60}$$

$$\boldsymbol{z} = \boldsymbol{y} \cos\theta + (\boldsymbol{a} \times \boldsymbol{y}) \sin\theta + \boldsymbol{a}(\boldsymbol{a}^\top \boldsymbol{y})(1 - \cos\theta) \tag{4.61}$$

$$\boldsymbol{v} = \frac{\boldsymbol{z}_{1..2}}{z_3} \tag{4.62}$$

$$\boldsymbol{u} = \boldsymbol{v}(1 + \kappa_1 ||\boldsymbol{v}||^2 + \kappa_2 ||\boldsymbol{v}||^4) \tag{4.63}$$

$$p(\mathcal{C}, \boldsymbol{x}) = \boldsymbol{u} f + \boldsymbol{x}_0 \tag{4.64}$$

where $\boldsymbol{x}_0$ is the principal point which is fixed to be in the middle of the image.

In case that EXIF tags of an image contain information about focal lenght in millimeters and CCD width in millimeters, we use them to compute focal length in pixels which is a camera parameter. We also use a database of CCD widths from Bundler [74] which is used to find CCD widths of cameras that do not have it in EXIF tags. The focal length in pixels is then computed as

$$\text{focal in px} = \max(\text{image width}, \text{image height}) \frac{\text{focal in mm}}{\text{CCD width in mm}}. \tag{4.65}$$

4.2.3 Reconstruction initialization

We initialize the reconstruction with a pair of cameras which is selected as the one with the most matches. The initial cameras are calibrated with zero radial distortion parameters. In case that the focal length could not be computed from EXIF tags, we compute it by assuming the angle of view 55 degrees.

Having both initial cameras calibrated, we estimate their relative pose using five-point algorithm [41] in RANSAC [23]. The first camera is set to zero rotation and put into the origin while the pose of the second camera is computed from the relative pose. Finally, 3D points corresponding to matches that are seen by both of the cameras are triangulated.

4.2.4 Incremental reconstruction

In the following, let us assume that the reconstruction has been initialized and thus there is a set of reconstructed cameras and a set of reconstructed 3D points. All steps in the following are repeated until there are no cameras to be reconstructed.

First, we find camera-to-scene matches, matches between features and 3D points. They are easily obtained as every reconstructed 3D point has an associated n-view match which defines in which cameras it is seen and to which features it corresponds.

Second, we select the camera with the most camera-to-scene matches. Assuming that it has n matches, we also select all cameras that have at least $0.75n$ matches. In addition, no cameras that have less than 16 matches are selected as they are deemed to be unreliable. If there is no camera satisfying the conditions, the reconstruction is terminated.

Third, having selected the most reliably connected cameras to the reconstruction, we estimate their parameters. The estimation is done for every camera independently by computing a full projection matrix using a RANSAC [23]. The RANSAC is used with a non-minimal least squares solver taking six camera-to-scene matches [31]. The best projection matrix is then decomposed using RQ decomposition [31] into a calibration matrix $\mathtt{K}$, rotation matrix, and camera center. The rotation matrix is converted to the angle-axis representation and the calibration is used to initialize the focal length as

$$\frac{1}{2}(\mathtt{K}_{11} + \mathtt{K}_{22}). \tag{4.66}$$

The parameters of the newly added cameras are then refined using non-linear optimization [2].

Furthermore, unreconstructed 3D points, *i.e.* n-view matches without an associated 3D point, which are visible by at least two reconstructed cameras are triangulated and added to the point cloud. The triangulation is done using numerically conditioned homogeneous method (DLT) [31]. The recovered points that have too high reprojection error are removed.

Next, bundle adjustment [79, 2], a simultaneous optimization of all camera parameters and 3D points, is ran and points with too high reprojection error are removed. These two steps are repeated until there are some points to be removed.

Finally, ill-conditioned points are removed. Ill-conditioned points are those for which the maximum angle between rays from different camera centers to the 3D point is too small. Even though these points have small reprojection error, they are not well reconstructed as the depth is not estimated accurately.

5 Implementation

We have developed a brand new C++ library called *yasfm (Yet Another Structure from Motion)*. yasfm is intended to be reliable, easy to extend and fast. It is meant to be further used for research.

Features of yasfm include feature detection, feature matching, geometric verification of matches, n-view matches handling, triangulation of 3D points, bundle adjustment, image similarity, current state storing and loading, unified RANSAC framework (see Sec. 5.2) designed for, but not limited to, various relative pose and absolute pose problems, and various auxiliary functions.

The goal of this chapter is not to fully document yasfm but rather to get the reader acquainted with the main principles and concepts. It is a sufficient documentation for developers who want to use yasfm and a good starting point for developers wishing to extend it.

We first describe which third-party libraries are used and for what in Sec. 5.1. Second, we describe the RANSAC framework and how to create a new estimator in Sec. 5.2. Lastly, we point out the main classes and structures in Sec. 5.3.

5.1 Third-party libraries

yasfm utilizes third-party libraries for several complex tasks. Some dependencies are less important than others and the following clarifies why exactly are individual libraries included.

5.1.1 Linear algebra

First and foremost, yasfm makes heavy use of linear algebra library Eigen [28]. It implements matrices, vectors, raw data interface, various decompositions (*e.g.* SVD), geometry specializations (*e.g.* quaternions or angle-axis rotation parameterization), to name a few of its features. Therefore, we are able to elegantly and quickly write matrix operations. In addition, Eigen is fast as it makes use of expression templates, vectorization via SIMD instructions, loop unrolling and cache friendliness principles.

5.1.2 Image loading

Image loading capabilities necessary for our needs are covered by one larger and one lightweight library. We employ DevIL [89], a powerful image library, for two purposes. First, for reading image dimensions to set principal points of cameras to the middle of the image. Second, for reading colors of reconstructed 3D points to get nicer visualisation. Importantly, DevIL is required also indirectly by SiftGPU (see Sec. 5.1.3).

Furthermore, the second direct image-handling dependency of yasfm is jhead [85], a lightweight library for reading EXIF tags of JPEG images. We read the following entries of EXIF tags if they are in an image: focal length in millimeters, CCD width in millimeters, camera make, and camera model. These information are, if available, used for initialization and constraining of focal lengths of cameras (see Sec. 4.2.2).

5.1.3 Feature detection and description

Detection and description of SIFT features is handled by SiftGPU [90]. We have chosen it because it is capable of utilizing GPU and is therefore fast. Importantly, we run its GLSL version which can run on all GPUs (even laptop integrated ones) so that we would not be restricted to individual makes. Note that SiftGPU has two dependencies which in turn have to be included in yasfm. One is DevIL, which yasfm needs anyway and the other is The OpenGL Extension Wrangler Library (glew).

5.1.4 Feature matching

Feature matching requires an algorithm to search for nearest neighbors in 128 dimensional space. See Sec. 4.1.1 for explanation why. For this purpose, we employ Fast Library for Approximate Nearest Neighbor (FLANN) [55]. It is a library containing several structures for performing fast approximate nearest neighbor search in high dimensional spaces. Note that we make use of randomized kd-tree forest structure only.

5.1.5 Optimization

It is critical for yasfm to have support for non-linear optimization. It is needed to refine parameters in relative and absolute pose estimators. Such optimization problems typically consist of small constant number of parameters and many residuals (one or two for every point correspondence). Non-linear optimization is also necessary for bundle adjustment, which is simultaneous optimization of all cameras and 3D points.

We choose to use popular Ceres Solver [2], a library designed for solving bundle adjustment and other computer vision optimization problems. It even offers an inbuilt automatic differentiation tool, which computes exact derivatives automatically given only a templated objective function. Note that other alternatives to automatic differentiation would be finite differences which give an approximation, symbolic differentiation which might not handle large problems, and differentiation by hand which is tedious and error prone. Ceres Solver depends on Eigen which is already included in yasfm and SuiteSparse for solving large sparse problems such as bundle adjustment. SuiteSparse in turn depends on Basic Linear Algebra Subprograms (BLAS) and Linear Algebra Package (LAPACK).

5.1.6 Other

Only two other third-party libraries are required. First, the implementation of Li and Hartley of the 5-point algorithm [41] is used for solving relative pose problem from five feature matches. Second, compression library zlib [40] enables storing features and descriptors in compressed files.

5.2 RANSAC framework

Having in mind ease of extension, we notice reoccurring pattern for various camera pose estimators. Both relative and absolute pose estimators are often put in an algorithm from the RANSAC family (see Sec. 3.1.2). The estimators differ in parameters to be estimated, minimal-sample size needed to compute a hypothesis, a way how an error is computed and what kinds of points are in correspondence.

5.2.1 RANSAC mediators

We propose to unify the estimator differences in classes we call RANSAC mediators. All the mediators are derived from a base abstract class MediatorRANSAC shown below.

Listing 5.1 Abstract base class for problem-specific features of estimators. Classes deriving from MediatorRANSAC can be plugged in RANSAC-family algorithms implemented in yasfm. Note our convention of passing a pointer of a variable to a function to emphasize that the function modifies the variable.

```
template<typename T>
class MediatorRANSAC
{
public:
  /// Total number of matches.
  virtual int numMatches() const = 0;

  /// Minimal-sample size.
  virtual int minMatches() const = 0;

  /// Is a sample valid valid for hypothesis computation?
  virtual bool isPermittedSelection(
    const std::vector<int>& indices) const
  { return true; }

  /// Compute hypotheses from a minimal sample.
  virtual void computeTransformation(
    const std::vector<int>& indices,
    std::vector<T> *hypotheses) const = 0;

  /// Compute squared error for a hypothesis and a match.
  virtual double computeSquaredError(
    const T& hypothesis,
    int matchIdx) const = 0;

  /// Optimize an initialized hypothesis on its inlier set.
  virtual void refine(
    double tolerance,
    const std::vector<int>& inliers,
    T *hypothesis) const = 0;
};
```

This design effectively makes adding new estimators only a problem of implementing a minimal solver, an error function, and optionally a refinement procedure for which one can use Ceres Solver [2].

5.2.2 Implemented solutions

At the moment, yasfm offers frameworks for "vanilla" RANSAC [23], PROSAC [15], and their variants with local optimization [16]. As for the individual camera pose problems, there are already implemented the following mediators:

- Fundamental matrix mediator for uncalibrated camera pair (using seven-point minimal solver [73]).
- Essential matrix mediator for calibrated camera pair (using five-point minimal solver [57, 41]).

- Homography mediator for uncalibrated camera pair (using four-point minimal solver [31]).

- Projection matrix mediator for camera resection (using six-point least squares solver [31]).

5.3 Main classes

This section presents main classes and structures of yasfm, *i.e.* Camera and NViewMatch classes, CameraPair and Point structures, and container class Dataset. Note that all these classes are capable of serialization. It is therefore possible to save and load the current state.

5.3.1 Camera

Camera is the most complex class. It has two main purposes: image related and camera related. First, Camera stores image filename, image dimensions, features, and descriptors. Noteworthy trait is an optimization for memory. Consider that one SIFT descriptor is stored as an array of 128 single precision floats and there are typically around 10k features in average per camera. That makes up 5.12 MB. Hence, for bigger problem instances with more than a thousand cameras, there could be memory issues (depending on memory size). We propose to use a static variable counting the number of descriptors in memory and another one keeping track of cameras that have their descriptors in memory.

```
class Camera
{
  ...
private:
  /// The number of descriptors in memory.
  static size_t nDescrInMemoryTotal_;
  /// Cameras which have their descriptors in memory.
  static std::list<Camera *> camsWithLoadedDescr_;
};
```

Further assume that the descriptors of individual cameras are stored on the disc. Hence, a camera is able to read its descriptor from the disc every time it receives a request for its descriptor and it is not currently in the memory. Finally, every time a camera reads its descriptors from the disc, the longest living set of descriptors gets released in case that the maximum number of descriptors in memory was exceeded.

Secondly, Camera serves as the base class for different camera types. Camera class itself does not have any specific parameters such as focal length, rotation or camera center. As far as the camera parameters are concerned, it has only pure virtual functions which are meant to be overloaded by derived classes. This architecture enables us to use and experiment with different camera models. It is even possible to use multiple different camera models in one scene.

5.3.2 CameraPair

CameraPair is a simple structure meant to hold pairwise information. It stores feature matches, their scores (*i.e.* Lowe ratio), and fundamental or homography matrix.

5.3.3 NViewMatch and Point

A n-view match is a set of projections of a 3D point in multiple images. N-view matches are sometimes also called tracks or point views. We represent them by NViewMatch class. This class is simply a hash map, where keys are camera indices and values are indices of features of respective cameras.

A 3D point is basically a n-view match with known 3D coordinates. We, however, define Point structure which contains coordinates and two NViewMatch objects. One with entries only with cameras which were already reconstructed as those point views are valid in the current scene and the other object for cameras that are not yet recovered.

5.3.4 Dataset

The Dataset class serves as a container for the current state. It holds a std::vector of pointers to Camera objects, CameraPair objects for matched camera pairs, and a std::vector of Point objects, for example.

Important trait of Dataset is the ability of writing the current state in a file and then reading it back. The file contains all information except features and descriptors which are stored separately in compressed binary files. In addition, the output file is a text file and is meant to be easy to read by a human. This proves useful when an immediate inspection is preferred over running additional tools. The ability to save and load the current state turns out to be very valuable for research. For instance, one can save the current state in every iteration of an incremental structure-from-motion algorithm in order to be able to start from any iteration again in case of any crashes or other problems.

6 Experiments

In this chapter, we evaluate the proposed image matching and structure-from-motion approaches detailed in Sec. 4.1 and Sec. 4.2. In the first section, we focus on image matching for scenes with planar motions only and then also on rigid-object motions. Second, we assess the performance of our structure-from-motion pipeline on a dynamic scene.

A single laptop with Intel(R) Core(TM) i7-4702MQ CPU @ 2.20GHz processor, 8GB memory, NVIDIA GeForce GT 750M GPU, and OS Windows 10 64-bit was used for all the experiments.

6.1 Image matching for dynamic scenes

Considering that the method we use for obtaining tentative matches is widely used [74, 1, 18, 8], we evaluate only our geometric verification approach which is responsible for detecting motions and discarding mismatches. Hence, assume that we have tentative initial matches $\mathcal{M}_0 \subseteq \{1, 2, \ldots, n\} \times \{1, 2, \ldots, m\}$ where n and m are numbers of features detected in images I and I' respectively.

For the purpose of measuring quality of the matches verification, we use

$$\text{precision} = \frac{\text{true positive}}{\text{true positive} + \text{false positive}} \tag{6.1}$$

$$\text{recall} = \frac{\text{true positive}}{\text{true positive} + \text{false negative}} \tag{6.2}$$

which in our case becomes

$$\text{precision} = \frac{|\{\text{correct matches}\}| \cap |\{\text{matches labelled as correct}\}|}{|\{\text{matches labelled as correct}\}|} \tag{6.3}$$

$$\text{recall} = \frac{|\{\text{correct matches}\}| \cap |\{\text{matches labelled as correct}\}|}{|\{\text{correct matches}\}|}. \tag{6.4}$$

Precision therefore measures how correct is the set of matches we have labelled as correct and recall how many correct matches is obtained from all correct ones there are.

We want to extract (as shown in Alg. 2) mutually exclusive motion groups

$$\mathcal{G} = \left\{ \mathcal{G}_k \;\middle|\; \mathcal{G}_k \subseteq \mathcal{M} \;\wedge\; k \neq k' : \; \mathcal{G}_k \cap \mathcal{G}_{k'} = \varnothing \right\}. \tag{6.5}$$

The ground truth groups are similarly denoted by

$$\overline{\mathcal{G}} = \left\{ \overline{\mathcal{G}}_k \;\middle|\; \overline{\mathcal{G}}_k \subseteq \mathcal{M} \;\wedge\; k \neq k' : \; \overline{\mathcal{G}}_k \cap \overline{\mathcal{G}}_{k'} = \varnothing \right\}. \tag{6.6}$$

Note that the number of estimated groups $|\mathcal{G}|$ can be different from the number of ground truth groups $\left|\overline{\mathcal{G}}\right|$ as the ground truth number of motions is not required on input by our approaches. Hence, we propose to find a mapping which determines the

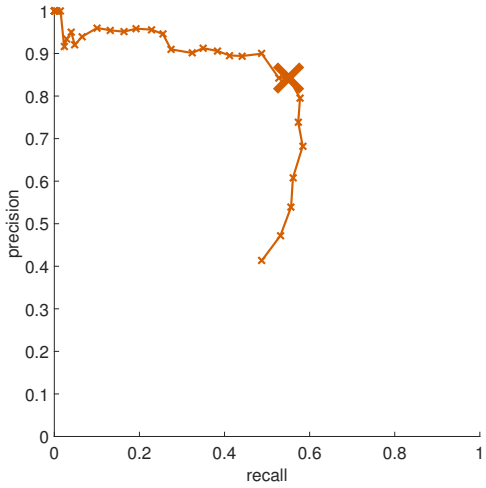


Figure 6.1 The precision-recall curve for the image pair F for varying Lowe ratio in [0,1] with step 0.025. The set of all matches are all tentative initial matches $\mathcal{M}_0$. Matches are labelled as correct if they pass Lowe-ratio and unique-matches filters described in Sec. 4.1. The large cross corresponds to Lowe ratio 0.8.

most similar estimated group for every ground truth group. We require the mapping to map one estimated group to every ground truth group at maximum. The mapping is more formally written as

$$h : \mathcal{G} \to (\overline{\mathcal{G}} \cup \{\varnothing\}) \qquad \text{s.t.} \left|\left\{\mathcal{G}_k \;\middle|\; \mathcal{G}_k \in \mathcal{G} \wedge h(\mathcal{G}_k) = \overline{\mathcal{G}}_{k'}\right\}\right| \leq 1 \;\forall \overline{\mathcal{G}}_{k'} \in \overline{\mathcal{G}} \tag{6.7}$$

such that, the following is maximized

$$|\mathcal{G}_k \cap h(\mathcal{G}_k)| \;\forall \mathcal{G}_k \in \mathcal{G}. \tag{6.8}$$

Subsequently, we use the mapping h to compute precision and recall for every group individually, *i.e.* correct matches for a group, as denoted in Eq. (6.3) and Eq. (6.4), represent inliers that belong to the group. That makes correct matches which successfully passed geometric verification, but were assigned to the wrong group, to be handled as mismatches.

To evaluate our approaches for geometric verification, we use three publicly available datasets. First, we use the dataset of Cho *et al.* [14] designed for object-level matching. It features six image pairs of multiple objects (toys, posters, books) moving independently. In addition, all images have different unrelated backgrounds. Second, the dataset of Wang *et al.* [86] contains seven image pairs of various objects moving independently. [86] detects the motions and does 3D reconstruction in order to find scene changes.

Third dataset is called Hopkins155 [80]. It is primarily meant to be used for motion segmentation from multiple views and contains 155+16 short videos. 104 videos show objects covered with checkerboard pattern and were captured under controlled conditions. We discard those videos and consider only the other ones since they model real world conditions. Most of them capture cars on the road. Since our method needs just a pair of images, we follow [69, 64] and use first and last frames only. Further, we do not use tracked points that are shipped with the dataset and instead detect SIFT features as detailed in Sec. 4.1.

Id	Name	$\lvert\mathcal{M}_0\rvert$	$\lvert\mathcal{M}\rvert$	$\bar{p}$	Group sizes	Motion types
A	Hopkins155 cars10	8166	736	3	(341,150,20)	planar
B	Hopkins155 kanatani1	989	298	2	(231,12)	planar
C	Hopkins155 books	2303	469	5	(53,49,63,120,59)	planar
D	Cho *et al.* Bulletins	4223	485	3	(147,54,51)	planar
E	Cho *et al.* Jigsaws	5673	261	4	(28,44,12,5)	planar
F	Ours daliborka	1953	425	1	(297)	rigid
G	Ours indoor	10595	948	1	(398)	rigid
H	Wang *et al.* 1	968	375	3	(288,51,1)	rigid
I	Wang *et al.* 2	838	250	3	(142,58,22)	rigid
J	Wang *et al.* 5	2162	577	2	(465,49)	rigid
K	Hopkins155 kanatani2	633	110	2	(37,33)	planar

Table 6.1 The image pairs for which we create ground truth. Id is given by us whereas name column presents dataset and image pair names as given by their authors. $\mathcal{M}_0$ are tentative initial matches and $\mathcal{M}$ those matches with Lowe ratio smaller than 0.8. We also provide the number of groups $\bar{p}$, their sizes in $\mathcal{M}$, and motion type. The pairs A-E are used to evaluate the homography growing approach and F-J the rigid motion approach.

In order to evaluate the proposed approaches using precision and recall, we need ground truth data. We therefore select several image pairs (see Tab. 6.1) and compute tentative initial matches $\mathcal{M}_0$ on them. Next, we manually label the matches. To ease up the process of labelling, we assume matches with Lowe ratio higher than 0.8 to be mismatches. Values larger than 0.8 empirically proved to be too permissive as majority of matches with larger Lowe ratio are mismatches. This is justified on an example in Fig. 6.1. The image pairs for manual labelling are selected so that there are representatives of different types of scenes (*e.g.* cars, posters) and different numbers of motions. We also take into account how easy it is for a human to determine correctness of a match (*e.g.* a scene consisting of sand and rocks would be difficult to label correctly).

6.1.1 Growing homographies

We compare our approach from Sec. 4.1.1 with two others. First one is well-known RANSAC [23]. We run it with the same inlier threshold as in our approach, *i.e.* five pixels. In addition, we limit the RANSAC to run for 4096 rounds at maximum to speed it up. Similar limitation was used in [74], for instance. After the RANSAC finishes, we refine the hypothesis by fitting a homography to inliers as in Sec. 2.2 and then find inliers to the new homography. To extract multiple motions, the RANSAC is applied sequentially. Such a scheme was applied also in [8, 76], for example. The second

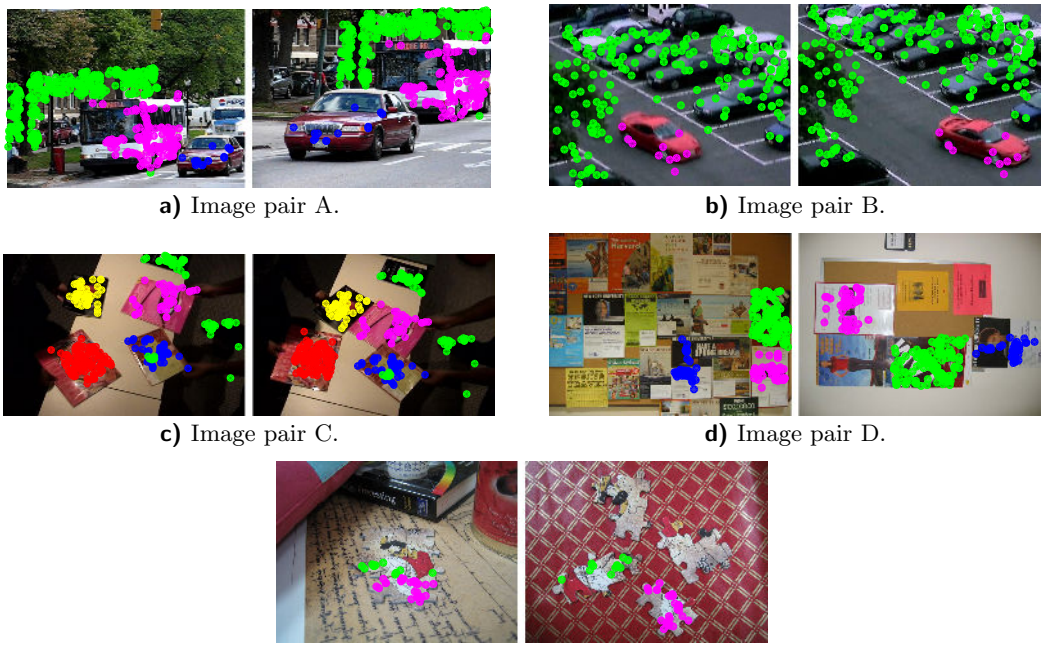
		Sequential RANSAC	Sequential VGG-P	Ours (Sec. 4.1.1)
A	p	**3**	**3**	**3**
	# inliers	(280,118,12)	(280,118,13)	(281,118,13)
	precision	(1.00,1.00,1.00)	(1.00,1.00,1.00)	(1.00,1.00,1.00)
	recall	(0.82,0.77,0.60)	(0.82,0.77,0.65)	(0.82,0.77,0.65)
	time	0.181	0.437	**0.016**
B	p	1	**2**	**2**
	# inliers	(191,0)	(191,10)	(192,10)
	precision	(0.98,1.00)	(0.98,0.90)	(0.98,0.90)
	recall	(0.81,0.00)	(0.81,0.75)	(0.81,0.75)
	time	**0.003**	0.152	0.006
C	p	**5**	**5**	**5**
	# inliers	(56,28,35,97,40)	(56,36,36,97,41)	(56,36,36,97,40)
	precision	(0.89,1.00,1.00, 1.00,0.83)	(0.89,1.00,1.00, 1.00,0.98)	(0.89,1.00,1.00, 1.00,1.00)
	recall	(0.89,0.57,0.56, 0.81,0.56)	(0.89,0.74,0.57, 0.81,0.68)	(0.89,0.74,0.57, 0.81,0.68)
	time	0.206	0.201	**0.030**
D	p	4	**3**	**3**
	# inliers	(130,41,35)	(130,41,35)	(130,41,36)
	precision	(1.00,0.98,1.00)	(1.00,0.98,1.00)	(1.00,0.98,0.98)
	recall	(0.88,0.74,0.69)	(0.88,0.74,0.69)	(0.88,0.74,0.69)
	time	0.130	0.052	**0.009**
E	p	2	2	2
	# inliers	(10,23,0,0)	(11,30,0,0)	(11,29,0,0)
	precision	(1.00,0.96,1.00,1.00)	(1.00,0.97,1.00,1.00)	(1.00,1.00,1.00,1.00)
	recall	(0.36,0.50,0.00,0.00)	(0.39,0.66,0.00,0.00)	(0.39,0.66,0.00,0.00)
	time	0.130	0.006	**0.003**

Table 6.2 The number of motions p is written in bold if it was correctly estimated. The number of inliers, precision, and recall are given for ground truth groups as mapped by Eq. (6.7) and colored red when a motion was not successfully detected. Time is measured in seconds and is averaged over 100 runs.

approach is sequential application of VGG practical [81] which we denote VGG-P. Note that VGG-P's homography-growing core is used in our approach too and is described in Alg. 1. In contrast to our method, VGG-P grows a homography from every match and is thus expected to be slower. A hypothesis is accepted by all methods if it has at least ten inliers.

Tab. 6.2 gives the results for image pairs A-E. The methods were supplied with tentative matches filtered with Lowe ratio set to 0.7. We observe that our approach performs just as well or better than sequential RANSAC and just as well as sequential VGG. Additionally, our approach is significantly more efficient as it is in most cases the fastest one. In some cases, it performs faster by an order of magnitude. The matches verified by our approach are visualized in Fig. 6.2.

Moreover, we have empirically observed that Lowe ratio threshold greatly influences the subsequent geometric verification. Lowe-ratio filtering predominantly changes the proportion of mismatches in the set of matches fed to the geometric verification algorithms. Nevertheless, in case the threshold is set too low, it could easily become



a) Image pair A. **b)** Image pair B. **c)** Image pair C. **d)** Image pair D. **e)** Image pair E.

Figure 6.2 Matches verified by our homography growing method corresponding to the right column of Tab. 6.2. The colors green, magenta, blue, red, and yellow represent matches of different groups. This order of colors is consistent with the order of groups across all image pairs and corresponds to the groups ordering in tables.

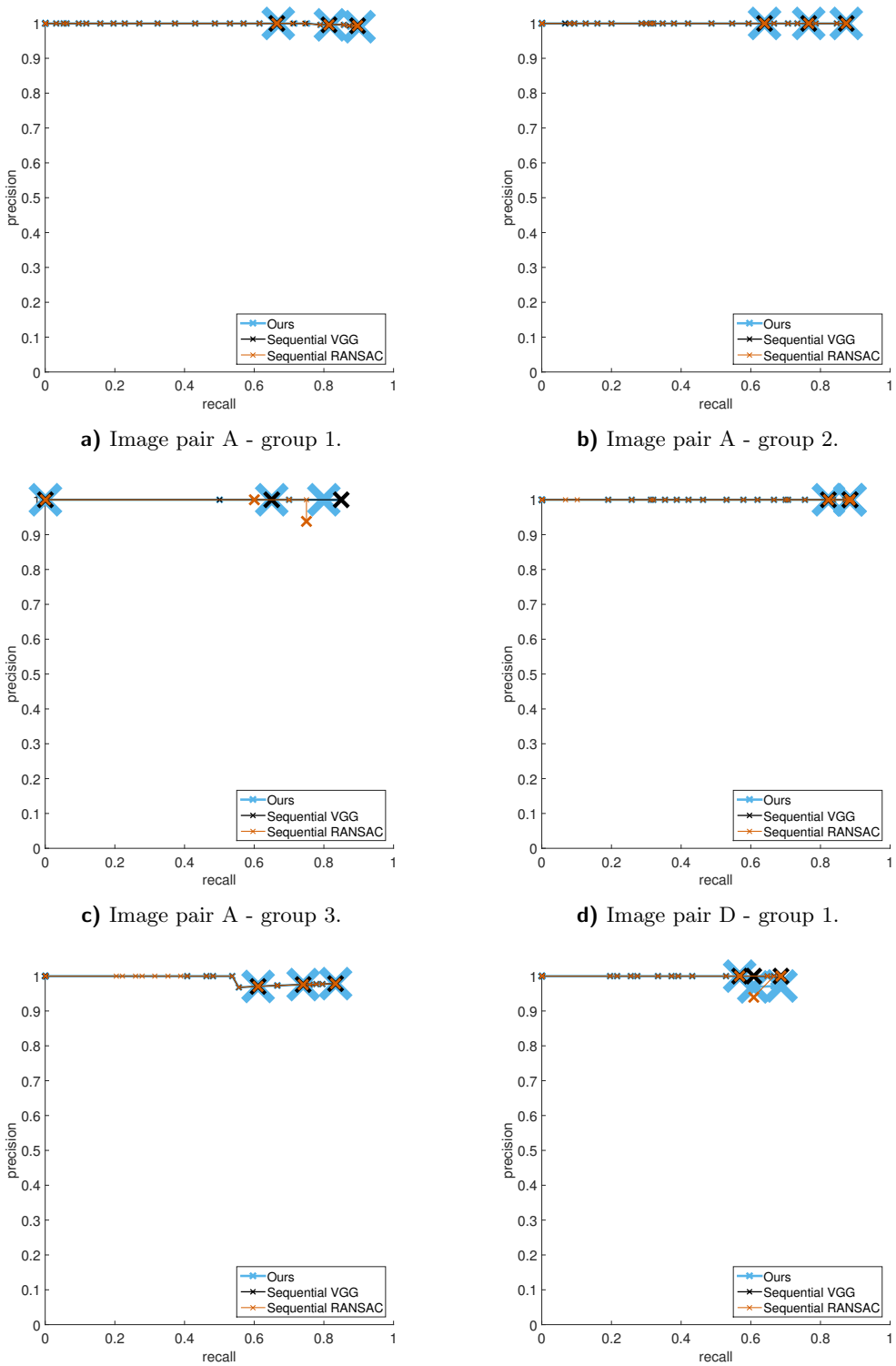
impossible for geometric verification to find motions which are supported by only a few matches scoring too high. Therefore, we deem important to show precision-recall curves for varying Lowe ratio threshold. We visualize them for threshold varying in interval [0,0.8] for image pairs A and D in Fig. 6.3.

Fig. 6.3 confirms that setting Lowe ratio threshold too low can result in discarding too many correct matches for a group to be detected. For example, setting the threshold as high as 0.6 would be still too strict for the third group in the image pair A. Nevertheless, all the approaches perform similarly on all the groups with two minor exceptions. One exception is Fig. 6.3c where our approach has slightly worse recall with the most permissive threshold and sequential RANSAC has slightly worse recall and in one case also slightly worse precision. The other exception is Fig. 6.3f where our approach has slightly worse precision for threshold values larger than 0.7 and sequential RANSAC which has slightly worse precision for threshold 0.8. Note that we call the differences slight because they are induced by accepting one wrong or discarding one correct match only.

We conclude that our homography growing approach is successful in dynamic-scene matching task for planar motions. It is capable of finding multiple motion groups among a set of matches corrupted by outliers. Our approach is just as good or better than other compared approaches. It is also generally faster.

6.1.2 Rigid motions

Let us now consider general rigid motions. Such motions cannot be modelled by a single homography anymore. Fig. 6.4 shows what happens if we run our approach for growing multiple homographies on a scene featuring non-planar motions. Notice that in scenes



a) Image pair A - group 1.

b) Image pair A - group 2.

c) Image pair A - group 3.

d) Image pair D - group 1.

e) Image pair D - group 2.

f) Image pair D - group 3.

Figure 6.3 Precision-recall curves for varying Lowe ratio in interval [0,0.8] with step 0.025. Individual curves have different line width and marker size to enable a nice visualization. Additionally, every curve has three larger markers. The larger crosses represent from left to right values of the ratio 0.6, 0.7, and 0.8.

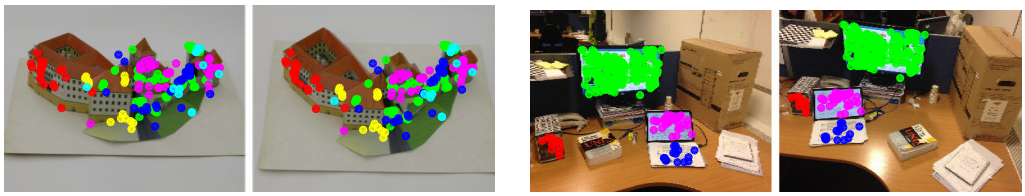


Figure 6.4 The approach for growing multiple homographies applied to rigid motions. Different colors represent different homography groups. The left image pair is F and the right one is H. F has six detected groups and H four. Note that F has only one motion and is supposed to illustrate what homographies can get detected. However, also note that all the matches on F are correct. The only problem is that the detected homographies correspond to some "virtual" planes.

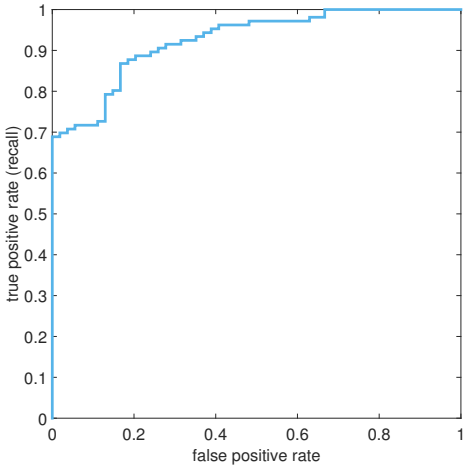


Figure 6.5 ROC curve for homography merging score s. Array under curve, which represents the quality of the score, is 0.93. Positive hypothesis is that a homography pair belongs to one motion.

with no dominant plane, our greedy approach can draw a homography which does not respect any real world plane. That is not a problem for our extended approach (see Sec. 4.1.2).

To evaluate the homography merging score presented in Sec. 4.1.2, we run our homography growing method on 38 image pairs. There are 12 pairs from Hopkins155 [80], six from Cho *et al.* [14], seven from Wang *et al.* [86] and we have added 13 of our own single motion image pairs. The multiple motion image pairs show scenes with moving cars on the road, books, posters, and toys. The single motion image pairs capture a paper castle model, Notre Dame, a statue, indoor scenes, and Mars landscape. After running the homography growing, there are 160 homography pairs that could be possibly merged. We manually create ground truth to enable quantitative evaluation of the merge score.

To this end, we have been using precision-recall curves but we find ROC curves more suitable in this case. Note that precision-recall curves do not visualize how good the set of negative labels is which is in the homography merging just as important as how the set of positive labels is. ROC on the other hand shows recall, also called true positive

		Sequential RANSAC	Ours (Sec. 4.1.2)
F	p	**1**	**1**
	# inliers	(200)	(203)
	precision	(0.98)	(0.97)
	recall	(0.66)	(0.66)
	time	**0.042**	0.055
G	p	**1**	**1**
	# inliers	(273)	(276)
	precision	(1.00)	(1.00)
	recall	(0.69)	(0.69)
	time	0.175	**0.045**
H	p	3	2
	# inliers	(220,43,28)	(233,45,0)
	precision	(0.98,0.93,0.00)	(1.00,1.00,1.00)
	recall	(0.75,0.78,0.00)	(0.81,0.88,0.00)
	time	**0.015**	0.036
I	p	2	2
	# inliers	(120,30,0)	(117,30,0)
	precision	(0.98,0.87,1.00)	(1.00,1.00,1.00)
	recall	(0.83,0.45,0.00)	(0.82,0.52,0.00)
	time	0.034	**0.014**
J	p	**2**	**2**
	# inliers	(421,42)	(418,27)
	precision	(1.00,0.98)	(1.00,1.00)
	recall	(0.90,0.84)	(0.90,0.55)
	time	**0.042**	0.051
K	p	1	**2**
	# inliers	(63,0)	(36,26)
	precision	(0.53,1.00)	(1.00,1.00)
	recall	(0.87,0.00)	(0.87,0.76)
	time	**0.001**	0.003

Table 6.3 The number of motions p is written in bold if it was correctly estimated. The number of inliers, precision, and recall are given for ground truth groups as mapped by Eq. (6.7) and colored red when a motion was not successfully detected. Time is measured in seconds and is averaged over 100 runs.

rate, on one axis and false negative rate on the other. The rates are in our case

$$\text{true positive rate} = \frac{|\{\text{labelled as one motion}\}| \cap |\{\text{is one motion}\}|}{|\{\text{is one motion}\}|} \tag{6.9}$$

$$\text{false positive rate} = \frac{|\{\text{labelled as one motion}\}| \cap |\{\text{is different motion}\}|}{|\{\text{is different motion}\}|}. \tag{6.10}$$

The resulting curve for our score s is shown in Fig. 6.5.

In order to evaluate the final matching method for rigid motions, as detailed in Sec. 4.1.2, we propose to compare it to a sequential RANSAC. The RANSAC estimates fundamental matrices using seven-point algortihm (see Sec. 2.3) unlike the previous section, where it estimated homographies. A hypothesis is accepted if it is supported

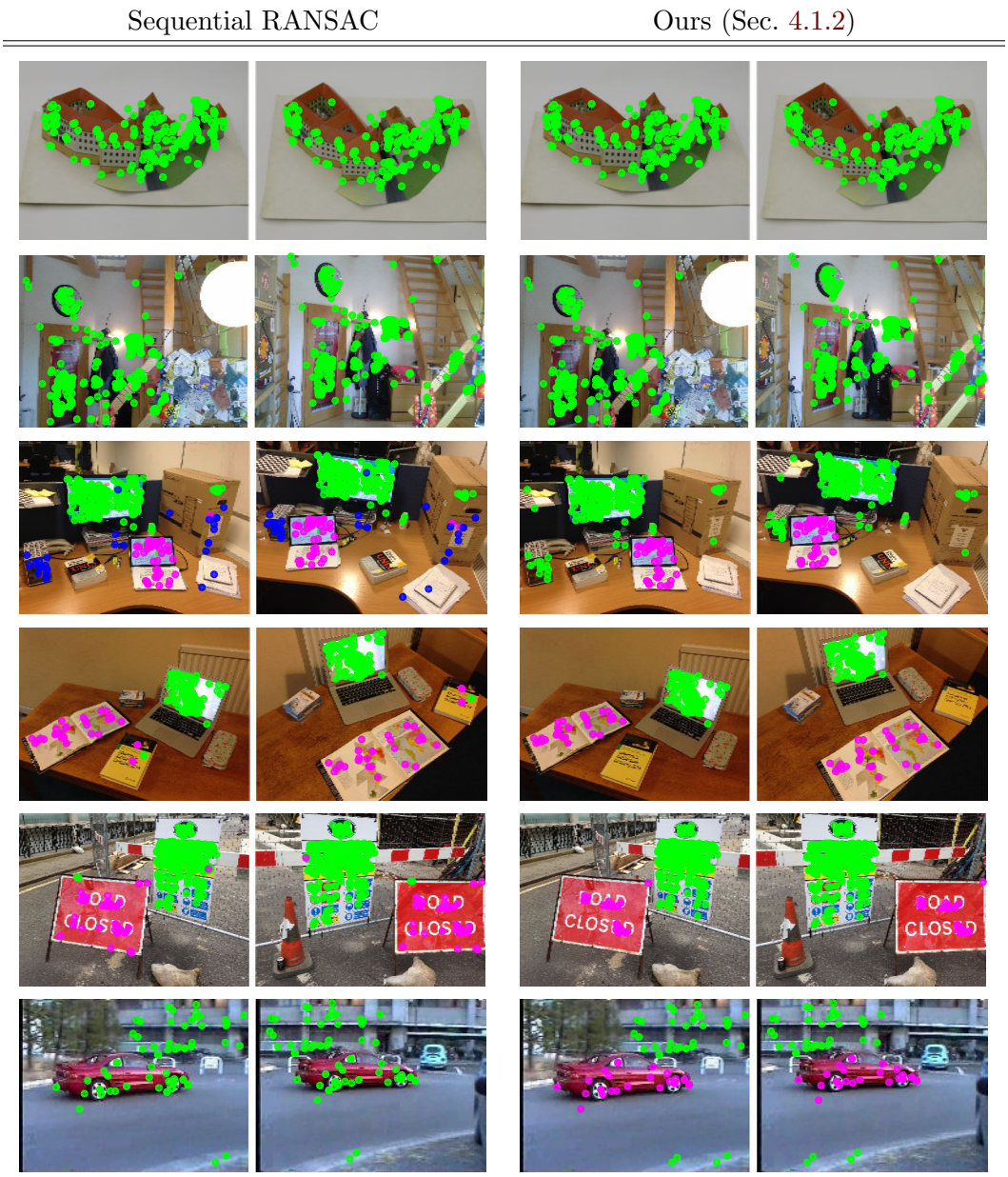


Figure 6.6 The figure shows image pairs F-K from top to bottom and visualizes matches corresponding to Tab. 6.3. The colors green, magenta, and blue represent matches of different groups. This order of colors is consistent with the order of groups across all image pairs and corresponds to the groups ordering in tables.

by at least 16 inliers and a match is considered to be an inlier if its Sampson distance is below three pixels. The results are given in Tab. 6.3 and visualized in Fig. 6.6. The tentative matches fed to the verification are again generated with Lowe ratio 0.7.

To summarize, our approach is in comparison to the RANSAC better for H, I, and K, just as good for F and G and slightly worse for J. Importantly, our approach avoids generating too many groups for H and too few groups for K. The results for J are worse only in terms of the number of found matches and not in the number of detected groups.

Figure 6.7 Several image pairs with geometrically verified matches visualized as green, magenta, and blue dots. The bottom right image pair shows a failure example where a repetitive pattern on the ground was incorrectly matched.

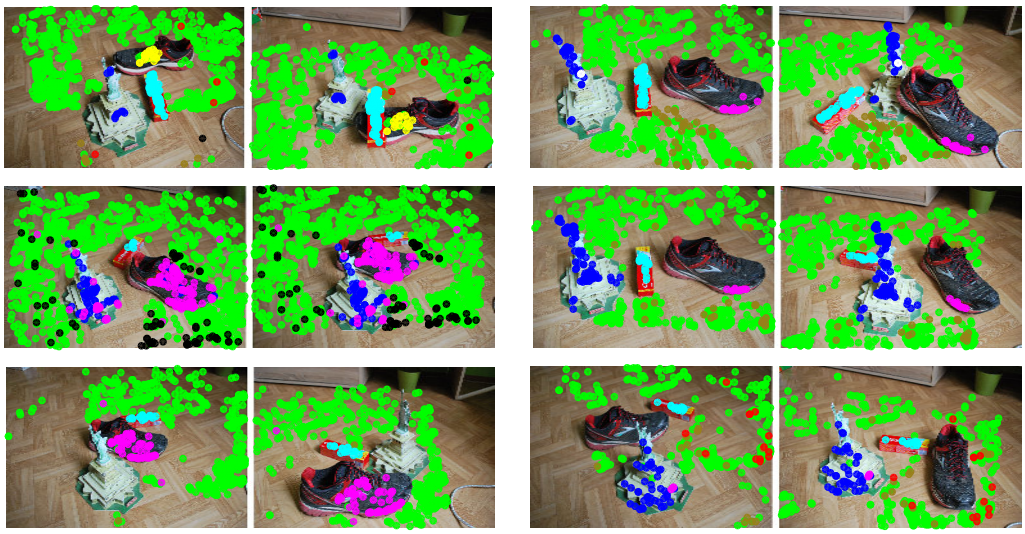
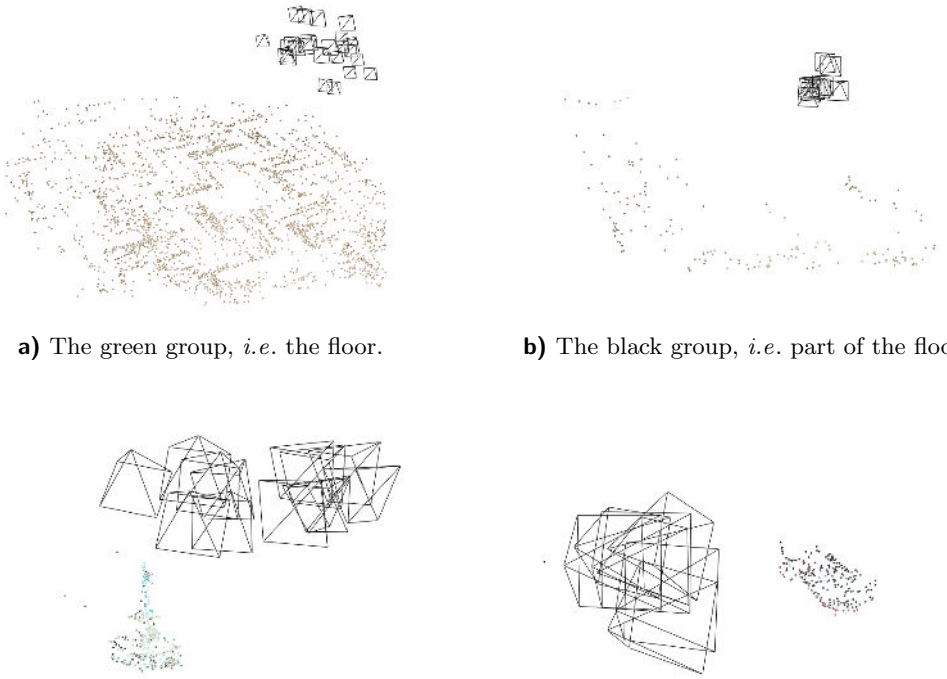


Figure 6.8 Several image pairs with visualized n-view matches which are seen by both images in a pair. Note that the groups are consistent on different image pairs as opposed to Fig. 6.7. The color green corresponds to the ground, blue to the paper statue, magenta and yellow to the two sides of the shoe, and cyan to the toothpaste box. In addition, the colors red, black, brown, and white symbolize matches of the ground which were not merged to the green group.

6.2 Structure from motion

This section evaluates the proposed structure-from-motion approach for dynamic scenes. We have captured a dataset where three objects move independently, *i.e.* there are four motions including the background. The dataset consists of 24 images with known focal length. The objects are a shoe, a paper Statue of Liberty, and a toothpaste box.

We run the approach as described in Sec. 4.2 and hence start by running feature matching for all image pairs. Fig. 6.7 visualizes several results of two-view matching,



a) The green group, *i.e.* the floor.

b) The black group, *i.e.* part of the floor.

c) The blue group, *i.e.* the paper statue.

d) The magenta group, *i.e.* half of the shoe.

Figure 6.9 A visualization of sparse reconstructions computed by our structure-from-motion system. The pyramids symbolize reconstructed cameras and their viewing directions.

i.e. geometrically verified feature matches. These examples cover various situations that can happen when matching dynamic scenes with our approach for rigid motions. In case that all objects move sufficiently differently and have enough tentative matches, we are able to group matches correctly. On the other hand, matches of two objects can be grouped together if they move together (with respect to one image pair). Nevertheless, note that our approach for connecting two-view matches into n-view matches fuses the grouping information available from all image pairs and aims at distinguishing different motions even if some are not distinguishable on the image-pair level. Furthermore, Fig. 6.7 illustrates a failure where a repetitive pattern on the floor was incorrectly matched. Note that similar failures can possibly confuse the fusing step.

Fig. 6.8 takes the same image pairs as in Fig. 6.7 and visualizes n-view matches that are seen by both images in a pair. The figure gives an idea of how the final grouping ended up. It is clear that the largest group is the ground one. Then there is a group for the paper statue and a group for the toothpaste box. The figure also reveals that the shoe was split in two groups which correspond to the two sides of it. Furthermore, there are five other groups which are mostly on the ground. We reason that they were induced by the incorrect matches of repetitive patterns and then could not be merged to the main ground group as it would be contradicting based on Sec. 4.2.1.

Finally, the reconstructed sparse point clouds with recovered cameras are displayed in Fig. 6.9. Only the groups which resulted in point clouds which allow for a nice visualization are shown. In order to further verify the results of our algorithm, we supply the camera parameters to CMPMVS [36] to obtain dense reconstructions (see Fig. 6.10). Note that correctly estimated camera parameters are needed to obtain good

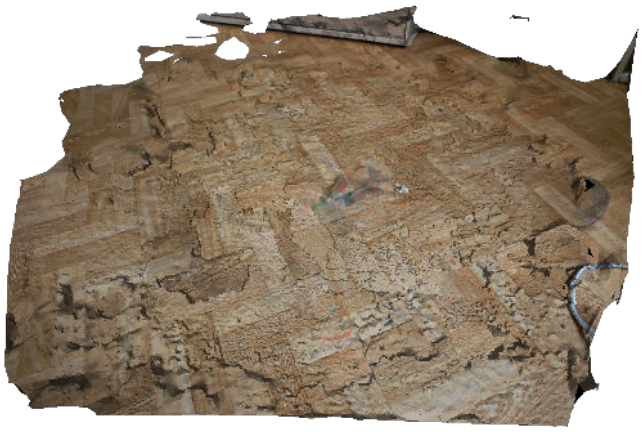
Group	# matches	Avg. match size	# points	# cams	Avg. error
Green	8261	6.65	3018	24	0.623
Blue	812	3.62	466	14	0.888
Magenta	752	3.18	253	7	0.746
Yellow	176	2.03	101	2	0.432
Cyan	745	3.61	60	7	1.746
Red	752	2.31	135	2	0.444
Black	416	3.24	194	11	0.600
Brown	1345	3.54	260	10	1.146
White	31	2.19	18	2	0.287

Table 6.4 The table gives the number of n-view matches and their average size. A size of a n-view match is the number of images in which it is seen. Then, there is the number of reconstructed 3D points, cameras and average reprojection error. The colors correspond to Fig. 6.8 and Fig. 6.9.

dense reconstructions.

Quantitative results are given in Tab. 6.4. The table gives the number of n-view matches and their size. The higher these quantities are, the better as we want many possible 3D points which are seen in many images. Next, there are the number of recovered 3D points, cameras and average reprojection error. The groups that correspond to the objects and ground were reconstructed successfully with one exception. Despite the number of matches the toothpaste box has it did not reconstruct well. We reason that the structure-from-motion system did not manage to initialize the reconstruction from matches of a single small plane as it is using five-point algorithm (see Sec. 4.2.3).

To summarize, our feature matching approach for rigid motions coarsely grouped two-view matches which were then fused into n-view matches with better grouping. Even though a few redundant groups appeared, there were major groups representing different objects and the background. Our structure-from-motion pipeline was then able to reconstruct the groups individually.



a) The green group, *i.e.* the floor.

b) The blue group, *i.e.* the paper statue.

c) The magenta group, *i.e.* the shoe.

Figure 6.10 Dense reconstructions obtained by CMPMVS [36] when supplied with the camera parameters found by our approach. Note that this proves the correctness of the found camera parameters. The artifacts around the objects of interest in (b) and (c) are caused by the fact that the objects are not observed from the other side.

7 Open Problems and Future Work

This chapter covers limitations of the proposed approaches and future work. We identify two problematic situations related to the homography-growing approach. First, note that there have to be a sufficient number of tentative matches for every motion in order to detect all motions. Consider Fig. 6.2e, for example, where our approach detected two motions out of four. A possible solution could utilize generation of synthetic views to increase the number of tentative matches (see Sec. 3.1.1). Second, the greedy nature of our approach could make one motion take over some matches of another motion. For instance, see the green and blue groups in Fig. 6.2c. There are matches which should be part of the blue group but are not. The green group was detected before the blue one and it greedily extracted all its inliers including a few of the blue-group object matches. Note, however, that both of the problems also arise for the other compared methods.

Furthermore, we have noticed a limitation of the proposed approach for rigid motions. Even though we aim at identification of degeneracies, there are still some that elude us. One such degeneracy is the image pair A. We detect three homographies as shown in Fig. 6.2a corresponding to the background, the bus and the car. Then, we compute pairwise scores for all three pairs of homographies to determine which ones belong to the same motion. Next, two pairs of the homographies are incorrectly labelled, *i.e.* all three homographies are a connected component and they all get merged (see Fig. 7.1). This should not happen since it is impossible to fit an epipolar geometry to all three homography groups in this case even though it is possible to fit it to two pairs of homographies. A question arises how to determine which homography groups should be merged in similar situations, if any. We plan to address this issue in the future.

Finally, the approach for grouping n-view matches can have problems when some of the two-view groups are incorrect. In this case, two groups of matches can remain separate even though they should be merged.

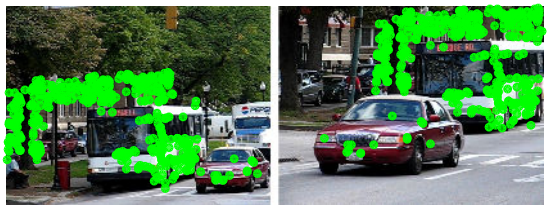


Figure 7.1 A degenerate situation not handled by the approach for rigid motions. All the matches have been grouped into one group.

8 Conclusion

This work first established the notation, basic concepts and reviewed the related state of the art. Then, the proposed approaches were introduced and subsequently, the implementation described. Furthermore, the proposed approaches were evaluated and their limitations identified.

We propose a two-view feature matching approach suitable for dynamic scenes. It is important to consider dynamic scenes with differently moving objects if one wants to handle uncontrolled conditions since the real world is dynamic. We take a classical tentative feature matching scheme which is widely used and design a method for geometric verification of the matches. We propose to grow homographies from feature correspondences which readily provide similarity transformations. Importantly, our contribution is the application of the homography growing scheme to extract multiple homographies and speed improvement of the scheme.

Furthermore, we propose to apply the scheme for growing multiple homographies to scenes with rigid motions which returns multiple homographies where some may belong to the same motion. We compute a score to measure how far a pair of homographies is from the situation of being from one motion. The score is then used to merge homographies into motion groups. The score utilizes a property that a composition of two homographies should be a planar homology if they belong to the same motion. Also, the score takes advantage of an observation that a fundamental matrix fitted to inliers of two homographies should model the inliers and possibly some other matches.

Moreover, we present a C++ library intended for building and experimenting with structure-from-motion systems. It is designed to be reliable, easy to extend and fast as it is meant to be further used for research.

Finally, we propose how our feature matching approach can be used to match pairs of images and then fuse the information about pairwise groups into global groups. Every global group then consists of a set of n-view matches. Hence, we are able to run a classical incremental structure-from-motion system, implemented using our library, for every group independently.

Bibliography

[1] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski. Building rome in a day. *Commun. ACM*, 54(10):105–112, October 2011. 14, 16, 18, 35

[2] S. Agarwal, K. Mierle, et al. Ceres solver. http://ceres-solver.org, 2010. 21, 29, 31, 32

[3] R. Arandjelović and A. Zisserman. Efficient image retrieval for 3D structures. In *BMVC*, 2010. 11

[4] R. Arandjelović and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, 2012. 10

[5] M. Arie-Nachimson, S. Z. Kovalsky, I. Kemelmacher-Shlizerman, A. Singer, and R. Basri. Global motion estimation from point matches. In *3DIMPVT*, pages 81–88, 2012. 14, 15, 16

[6] Y. Avrithis and G. Tolias. Hough pyramid matching: Speeded-up geometry re-ranking for large scale image retrieval. *IJCV*, 107(1):1–19, March 2014. 11

[7] D. H. Ballard. Generalizing the hough transform to detect arbitrary shapes. In Martin A. Fischler and Oscar Firschein, editors, *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms*, pages 714–725. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1987. 11

[8] P. Bhat, K. C. Zheng, N. Snavely, A. Agarwala, M. Agrawala, M. F. Cohen, and B. Curless. Piecewise image registration in the presence of multiple large motions. In *CVPR*, volume 2, pages 2491–2497, 2006. 12, 16, 18, 35, 37

[9] A. S. Brahmachari and S. Sarkar. Hop-diffusion monte carlo for epipolar geometry estimation between very wide-baseline images. *PAMI*, 35(3):755–762, 2013. 11

[10] A. Chatterjee and V. M. Govindu. Efficient and robust large-scale rotation averaging. In *ICCV*, pages 521–528, Dec 2013. 14, 15

[11] H.-I Chen, Y.-Y. Lin, and B.-Y. Chen. Co-segmentation guided hough transform for robust feature matching. *PAMI*, 37(12):2388–2401, 2015. 11

[12] A. M. Cheriyadat and R. J. Radke. Non-negative matrix factorization of partial track data for motion segmentation. In *ICCV*, pages 865–872, Sept 2009. 13

[13] T.-J. Chin, J. Yu, and D. Suter. Accelerated hypothesis generation for multistructure data via preference analysis. *PAMI*, 34(4):625–638, April 2012. 11

[14] M. Cho, Y. M. Shin, and K. M. Lee. Co-recognition of image pairs by data-driven monte carlo image exploration. In *ECCV*, pages 144–157, 2008. 36, 41

[15] O. Chum and J. Matas. Matching with prosac progressive sample consensus. In *CVPR*, pages 220–226, 2005. 11, 32

[16] O. Chum, J. Matas, and S. Obdrzalek. Enhancing ransac by generalized model optimization. In *ACCV*, volume 2, pages 812–817, 2004. 11, 20, 32

[17] O. Chum, T. Werner, and J. Matas. Two-view geometry estimation unaffected by a dominant plane. In *CVPR*, volume 1, pages 772–779 vol. 1, June 2005. 11

[18] D. Crandall, A. Owens, N. Snavely, and D. Huttenlocher. Discrete-continuous optimization for large-scale structure from motion. In *CVPR*, 2011. 14, 16, 18, 35

[19] Z. Cui and P. Tan. Global structure-from-motion by similarity averaging. In *ICCV*, December 2015. 14

[20] E. Elhamifar and R. Vidal. Sparse subspace clustering. In *CVPR*, 2009. 14

[21] P. Favaro, R. Vidal, and A. Ravichandran. A closed form solution to robust subspace estimation and clustering. In *CVPR*, pages 1801–1807, June 2011. 13

[22] J. Fayad, C. Russell, and L. Agapito. Automated articulated structure and 3d shape recovery from point correspondences. In *ICCV*, pages 431–438, Nov 2011. 15

[23] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981. 10, 20, 29, 32, 37

[24] D. Fouhey, D. Scharstein, and A. Briggs. Multiple plane detection in image pairs using J-linkage. In *ICPR*, pages 336 – 339, 2010. 12

[25] V. Fragoso and M. Turk. Swigs: A swift guided sampling method. In *CVPR*, pages 2770–2777, June 2013. 10

[26] J.-M. Frahm, P. Fite-Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys. Building rome on a cloudless day. In *ECCV*, pages 368–381, 2010. 14

[27] L. Goshen and I. Shimshoni. Balanced exploration and exploitation model search for efficient epipolar geometry estimation. *PAMI*, 30(7):1230–1242, 2008. 11

[28] G. Guennebaud, B. Jacob, et al. Eigen v3. `http://eigen.tuxfamily.org`, 2010. 4, 30

[29] R. I. Hartley, K. Aftab, and J. Trumpf. L1 rotation averaging using the weiszfeld algorithm. In *CVPR*, pages 3041–3048, June 2011. 15

[30] R. I. Hartley, J. Trumpf, Y. Dai, and H. Li. Rotation averaging. *IJCV*, 103(3):267–305, 2013. 15

[31] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004. 4, 6, 7, 8, 9, 29, 33

[32] J. Heinly, J. L. Schonberger, E. Dunn, and J.-M. Frahm. Reconstructing the world* in six days *(as captured by the yahoo 100 million image dataset). In *CVPR*, June 2015. 14

[33] H. Isack and Y. Boykov. Energy-based geometric multi-model fitting. *IJCV*, 97(2):123–147, April 2012. 12

[34] B. Jacquet, R. Angst, and M. Pollefeys. Articulated and restricted motion subspaces and their signatures. In *CVPR*, 2013. 13

[35] B. Jacquet, C. Häne, R. Angst, and M. Pollefeys. Multi-body depth-map fusion with non-intersection constraints. In *ECCV*, pages 735–750. Springer International Publishing, 2014. 15

[36] M. Jancosek and T. Pajdla. Multi-view reconstruction preserving weakly-supported surfaces. In *CVPR*, pages 3121–3128, 2011. 45, 47

[37] N. Jiang, Z. Cui, and P. Tan. A global linear method for camera pose registration. In *ICCV*, pages 481–488, Dec 2013. 14, 15

[38] H. Jung, J. Ju, and J. Kim. Rigid motion segmentation using randomized voting. In *CVPR*, pages 1210–1217, June 2014. 13

[39] J. Klappstein, T. Vaudrey, C. Rabe, A. Wedel, and R. Klette. Moving object segmentation using optical flow and depth information. In *PSIVT*, pages 611–623, 2008. 13

[40] J. l. Gailly and M. Adler. zlib compression library. `http://www.zlib.net/`, 1995. 31

[41] H. Li and R. Hartley. Five-point motion estimation made easy. In *ICPR*, pages 630–633, August 2006. `http://users.cecs.anu.edu.au/~hartley/Software/5pt-6pt-Li-Hartley.zip`. 29, 31, 32

[42] X. Li, C. Wu, C. Zach, S. Lazebnik, and J.-M. Frahm. Modeling and recognition of landmark image collections using iconic scene graphs. In *ECCV*, pages 427–440, 2008. 14

[43] Z. Li, J. Guo, L. F. Cheong, and S. Z. Zhou. Perspective motion segmentation via collaborative clustering. In *ICCV*, pages 1369–1376, Dec 2013. 13

[44] W. Liu, Y. Wang, J. Chen, J. Guo, and Y. Lu. A completely affine invariant image-matching method based on perspective projection. *Mach. Vis. Appl.*, 23(2):231–242, 2012. 10

[45] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 3, 10, 11, 16, 17

[46] L. Magri and A. Fusiello. T-linkage: A continuous relaxation of j-linkage for multi-model fitting. In *CVPR*, June 2014. 12

[47] D. Martinec and T. Pajdla. Robust rotation and translation estimation in multiview reconstruction. In *CVPR*, pages 1–8, June 2007. 15

[48] J. Matas and O. Chum. Randomized ransac with sequential probability ratio test. In *ICCV*, volume 2, pages 1727–1732 Vol. 2, Oct 2005. 11

[49] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *BMVC*, pages 36.1–36.10, 2002. 3, 10

[50] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *IJCV*, 60(1):63–86, October 2004. 10

[51] D. Mishkin, J. Matas, and M. Perdoch. MODS: Fast and robust method for two-view matching. *CVIU*, 141:81 – 93, 2015. 10

[52] Q. Mo and B. A. Draper. Semi-nonnegative matrix factorization for motion segmentation with missing data. In *ECCV*, pages 402–415, 2012. 13

[53] J.-M. Morel and G. Yu. Asift: A new framework for fully affine invariant image comparison. *SIAM J. Img. Sci.*, 2(2):438–469, April 2009. 10, 16

[54] P. Moulon, P. Monasse, and R. Marlet. Global fusion of relative motions for robust, accurate and scalable structure from motion. In *ICCV*, December 2013. 14, 15

[55] M. Muja and D. G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *PAMI*, 36(11):2227–2240, 2014. http://www.cs.ubc.ca/research/flann/. 3, 17, 31

[56] M. Narayana, A. Hanson, and E. Learned-Miller. Coherent motion segmentation in moving camera videos using optical flow orientations. In *ICCV*, pages 1577–1584, Dec 2013. 13

[57] D. Nistér. An efficient solution to the five-point relative pose problem. *PAMI*, 26(6):756–777, June 2004. 32

[58] P. Ochs, J. Malik, and T. Brox. Segmentation of moving objects by long term video analysis. *PAMI*, 36(6):1187–1200, 2014. 13

[59] K. E. Ozden, K. Schindler, and L. Van Gool. Multibody structure-from-motion in practice. *PAMI*, 32(6):1134–1141, June 2010. 15

[60] T. Pajdla. Elements of geometry for computer vision. University lectures: http://cmp.felk.cvut.cz/~pajdla/gvg/GVG-2015-Lecture.pdf, 2015. 22

[61] Y. Pang, W. Li, Y. Yuan, and J. Pan. Fully affine invariant surf for image matching. *Neurocomput.*, 85:6–10, May 2012. 10

[62] T. T. Pham, T. J. Chin, J. Yu, and D. Suter. The random cluster model for robust geometric fitting. *PAMI*, 36(8):1658–1671, Aug 2014. 11

[63] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, 2007. 4, 11

[64] B. Poling and G. Lerman. A new approach to two-view motion segmentation using global dimension minimization. *IJCV*, 108(3):165–185, July 2014. 13, 36

[65] J. Pritts, O. Chum, and J. Matas. Approximate models for fast and accurate epipolar geometry estimation. In *IVCNZ*, pages 106–111, Nov 2013. 11

[66] R. Raguram, O. Chum, M. Pollefeys, J. Matas, and J.-M. Frahm. Usac: A universal framework for random sample consensus. *PAMI*, 35(8):2022–2038, 2013. 10

[67] R. Raguram and J.-M. Frahm. Recon: Scale-adaptive robust estimation via residual consensus. In *ICCV*, pages 1299–1306, 2011. 11

[68] S. Rao, R. Tron, R. Vidal, and Y. Ma. Motion segmentation in the presence of outlying, incomplete, or corrupted trajectories. *PAMI*, 32(10):1832–1845, October 2010. 14

[69] S. R. Rao, A. Y. Yang, S. S. Sastry, and Y. Ma. Robust algebraic segmentation of mixed rigid-body and planar motions from two views. *IJCV*, 88(3):425–446, 2010. 12, 36

[70] A. Roussos, C. Russell, R. Garg, and L. Agapito. Dense multibody motion estimation and reconstruction from a handheld camera. In *ISMAR*, pages 31–40, 2012. 15

[71] C. Rubino, M. Crocco, V. Murino, and A. D. Bue. Semantic multi-body motion segmentation. In *WACV*, pages 1145–1152, Jan 2015. 13

[72] C. Russell, R. Yu, and L. Agapito. Video pop-up: Monocular 3d reconstruction of dynamic scenes. In *ECCV*, pages 583–598, 2014. 15

[73] R. Sara and M. Matousek. 3D computer vision. University lectures: `http://cmp.felk.cvut.cz/cmp/courses/TDV/2015W/lectures/tdv-2015-all.pdf`, 2015. 21, 32

[74] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. In *SIGGRAPH*, pages 835–846, 2006. 14, 16, 18, 27, 28, 35, 37

[75] C. Sweeney, T. Sattler, T. Hollerer, M. Turk, and M. Pollefeys. Optimizing the viewing graph for structure-from-motion. In *ICCV*, December 2015. 14, 15

[76] C. Tao, H. Sun, C. Yang, and J. Tian. Efficient image stitching in the presence of dynamic objects and structure misalignment. *JSIP*, 2(3):205–210, 2011. 12, 16, 37

[77] R. B. Tennakoon, A. Bab-Hadiashar, Z. Cao, R. Hoseinnezhad, and D. Suter. Robust model fitting using higher than minimal subset sampling. *PAMI*, 38(2):350–362, 2016. 11

[78] R. Toldo and A. Fusiello. Robust multiple structures estimation with j-linkage. In *ECCV*, pages 537–547, 2008. 12

[79] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, pages 298–372, 2000. 14, 15, 29

[80] R. Tron and R. Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. In *CVPR*, pages 1–8, June 2007. 4, 36, 41

[81] A. Vedaldi and A. Zisserman. Object instance recognition practical. `https://github.com/vedaldi/practical-object-instance-recognition`, 2011. 4, 11, 19, 20, 38

[82] R. Vidal and R. Hartley. Three-view multibody structure from motion. *PAMI*, 30(2):214–227, Feb 2008. 13

[83] R. Vidal, Y. Ma, S. Soatto, and S. Sastry. Two-view multibody structure from motion. *IJCV*, 68(1):7–25, 2006. 12

[84] R. Vidal, R. Tron, and R. Hartley. Multiframe motion segmentation with missing data using powerfactorization and gpca. *IJCV*, 79(1):85–105, August 2008. 13

[85] M. Wandel. jhead: exif jpeg header manipulation tool. http://www.sentex.net/~mwandel/jhead/, 1999. 30

[86] T. Y. Wang, P. Kohli, and N. J. Mitra. Dynamic sfm: Detecting scene changes from image pairs. In *SGP*, 2015. 15, 16, 36, 41

[87] K. Wilson and N. Snavely. Robust global translations with 1dsfm. In *ECCV*, 2014. 14, 15

[88] H. S. Wong, T.-J. Chin, J. Yu, and D. Suter. A simultaneous sample-and-filter strategy for robust multi-structure model fitting. *CVIU*, 117(12):1755 – 1769, 2013. 11

[89] D. Woods, N. Weber, and M. Dario. DevIL: developer's image library. http://openil.sourceforge.net/, 2002. 30

[90] C. Wu. SiftGPU: A GPU implementation of scale invariant feature transform (SIFT). http://cs.unc.edu/~ccwu/siftgpu, 2007. 31

[91] C. Wu. Towards linear-time incremental structure from motion. In *3DV*, pages 127–134, 2013. 14, 27

[92] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz. Multicore bundle adjustment. In *CVPR*, pages 3057–3064, June 2011. 27

[93] L. Zappella, A. Del Bue, X. Lladó, and J. Salvi. Joint estimation of segmentation and structure from motion. *CVIU*, 117(2):113–129, 2013. 15

[94] Y. Zhang, Z. Sun, R. He, and T. Tan. Robust subspace clustering via half-quadratic minimization. In *ICCV*, pages 3096–3103, Dec 2013. 14

[95] V. Zografos and K. Nordberg. Fast and accurate motion segmentation using linear combination of views. In *BMVC*, pages 12.1–12.11, 2011. 13

[96] M. Zuliani, C. S. Kenney, and B. S. Manjunath. The multiransac algorithm and its application to detect planar homographies. In *ICIP*, pages 153–156, 2005. 12