TECHNISCHE UNIVERSITÄT
BERGAKADEMIE FREIBERG

The University of Resources. Since 1765.

Structureless Camera Motion Estimation of Unordered Omnidirectional Images

By the Faculty of Mathematics and Computer Science
of the Technische Universität Bergakademie Freiberg

approved

Thesis

to attain the academic degree of

Doktor-Ingenieur
(Dr.-Ing.)

submitted by **Dipl.-Ing. Mark Sastuba**

born on the 28th June 1985 in Lauchhammer

Assessor: **Prof. Dr.-Ing. habil. Bernhard Jung**
Prof. Dr.-Ing. Danilo Schneider

Date of the award: **Freiberg, 10th June 2022**

Declaration

I hereby declare that I completed this work without any improper help from a third party and without using any aids other than those cited. All ideas derived directly or indirectly from other sources are identified as such.

In the selection and in the use of materials and in the writing of the manuscript I received support from the following persons:

Professor Dr.-Ing. habil. Bernhard Jung

Persons other than those above did not contribute to the writing of this thesis. I did not seek the help of a professional doctorate-consultant. Only persons identified as having done so received any financial payment from me for any work done for me.

This thesis has not previously been submitted to another examination authority in the same or a similar form in Germany or abroad.

Freiberg, 13th December 2021

Mark Sastuba

Abstract

This work aims at providing a novel camera motion estimation pipeline from large collections of unordered omnidirectional images. In oder to keep the pipeline as general and flexible as possible, cameras are modelled as unit spheres, allowing to incorporate any central camera type. For each camera an unprojection lookup is generated from intrinsics, which is called P2S-map (Pixel-to-Sphere-map), mapping pixels to their corresponding positions on the unit sphere. Consequently the camera geometry becomes independent of the underlying projection model. The pipeline also generates P2S-maps from world map projections with less distortion effects as they are known from cartography. Using P2S-maps from camera calibration and world map projection allows to convert omnidirectional camera images to an appropriate world map projection in oder to apply standard feature extraction and matching algorithms for data association.
The proposed estimation pipeline combines the flexibility of SfM (Structure from Motion) - which handles unordered image collections - with the efficiency of PGO (Pose Graph Optimization), which is used as back-end in graph-based Visual SLAM (Simultaneous Localization and Mapping) approaches to optimize camera poses from large image sequences. SfM uses BA (Bundle Adjustment) to jointly optimize camera poses (motion) and 3d feature locations (structure), which becomes computationally expensive for large-scale scenarios. On the contrary PGO solves for camera poses (motion) from measured transformations between cameras, maintaining optimization managable. The proposed estimation algorithm combines both worlds. It obtains up-to-scale transformations between image pairs using two-view constraints, which are jointly scaled using trifocal constraints. A pose graph is generated from scaled two-view transformations and solved by PGO to obtain camera motion efficiently even for large image collections. Obtained results can be used as input data to provide initial pose estimates for further 3d reconstruction purposes e.g. to build a sparse structure from feature correspondences in an SfM or SLAM framework with further refinement via BA.
The pipeline also incorporates fixed extrinsic constraints from multi-camera setups as well as depth information provided by RGBD sensors. The entire camera motion estimation pipeline does not need to generate a sparse 3d structure of the captured environment and thus is called SCME (Structureless Camera Motion Estimation).

Acknowledgements

First, I would like to thank Prof. Dr. Jung who placed his trust in me and made me a part of his team, although I had no clue about robotics or computer science. Thank you for giving me the freedom to research the things I was interested in and for allowing me to try my own approaches, even if that meant that my equipment was a little more expensive. Thank you so much for the great trust you placed in me. Without all these possibilities, I would probably never have gotten so interested in computer vision and robotics. As it turns out, I ended up taking a completely different research path compared to what I originally studied.

Many thanks to the entire team, with who I was lucky to work with for many years. Thanks to every single team member for being so reliable. In particular, I would like to thank Marc Donner for providing me with a lot of data and for helping me with his programming wisdom. Thank you to Robert Lösch, who was always willing to help me, even if I only had a small request. You gave me a lot of useful feedback and always offered me help to find the right approach. Thanks also to Steve Grehl for the numerous conversations we had to discuss new insights and look for solutions. Thank you to Christian Schubert, who kept the IT running and took care of one or the other service request and who always took some time for a cup of coffee. Finally, I want to thank Mrs. Schüttauf for always keeping track of everything even in stressful situations and for always sharing a few nice words.

Moreover, I want to thank my parents and my in-laws for the support and the encouraging messages. Thank you so much!

The biggest thank you, however, goes out to the three most important people in my life.

Thank you to my children Katharina and Maximilian for being so understanding whenever I had to work on this project, but also for helping me to clear my head and for giving me the chance to look at the world from a different perspective.

Lastly, I want to thank my amazing wife Barbara for supporting me in all imaginable ways, for encouraging me and giving me the faith to complete this work. Without your help, I would not have been able to write this dissertation.

Contents

Abbreviations

BA	Bundle Adjustment
BRIEF	Binary Robust Independent Elementary Features
BRISK	Binary Robust Invariant Scalable Keypoints
Dl	Dogleg (non-linear solver)
DLT	Direct Linear Transform
DoF	Degree of Freedom
FAST	Features from Accelerated Segmented Test
FoV	Field of View
GN	Gauss-Newton (non-linear solver)
GNSS	Global Navigation Satellite System
GTSAM	Georgia Tech Smoothing and Mapping
GUI	Graphical User Interface
HDF	Hierarchial Data Format
HDR	High Dynamic Range
HMD	Head Mounted Display
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
iSAM	incremental Scanning and Mapping
KAZE	Japanese word for *Wind*
LIDAR	Light Detection and Ranging
LM	Levenberg-Marquardt (non-linear solver)
LoS	Line of Sight
LZW	Lempel–Ziv–Welch, lossless data compression algorithm
MAP	Maximum a Posteriori
MASAT	Multi-Ancestor Spatial Approximation Tree
MVS	Multi-View Stereo
ORB	Oriented FAST and Rotated BRIEF

PCM	Polynomal Camera Model
PGO	Pose Graph Optimization
PLY	Polygon File Format
PNG	Portable Network Graphics
PnP	Perspective-n-Point
RANSAC	Random Sample Consensus
RGB	Red Green Blue
RGBD	Red Green Blue Depth
ROS	Robot Operating System
SCM	Spherical Camera Model
SfM	Structure from Motion
SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SCME	Structureless Camera Motion Estimation
SURF	Speeded Up Robust Features
SVD	Singular Value Decomposition
TIFF	Tag Image File Format
UCM	Unified Camera Model
P2S-map	Pixel-to-Sphere-map
VIO	Visual Inertial Odometry
VO	Visual Odometry
VR	Virtual Reality

Nomenclature

Symbol	Description
A	stacking matrix
$\mathbf{a}$	vector
α	yaw angle
B	stacking matrix
$\mathbf{b}$	vector
β	pitch angle
$\mathcal{C}$	camera frame
$\mathbf{c}$	vector
$\mathbf{d}$	vector
δ	angle
δ	measured depth value
E	essential matrix
$\mathcal{E}$	edge set
Σ	diagonal matrix from SVD
ϵ	error/distance
$\mathbf{g}$	vector
γ	angle
γ	roll angle
H	homography matrix
I	identity matrix
i	counting variable
J	Jacobian matrix
j	counting variable
k	counting variable
l	counting variable
λ	depth value
$\tilde{\lambda}$	approximate/unscaled depth value
m	counting variable/quantity/size of dimension
n	quantity/size of dimension
$\mathbf{0}$	null vector
Ω	information matrix
p	quantity
ϕ	azimuth angle ($0° \leq \phi \leq 360°$)
φ	angle

Symbol	Description
$\boldsymbol{\pi_e}$	epipolar plane
$\boldsymbol{\pi_h}$	homography plane
$\boldsymbol{\pi_i}$	image plane
$\boldsymbol{\pi_p}$	pixel plane
R	rotation matrix
r	ratio factor
$\mathcal{S}$	camera sphere, subspace of $\mathcal{C}$
S	covariance matrix
s	scale factor
σ	singular value/standard deviation
$\mathbf{t}$	translation vector
$\hat{\mathbf{t}}$	normalized translation vector, $\|\hat{\mathbf{t}}\| = 1$
θ	polar angle ($0° \leq \theta \leq 180°$)
U	unitary left matrix from SVD
$\mathbf{U}$	3d point in world frame $\mathcal{W}$, $\mathbf{U} = (U, V, W)^T$
V	unitary right matrix from SVD
$\mathcal{W}$	world frame
$\mathbf{X}$	3d point in camera frame $\mathcal{C}$, $\mathbf{X} = (X, Y, Z)^T$
$\mathring{\mathbf{X}}$	rotated point on unit sphere $\mathcal{S}$, $\mathring{\mathbf{X}} = (\mathring{X}, \mathring{Y}, \mathring{Z})^T$, $\|\mathring{\mathbf{X}}\| = 1$
$\hat{\mathbf{X}}$	projected point on unit sphere $\mathcal{S}$, $\hat{\mathbf{X}} = (\hat{X}, \hat{Y}, \hat{Z})^T$, $\|\hat{\mathbf{X}}\| = 1$

List of Figures

List of Tables

1 Introduction

This chapter explains the different motivation points for this work, describes current issues of 3d reconstruction in challenging underground environments and gives a brief overview of novelties, implemented in the proposed camera motion estimation pipeline.

1.1 Motivation

This work is motivated by various reasons. First, there is an increasing demand for 3d reconstruction of existing facilities, buildings and places in order to accelerate monitoring, processing and maintaining tasks in industry using state-of-the-art augmented and virtual reality technology. Image based 3d reconstruction techniques such as SfM (Structure from Motion, Section 2.2, page 19) and Visual SLAM (Simultaneous Localization and Mapping, Section 2.1, page 14) have been improved and implemented in commercial products or made available via freeware as well as open-source repositories.

Second, further improvements of image-based omnidirectional sensor systems allow fast and convenient capturing for localization and mapping purposes even under harsh conditions.

Third, underground environments are still a challenging application field of 3d reconstruction since reliable localization techniques from aboveground tend to fail in these areas. They represent an interesting field of research in order to test novel localization and mapping techniques and hence are suitable to simulate and practice military cases and disaster scenarios.

Fourth, there is an increasing demand to minimize the time needed for measuring campaigns in industry. Distributed and non-chronological data acquisition allows faster mapping and reduces downtime, since data acquisition can be performed, when is doesn't influence certain processes.

1.1.1 Increasing Interest of Image-Based 3D Reconstruction

During the last decade, image-based localization and 3d reconstruction have become an increasing impact in computer vision, photogrammetry and robotics to exploit new applications in the fields of:

- selfdriving cars [90]
- indoor mapping[1] and navigation[2]
- land surveying[3]
- agriculture and forestry[4] [112]
- real estate[5][6]
- construction and civil engineering[7]
- medicine [271]
- film and entertainment[8]
- computer games[9]
- virtual[10] and augmented reality [42]
- forensics[11]
- digital heritage conservation[12]
- disaster scenarios[13]
- open pit [145] and underground mining [82]

Within most of these fields robust and reliable software tools have been developed. They enable users to see certain situations and scenarios from new perspectives and extend the group of experts contributing their solutions to complex issues, even if they are not on-site.

1.1.2 Underground Environments as Challenging Scenario

Underground environments still present a challenging scenario, which arouses increasing interest of industry and military[14].
Obtained 3d reconstructions from undergound environements such as mines are used for augmented and virtual reality applications for show cases, provide ground truth map data for robotics, contribute to digital heritage conservation and improve construction, maintenance and inspection sevices.

In [81, 169] the authors describe the environmental characteristics of an old ore mine, which influence different sensors used for navigation and mapping. The here presented camera motion estimation pipeline uses image data from the same mine. Underground areas have no access to GNSS (Global Navigation Satellite System),

[1] https://3dsurvey.si/case-studies/indoor-mapping-of-a-house-with-a-phone-camera
[2] https://dragonflycv.com
[3] https://www.pix4d.com/blog/mapping-faroe-islands
[4] https://tu-dresden.de/bu/umwelt/geo/ipf/photogrammetrie/forschung/forschungsfelder/forstwesen
[5] https://www.oddviz.com/work/hotels
[6] https://rooomy.com/rooomy-virtually-staged-matterport-3d-tours
[7] https://geo-matching.com/content/photogrammetry-as-a-tool-for-forensic-documentation
[8] https://www.capturingreality.com/RealityCapture-In-Ghost-In-The-Shell
[9] https://unity.com/solutions/photogrammetry
[10] https://kenwang-57215.medium.com/oculus-solution-to-room-scale-vr-37b2ff654dc9
[11] https://geo-matching.com/content/photogrammetry-as-a-tool-for-forensic-documentation
[12] http://culturalheritageimaging.org/Technologies/Photogrammetry/
[13] https://www.pix4d.com/blog/beirut-disaster-response
[14] https://www.subtchallenge.com/index.html

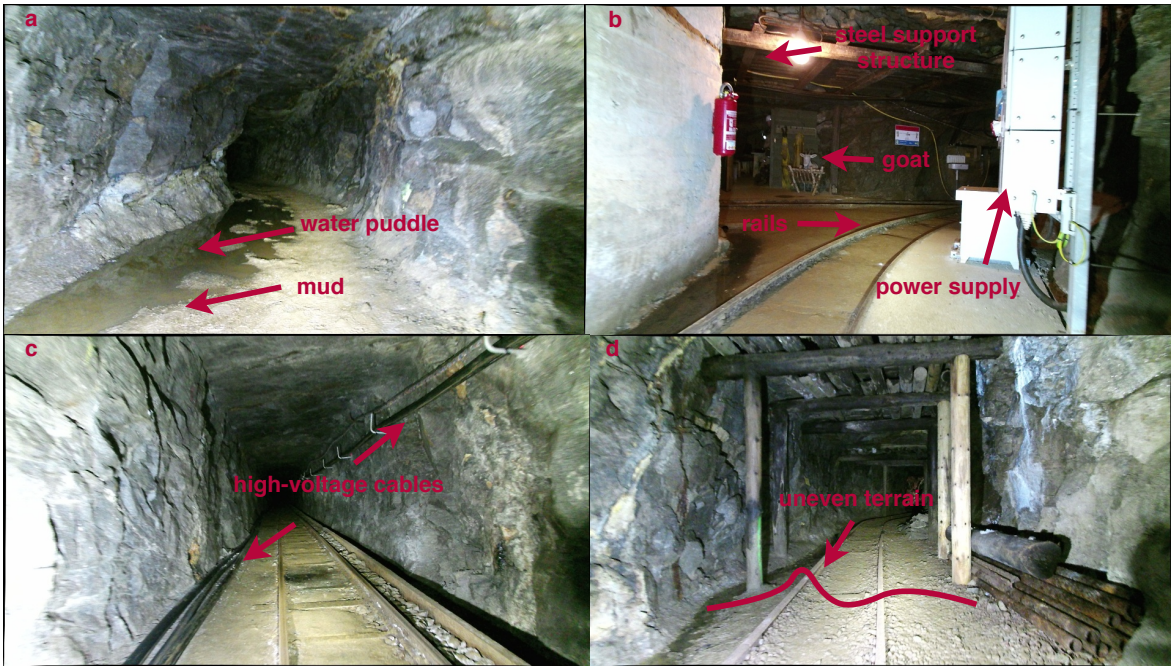

Fig. 1.1: Environmental undergorund characteristics of an old ore mine. **a)** Water puddles interfere with infrared radiation and mud influences wheel odometry. **b, c)** Steel support structures, rails and power supplies as well as high-voltag cables disturb magnetic field sensors. **d)** Uneven terrain affects laser-based scan machting techniques.

which is typically used as external referencing system aboveground. Orientation estimation based on IMU (Inertial Measurement Unit) may lead to erroneous results since the magnetic field is locally disturbed by rails, steel supporting structures or high-voltage cables and other power supply systems as shown in Fig. 1.1. Matching techniques based on laser scans are affected by uneven terrain and rough surfaces, which lead to rapid orientation changes of the scanning planes. Mud leads to wheel slip, which influences odometry and water puddles on the ground interfere with infrared radiation, limiting the use of range sensors, e.g. LIDAR (Light Detection and Ranging). Due to darkness, image based sensors require a lighting system since external illumination isn't available in all areas. However standard cameras have a limited FoV (Field of View). Due to the required lighting system, there is only a reduced viewing distance (ca. 5m). Combining both aspects leads to shortened feature tracks and hence yield erroneous camera pose estimates. Sometimes even feature loss might occure within consecutive images.
Finally, it should to be mentioned that sensors must be protected against dust, spray water and high humidity (up to 95%), which leads to increasing costs.

Fig. 1.2: Selection of available 360° action cameras on market (October 2021), e.g. *Insta360 One R*[15], *GoPro Max*[16], *Garmin VIRB 360*[17], *Ricoh Theta Z1*[18], *Insta360 ONE X2*[19], *VUZE XR*[20]and *KANDAO QooCam 8K*[21].

1.1.3 Improved Mobile Camera Systems for Full Omnidirectional Imaging

Feature-based image localization is still a promising approach for underground environments. During recent years, optics as well as image sensor technology have been improved and raised the level of available image resolution, FoV (Field of View), dynamic range, image noise reduction and frame rate. Initiated by increasing popularity of VR (Virtual Reality) content such as 360° videos through internet media plattforms and HMDs (Head Mounted Display), manufacturers started providing cameras with multi-fisheye lenses to the consumer market, which stitch full omnidirectional (360° × 180°) images. More on this topic describes Section 5.1, page 75.
This new generation of cameras can be remotely controlled and provide - depending on the camera model - the following features: video streaming and on-the-fly stichting, HDR (High Dynamic Range) still image capturing, dynamic exposure adaption, reduced image noise at low light level and built-in image stabilization, to name only a few features. They provide an increased optical resolution for videos and sill image captures up to $7680pix \times 3840pix$ (*KANDAO QooCam 8K*), which sometimes outperforms the angular resolution of standard directional cameras. These imaging devices

[15] https://www.insta360.com/de/product/insta360-oner_twin-edition/
[16] https://gopro.com/de/de/shop/cameras/max/CHDHZ-202-master.html
[17] https://buy.garmin.com/de-DE/DE/p/562010
[18] https://theta360.com/en/about/theta/z1.html
[19] https://www.insta360.com/de/product/insta360-onex2
[20] https://vuze.camera/camera/vuze-xr-camera
[21] https://www.kandaovr.com/qoocam-8k/index.html

are also referred as action cameras since they are especially designed to capture action sports like diving, surfing, mountain biking, skiing, parachuting and they even crossed the boarder to space[22]. They represent flexible and robust omnidirectional capturing devices at affordable prices (400€-1600€). A collection of selected cameras is shown in Fig. 1.2. These cameras meet the requirements to withstand the described underground conditions and capture full omnidirectional images and videos in order to improve feature matching and tracking.

1.2 Issues

This section briefly describes issues of image-based data acquisition and 3d reconstruction in underground environments.

1.2.1 Directional versus Omnidirectional Image Acquisition

Fig. 1.3: Image acquisition issues in underground environments such as **a)** motion blur due to fast camera movement and missing image stabilization, **b)** over- and underexposure caused by non-adaptive shutter and **d)** low dynamic range leads to an overall dark background without visible shades and causes bright light spots in the foreground.

Standard cameras have a limited FoV and capture the environment from a certain perspective only. As a consequence in underground, image alignment yields incorrect transformations or may fail in case of motion blur, under- and overexposure and low dynamic range as exemplary illustrated in Fig. 1.3. Low dynamic range hampers feature detection, since the camera is unable to capture extreme brights and darks. Hence distant areas as well as areas being close to the lighting system cannot be used for image processing. Moving obstacles like personnel or structureless surfaces also complicate feature tracking and thus the process of transformation estimation. However, structureless surfaces are uncommon for underground environments, but they play an important role for indoor navigation and mapping e.g. in office buildings with long monotonic white corridors.

Matching images from opposite directions (back and forth camera motions) is

[22] https://jalopnik.com/spend-over-an-hour-floating-in-space-thanks-to-nasa-1696865607

not straightforward and usually fails since features are observed from - more or less - complete different perspectives. Combining two or more cameras to increase the FoV in a fixed setup may solve this problem. However this requires sensor synchronization as well as an external calibration and the need to add fixed camera constraints to the reconstruction pipelines. Additional implementation effort is required, depending on the choosen algorithm.

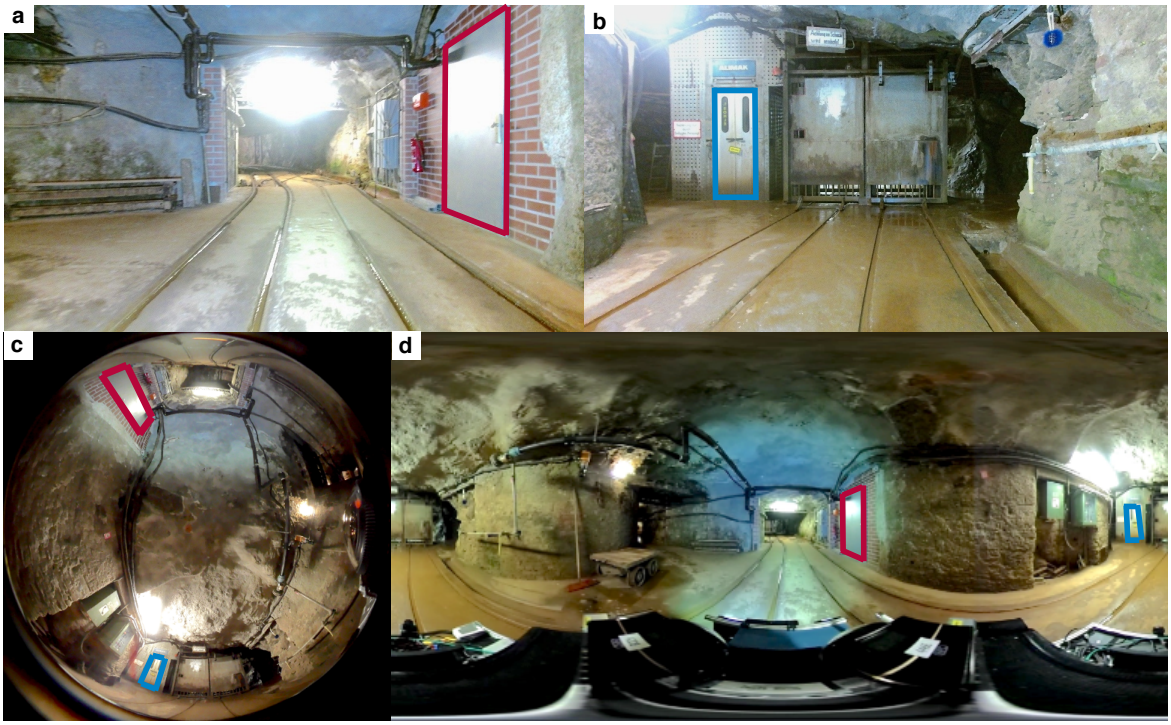

Fig. 1.4: An underground scene with labeled objects is captured by different cameras: **a)** image from a forward facing perspective camera (*Kinect v2*), **b)** image from a backward facing perspective camera (*Kinect v2*), **c)** omnidirectional image from fisheye lens (*Kodak SP360 4K*), **d)** full omnidirectional image from a stichting camera (*Ricoh Theta S*). The corresponding FoV of each camera is illustrated in Fig. 4.4, page 57.

Omnidirectional cameras widen the FoV up to 360° × 180° and allow feature tracking over longer image sequences, which is necessary in real-time Visual SLAM. Consequently, feature tracking depends less on camera orientation, which makes image alignment more robust. Omnidirectional cameras have no real front or back and thus they even enable feature matching from opposite directions. This circumstance is especially beneficial for loop closure detection in Visual SLAM or post-processing applications like SfM, which now are able to match images from different camera positions regardless their orientation. Fig. 1.4 illustrates the advantage of omnidirectional cameras over perspective ones observing objects from opposite directions within a single capture. However omnidirectional image projections - as they are presented in Fig. 1.4**c, d** - are affected by strong visible distortions, which interfere with feature matching.

Omnidirectional images are also affected by motion blur, over- and underexposure, insufficient dynamic range, moving obstacles or structureless surfaces. However these influcences are mostly limited to a certain area of the image and thus are less disturbing.
In order obtain realistic, detailed 3d reconstructions from underground environments, surface structures must be captured from different camera views. Fig. 1.5**a)** illustrates a recommended scheme[23] for capturing surface structures like walls. Using perspective cameras this leads to a large amount of overlapping images and thus becomes an exhausting task if performed by humans, especially in narrow sections. Capturing images in the same direction as the camera moves reduces the number of images, but leads to incomplete reconstruction models due to occlusion as shown in Fig. 1.5**b)**. Omnidirectional cameras capture the surrounding environment at each position. They decrease the number of captured images while increasing the number of perspective views.

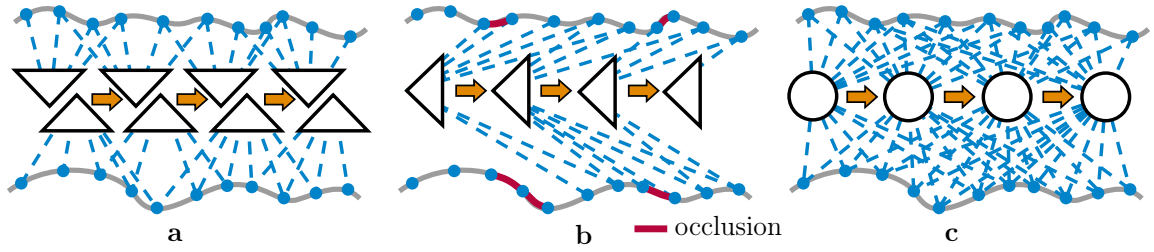

Fig. 1.5: Camera Motion for surface reconstruction. **a)** Recommended: Camera motion is orthogonal to the viewing direction in order to achieve good image alignment results and to capture all surface details from different perspectives for photorealistic 3d reconstrution. **b)** Actual: Camera motion and view are in the same direction. Surface details are only captured from a limited number of perspectives coming from the same direction, which leads to occlusions. **c)** Ideal: Omnidirectional cameras capture surface details from a wide range of perspectives, in this case independent of the camera motion.

1.2.2 Structure from Motion versus Visual Simultaneous Localization and Mapping

SfM is an image-based post-processing reconstruction pipeline. In contrast to that, Visual SLAM processes image data in real-time as it is required in robotics. On the one hand real-time applications allow the user to follow the reconstruction process in order to directly see the final result and allow to manually intervent in case of erroneous reconstruction. As experienced in serveral underground tests the forementioned image acquisition issues cause feature tracking loss, which disturbs Visual SLAM's real-time processing. Visual SLAM implementations provide relocalization capability, which is used in cases of lost feature tracks. Unfortunately, this seldomly succeeded and required to move the camera back to the position where feature loss first occured and to retry capturing from there again. Thus, image acquisition and reconstruction at

[23] https://www.3dflow.net/technology/documents/photogrammetry-how-to-acquire-pictures/

the same time became an overly time-consuming task, unnecessarily prolonging the expected time needed for measuring campaigns.

On the other hand post-processing gives more flexibility to the entire reconstruction process, however the final result can be seen only after image acquisition and reconstruction. It enables image pre-selection and modifications like histogram adjustment or masking certain areas of individual images as well as combining image series from different cameras and measuring compaigns. Thus, individual areas can be captured step-by-step at different times. This allows people to capture images more individually, precisely and whithout having to hurry, which is less intense compared to capturing a contineous image stream. In case of strong distortions - as it is typical for omnidirectional cameras - images can be converted to less distorted projections (Section 5.4, page 81) for data association and alignment purposes. Furthermore, images can be matched more robustly using an increased feature detection and different feature types (Chapter 10, page 161). They also can be matched to a larger quantity of neighboring images, making data association more robust against feature loss and thus yielding longer feature tracks for better alignment results.

Taking these mentioned facts into accout, a camera motion estimation pipeline based on post-processing like SfM is preferred. Until now, either SfM inplementations nor Visual SLAM ones comprehensively support full omnidirectional images. Chapter 2, page 12 takes a much closer look at image-based reconstruction techniques such as SfM (Section 2.2, page 19) and Visual SLAM (Section 2.1, page 14), their conceptual and mathematical backgrounds, as well as their behavior concerning image based underground reconstruction (Section 2.3, page 27).

1.3 Contribution

This work aims at providing an image feature-based, yet structureless camera pose estimation pipeline for unordered collections of full omnidirectional images. The pipeline is kept as general as possible and models cameras as spheres in order to incorporate a wide range of central camera types. SfM uses BA to jointly solve for 3d feature positions (structure) and cameras poses (motion), such that optimization problems quickly grow with the number of images and thus large-scale scenarios become computationally expensive. On the contrary Visual SLAM uses PGO as back-end, which solves for camera poses (motion) from a sequence of scaled two-view transformations, maintaining optimization tasks managable even in large-scale scenarios.

The here presented pipeline is called SCME (Structureless Camera Motion Estimation) and combines the flexibility of SfM to work with unordered images with the efficiency of PGO, but does not require any 3d reconstruction step. A missing link between SfM and PGO is that the former obtains up-to-scale two-view transformations

to verify feature matches for data association, while the latter requires scaled two-view transformations to solve for camera poses. SCME fills this gap. Like SfM it obtains up-to-scale two-view transformations between unordered image pairs but uses a novel technique to jointly scale them via trifocal constraints. The scaled two-view transformations are then solved by PGO to derive camera poses.

This work also introduces the concept of P2S-maps (Pixel-to-Sphere-map), which is a lookup that maps each pixel to its corresponding viewing direction as position on the unit sphere. By using a P2S-map the camera geometry is mapped onto unit sphere and hence becomes independent of individual projection models used for calibration. Consequently, SCME does not need to implement serveral camera projection models. A method is presented to convert between projections based on their P2S-maps only, e.g. in order to undistort omnidirectional images.

Using the idea of P2S-maps in combination with PGO enables the implementation of a flexible extrinsic calibration routine that is not restricted to neither a certain type of central camera nor to the quantity of cameras.

A detailed overview of novelties is given in Tab. 1.1, which lists each step of the camera motion estimation pipeline with identified current issues and proposed corresponding solutions.

Tab. 1.1: Current issues and proposed solutions for the new SCME pipeline

Issue	Proposed Solution
heterogeneity of camera/projection models	unit sphere as image domain • obtain unprojection from calibration, which maps each pixel onto unit sphere (Sections 3.2.1 and 3.2.2, pages 35 and 39) make camera geometry independent of camera model • save unprojection correspondences between pixel and unit sphere as lookup in P2S-map (Section 3.3, page 42)
strong distortions in omnidirectional images	convert images to less distortion affected projections • use world map projections from cartography (Section 5.2, page 78) • generate P2S-maps from world map projections (Section 5.3, page 79) • develop conversion between P2S-maps (Section 5.4, page 81)
selection of feature type	use multiple feature types jointly to improve robustness of image alignment (Chapter 10, page 161)
SfM is inefficient for large scale data sets and available implementations poorly support full omnidirectional images	convert SfM-like problem to be solved by PGO • obtain up-to-scale transformations between image pairs from two-view geometry (Chapter 6, page 87) • use three-view/trifocal geometry to scale derived two-view transformations (Chapter 7, page 125) • build and solve pose graph from unordered scaled two-view transformations (Chapter 8, page 139)
Additional	
SfM does not incorporate depth maps	scale two-view transformations to real world dimension using depth data (Section 6.7, page 113)
limited extrinsic camera calibration (restricted to certain camera types, models or number of cameras)	obtain camera extrinsics independently of camera model • use P2S-maps to incorporate different central camera types use PGO to solve for non-restricted number of camera and target poses • develop a 2d-2d and 3d-2d PnP method to obtain target-camera transformations (Sections 4.4.1 to 4.4.3, pages 60, 62 and 66) • build pose graph from derived target-camera transformations and solve for camera and target poses (Section 4.4.5, page 67) • refine poses using BA (Section 4.4.6, page 69)

1.4 Structure of this Work

Chapter 2 describes current image-based 3d reconstruction techniques such as Visual SLAM, SfM and VO, which are known in robotics, computer vision and photogrammetry. It further illustrates their relations to each other and explains the underlying optimization principles like BA, structureless BA and PGO. The proposed SCME pipeline is brought into this context, which is a combination of SfM and PGO.

Chapter 3 covers imaging related topics like camera types, their classifications and camera models. It also explaines an equation-free concept to describe the unprojection of any central projection as lookup, that maps pixels to their corresponding positions onto unit sphere. This lookup is saved as color image and called P2S-map.

Chapter 4 describes a calibration implementation to obtain intrinsics from different camera types and to generate corresponding P2S-maps. It introduces a new extrinsic calibration concept, that uses P2S-maps to incorporate a wide range of camera types independently of their underlying calibration model and uses PGO to solve for camera extrinsics in multi-camera setups.

Chapter 5 shows the generation of P2S-maps from selected world map projections, which are less affected by distortion. It proposes a new image conversion concept based on P2S-maps to convert camera images to an appropriate world map projection for undistortion purpose.

Chapter 6 focuses on point correspondences between two camera images. It explains different triangulation concepts, compares two-view estimation algorithms and presents a novel two-view optimization approach.

Chapter 7 concentrates on point correspondences between three camera images and explains major constraints, which are suitable to recover the translation ratio between up-to-scale two-view transformations.

Chapter 8 illustrates the integration of pose graph solvers as optimization backend to solve for camera poses from a set of unordered scaled two-view transformations.

Chapter 9 describes the proposed structureless camera motion estimation pipeline and explains the concept of global translation scaling based on derived translation ratios. It further illustrates how extrinsic camera constraints as well as depth data can be integrated into the pipeline.

Chapter 10 illustrates obtained SCME results for directional images as well as for omnidirectional images and compares them to results from SfM and Visual SLAM.

2 Related Work

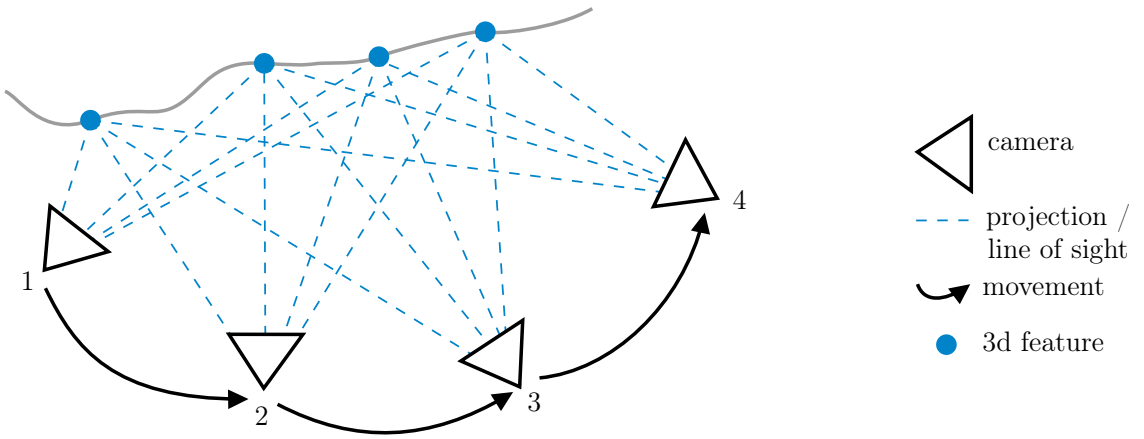

Fig. 2.1: Example case of a SLAM problem illustrating a moving monocular camera, which observes optical features in the surrounding environment.

The boundaries between Visual SLAM (Simultaneous Localization and Mapping), VO (Visual Odometry) and SfM (Structure from Motion) have become more diffuse since novel algorithms combine several ideas and functionalities from a wide spectrum of localization and mapping techniques, such that even terminology is sometimes mixed up.

There is a rich literature facing the topic of image-based localization and mapping techniques. The present discussion cannot describe individual implementations in detail, which would drastically extend the scope. Instead, each technique is explained as general as possible at hand of an example in order to clarify differences and similarities. Fig. 2.1 illustrates a simple example case of a moving monocular camera capturing optical features in the surrounding environment. Other sensor configurations - like stereo setups or RGBD cameras - are also possible to be used, but they belong to special cases and thus are not considered at this point. Having said previously that this work doesn't focus on real-time reconstruction techniques, it is however necessary to show how optimization processes like BA, which is known from SfM, and PGO, which is used as back-end in Visual SLAM, relate to each other in order to understand the novelties of the proposed SCME pipeline.

The SLAM problem is split into filtering and smoothing techniques [48] and can be described by three main probabilistic formulations based on *Kalman-filter* (EKF-SLAM [233]), particle-filter (FastSLAM [184], FastSLAM 2.0 [185, 26]) and a graph (GraphSLAM [252, 253]). Filter-based approaches are classified as *online* SLAM

[252] since they only estimate the current state of the camera pose and maybe parts of the map [83]. Thus they cannot change their past beliefs of the camera trajectory [272].

On the contrary graph-based approaches are classified as *full* SLAM [252]. Graph-based algorithms belong to smoothing techniques as they recover the MAP (Maximum a Posteriori) over all poses in the camera trajcetory and over all observed features in the environment [48]. They maintain the camera's trajectory within the state estimation problem, such that past and present states are simultaneously improved and do not suffer from inconsistency [129]. These kinds of algorithms apply non-linear optimization by exploiting the sparsity of the SLAM problem, allowing to solve large-scale problems [245].

A first graph-based formulation was proposed in [175] representing the state problem as a set of links between sensor poses and formulating a global optimization algorithm to obtain a map from such constraints [253]. Due to the comparably high complexity of solving the error minimization problem using standard techniques, filter-based approaches were preferred and more in focus of the research community. Graph-based methods experienced a renaissance as recent insights into the structure of the SLAM problem as well as advancements in the fields of sparse linear algebra resulted in efficient approaches to the optimization problem [83]. As a consequence in the last years the focus has clearly shifted from filter-based approaches towards graph-based ones [272] as efficient algorithms for solving the underlying optimization became available [245] (examples presented in Section 8.2, page 140). Nowadays these algorithms belong to the state-of-the-art techniques with respect to speed and accuracy.

There are different variants to model the SLAM problem in a graphical way such as Bayesian Belief Networks [198], factor graphs [149], Markov Random Fields [278] and Bayes trees [129]. For a deeper inside into graph-based SLAM representation approaches the reader is referred to [48, 83, 49].

Note on Terminology: In most SLAM literature landmarks denote destinct points in the surrounding environment, which are observed by a moving sensor system. A landmark can be described as a "recognizable natural or artificial feature [...], that stands out from its near environment and is often visible from long distances"[24].
Image-based algorithms like SfM use destinct points in the images, called image features. Feature detection algorithms search for interesting points of an image region, that has certain properties. Consequently image features do not necessarily correspond to recognizable landmarks in the environment. In order to prevent confusion and to have a common terminology for SLAM and SfM, landmarks are referred as 3d features and their observations as image features.
This work follows mostly the SfM terminology from [**lourakis2004a**, 96, 234, 117], where motion refers to camera poses and structure refers to reconstructed 3d

[24] https://en.wikipedia.org/wiki/Landmark

features forming a sparse representation of the captured environment. Furthermore, structureless refers to a reconstruction process, which does not include a structure generation based on 3d features.

2.1 Visual Simultaneous Localization and Mapping

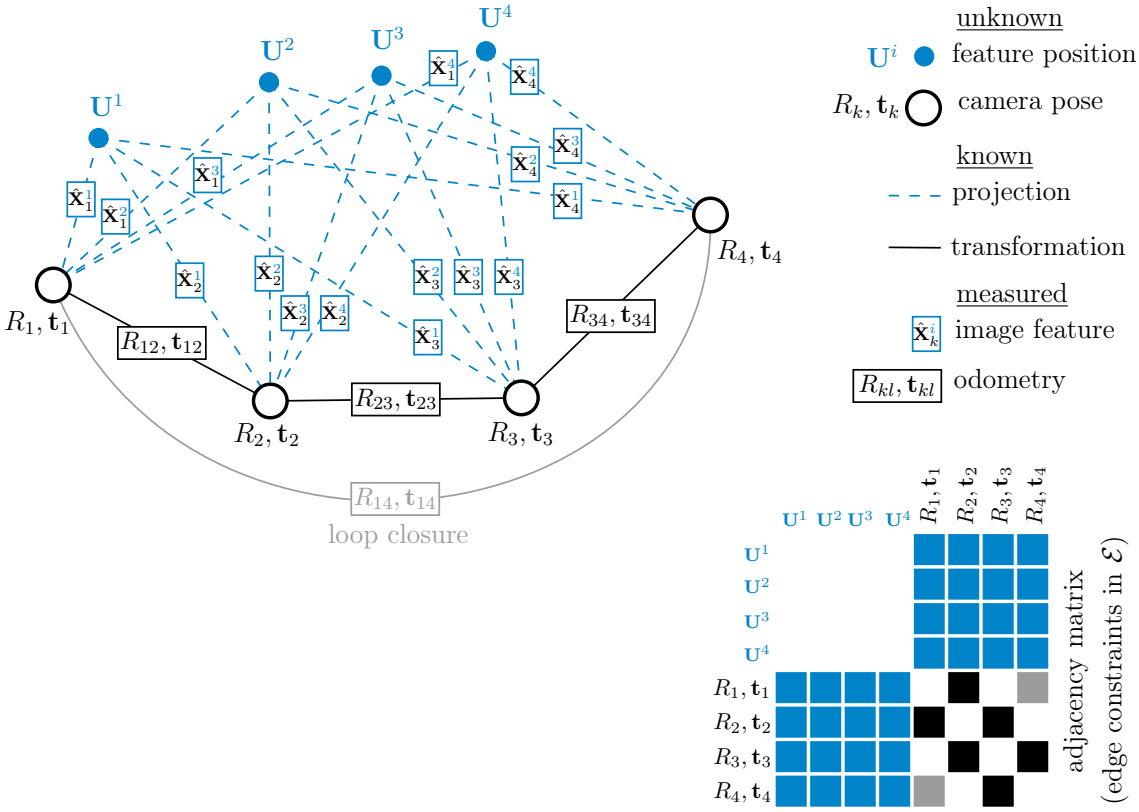

Fig. 2.2: Example case as *full* Visual SLAM problem illustrated as factor graph.

In most literature SLAM is illustrated by a robot equipped with a laser scanner, which directly measures the orientation and distance of 3d features in the surrounding environment. Consequently these 3d features can be recalculated based on their observations with respect to a known sensor pose.
In monocular Visual SLAM and SfM single cameras capture the projection of 3d features. As a consequence, depth information is lost and accordingly there is no metric distance relation between 3d features and cameras anymore. Thus monocular image-based 3d reconstruction is up-to-scale and requires reference points to adjust the results to real world dimensions. In contrast to laser scans, at least two camera poses are needed to reconstruct 3d features via triangulation from feature correspondences. There is a symbiotic relationship between image registration and triangulation in that images can only be registered to existing scene structure and scene structure can only be triangulated from registered images [288].

Visual SLAM's goal is to compute a consistent estimate of the camera's motion

(localization) while reconstructing the surrounding environment (mapping). A factor graph [149] is one way to express the Visual SLAM problem from Fig. 2.1 as graphical model, which is shown in Fig. 2.2. Factor graphs are bipartite, consisting of variable nodes (◯,●), factor nodes (□, ▭) and edges (—, - - -) [131, 49]. Variable nodes (◯,●) represent the unknown states of the cameras poses $[R_k, \mathbf{t}_k]$, $k = 1, \ldots, 4$ and feature locations $\mathbf{U}^i$, $i = 1, \ldots, 4$. Factor nodes (□, ▭) represent measurements such as observed features $\hat{\mathbf{X}}_k^i$ (projection of $\mathbf{U}^i$ in the k^{th} camera image) and relative transformations $[R_{kl}, \mathbf{t}_{kl}]$, between the k^{th} and l^{th} camera pose. The relative transformations are obtained from available VO (Visual Odometry), VIO (Visual Inertial Odometry) or scan matching if a laser-camera setup is used. Edges (—, - - -) link variable nodes and factor nodes by describing their relations, in this case as projection function and motion transformation function. Minimizing the sum of these constraints yields a maximum likelihood map and a corresponding set of camera poses [252]. In traditional factor graph representation, edges are illustrated as simple lines. For a better identification of different edge constraints, in this work they are drawn differently according their underlying function.
Referring to [132], the proposed Visual SLAM example can be expressed as least squares problem

$$\underset{R_{k,l},\mathbf{t}_{k,l},\mathbf{U}^i}{\operatorname{argmin}} \left\{ \sum_{(k,i)\in\mathcal{E}} \Big\| \underbrace{\hat{\mathbf{X}}_k^i - \pi\big(\mathbf{U}^i, [R_k, \mathbf{t}_k]\big)}_{\text{projective constraint}} \Big\|_{\Lambda_k^i}^2 + \sum_{(kl)\in\mathcal{E}} \Big\| \underbrace{[R_{kl}, \mathbf{t}_{k,k}] - [R_k^T R_l, R_k^T(\mathbf{t}_k - \mathbf{t}_l)]}_{\text{motion constraint}} \Big\|_{\Omega_{kl}}^2 \right\},$$

where $\pi(\cdot,\cdot)$ denotes a projection function and $\mathcal{E}$ covers all available edges in the graph corresponding to the adjacency matrix, whose structure is illustrated in Fig. 2.2. Λ_k^i and Ω_{kl} are measuring uncertainties relating to $\hat{\mathbf{X}}_k^i$ and $[R_{kl}, \mathbf{t}_{kl}]$, respectively. The presented equation is used as an example for demonstration purpose. Both parts, projective and motion constraints may vary depending on the used Visual SLAM implementation.

Fig. 2.3 shows the structure of the corresponding Jacobian. As can be seen, enlarging Visual SLAM by increasing the number of features $\mathbf{U}^i$ and camera poses $[R_k, \mathbf{t}_k]$, the problem becomes computationally expensive and even intractable to be solved in real time. In order to overcome this circumstance and to maintain real time capability, Visual SLAM algorithms split the problem into multiple data processing layers, incorporating VO, BA (Bundle Adjustment) and PGO (Pose Graph Optimization). Consequently they have become complex frameworks, each with its individual processing structure. Nevertheless, they all follow a common scheme, which roughly separates the data processing into three instances being processed simultaneously.

The **Tracking Instance** obtaines the camera pose from the current frame of

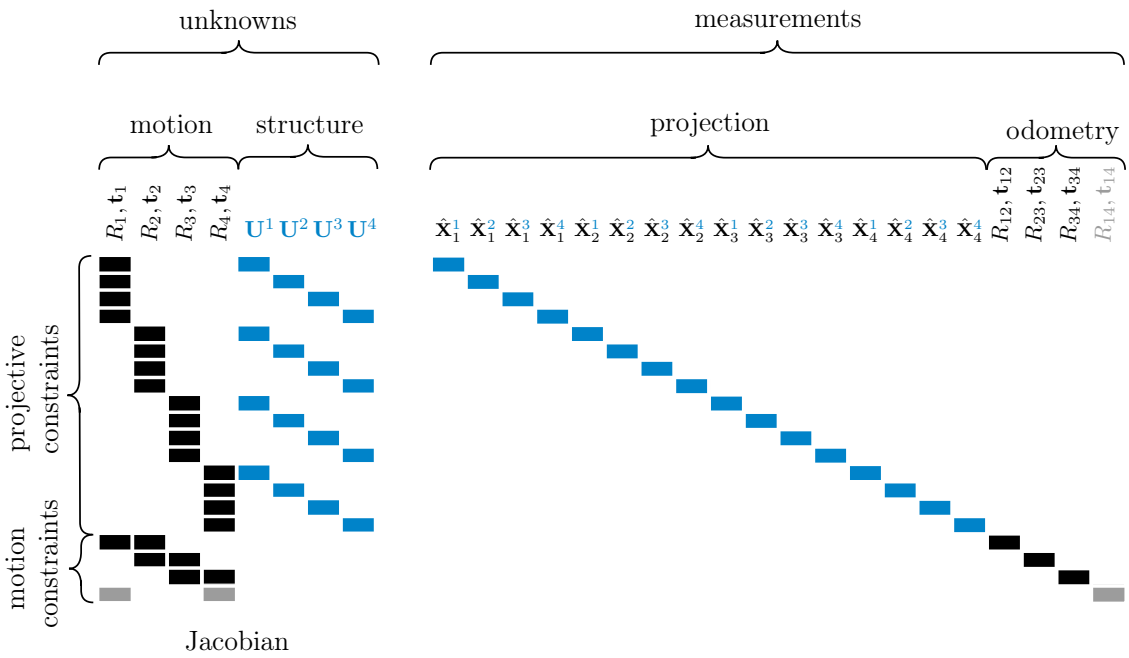

Fig. 2.3: Structure of the Jacobian related to *full* Visual SLAM.

an image sequence with respect to the active map using VO/VIO. It further separates the derived camera motion data into relevant and redundant information by selecting keyframes and thus discarding information from intermediate frames. This keyframing approach is used by various Visual SLAM implementations [160, 136, 190, 243, 27] in order to reduce data amount. As stated in [240], it is more profitable to increase the number of features than the number of frames in order to improve the accuracy. The tracking instance is also responsible for relocalization, which is required if tracking fails or a mapping session is resumed.

The **Mapping Instance** builds up a map using provided keyframes. In order to reduce drift induced by the tracking instance, it applies a local/windowed BA over a subset of keyframes, which - due to the small size - can be performed in real time. In case of loop closures the instance corrects the camera trajectory via PGO and updates the map according to the optimized camera poses. This allows a real time update of the camera trajectory over very long distances. Full BA jointly optimizes the camera trajectory and the map, which is done in a post processing step due to the large size of the optimization problem. Depending on the implementation, this instance also merges maps from previous sessions.

The **Data Assosciation Instance** builds up a visual vocabulary and a recognition database based on image featues in order to find suitable image matches for relocalization, loop closure detection and map fusion.

A detailed overview of state-of-the-art Visual SLAM algorithms is given in [211, 228, 27]. There is a wide range of Visual SLAM implementations, which deal with

various camera types in mono, stereo and multi-camera configurations, incorporate RGBD cameras and additional IMU sensors. Some of them are able to handle omnidirectional/fisheye images but poorly support full 360° × 180° ones. To the best of the author's knowledge, there is only one Visual SLAM algorithm using full 360° × 180° images (in equirectangular format) as input data [243].

2.1.1 Visual Odometry

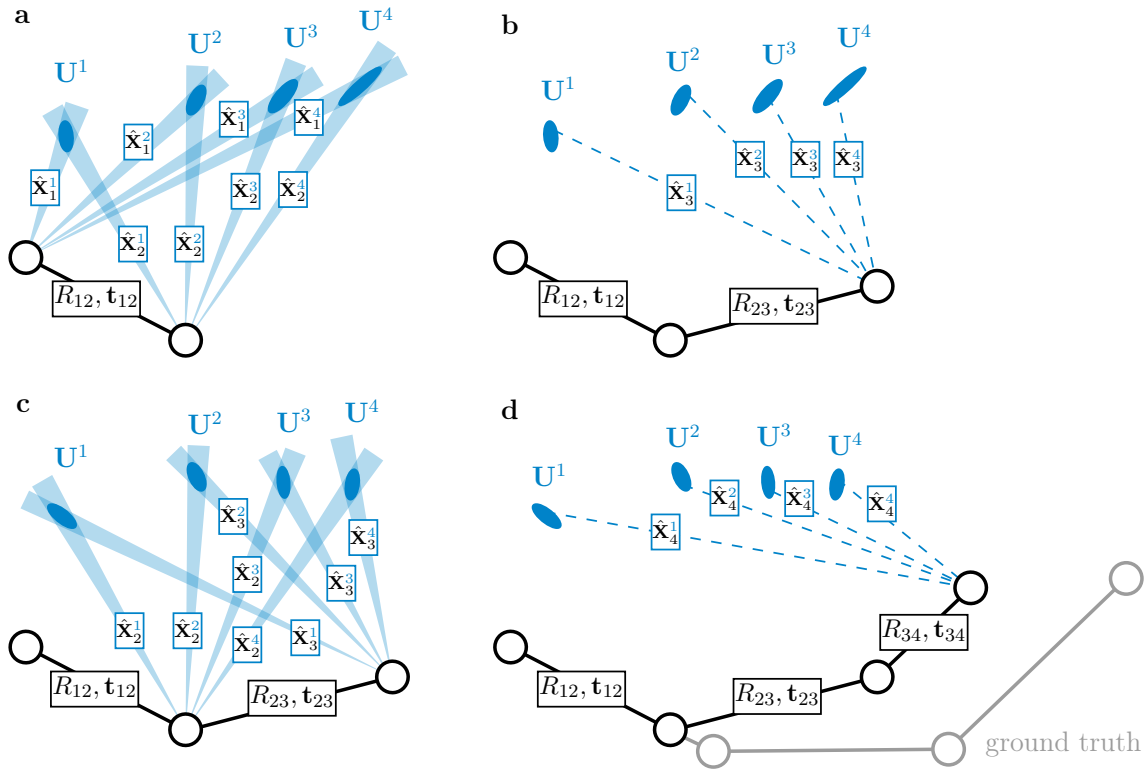

Fig. 2.4: Naïve VO approach as graphical representation. **a)** Feature corresponcences from 1st image and 2nd one are used to obtain an initial transformation $[R_{12}, \mathbf{t}_{12}]$. This transformation is used to triangulate $\mathbf{U}^i$ from feature correspondences $\hat{\mathbf{X}}_1^i \leftrightarrow \hat{\mathbf{X}}_2^i$. As illustrated, this triangulation is affected by observation uncertainties. **b)** PnP (Perspective-n-Point, Section 4.4.1,page 60) obtains the transformation $[R_{23}, \mathbf{t}_{23}]$ between 2nd image and 3rd one from 3d-2d correspondences $\mathbf{U}^i \leftrightarrow \hat{\mathbf{X}}_3^i$. **c)** Feature corresponcences from 2nd and 3rd image $\hat{\mathbf{X}}_2^i \leftrightarrow \hat{\mathbf{X}}_3^i$ are then triangulated to 3d space to obtain updated feature locations $\mathbf{U}^i$. **d)** Transformation $[R_{34}, \mathbf{t}_{34}]$ between 3rd image and 4th one is obtained from $\mathbf{U}^i \leftrightarrow \hat{\mathbf{X}}_4^i$ via PnP. This process is repeated for upcoming images. As can be seen from a comparison with the ground truth motion path, monocular VO is prone to transformation and scale drift. The presented example is a special case, where all cameras capture the entire feature set. Actually, this is not the case in many scenarios. Due to occlusion, there are individual feature matches between camera pairs. Triangulation is required to increase scene coverage by extending the set of triangulated features in order to register a new image to the existing scene. A more detailed description can be found in [218].

The name VO was first introduced in [192] and concentrates on the incremental estimation of camera poses [64] in order to obtain the camera's trajectory in real-time from an image sequence rather than to build up a map. It doesn't keep track of all the previous history and forgets environmental features once they get out of view [218].

This behaviour is prone to continuous estimation drift, which cannot be compensated even if the camera moves in the same area again. Monocular VO is also prone to scale drift and interesting concepts are made to tackle this problem for road and city environments [139, 80, 71, 289] using prior knowledge of the captured surrounding geometry.
Generally speaking, VO is a particular case of SfM [280] and can be used to provide initial estimates for BA and PGO. Consequently as mentioned in [218] VO is Visual SLAM before closing the loop. VO also provides much more accurate and reliable estimates over longer periods of time compared to wheel odometry [107], which is affected by wheel slippage usually caused by uneven terrain [280]. In order to reduce drift some VO algorithms perform BA on a subset of images to refine pose estimates of the trajectory [27, 218].

VO is mostly based on image features, which are matched across subsequent frames in order to obtain the camera motion. In monocular VO a 3d structure needs to be triangulated from two adjacent camera images, which is matched to 2d image features in the third camera image as illustrated in Fig. 2.4. Monocular VO is affected by a scale ambiguity problem [280], since transformations obtained from image features are up-to-scale only. In order to maintain a common scale, triangulation is required to align camera images via PnP (3d-2d pose estimation as describe in Section 4.4.1,page 60). Consequently this circumstance makes pose estimation dependent on triangulation uncertainties. In order to obtain a more robust camera trajectory with less drift, features are tracked over a subset of frames. Long tracks contain robust features which are used to align the current image. In [126] features are also separated into close features and distant ones. Distant features are less affected by small camera translations and hence are suitable to recover rotation, whereas close features are used to obtain translation. A more comprehensive description of image alignment techniques used in VO is given in [218, 280].

There are various implementations for different camera types in mono, stereo [140, 74, 137] and multi-camera setups [64, 227] as well as those incorporating additional sensors like IMU [107, 193] named VIO, wheel-odometry [178] or laser scanners [18] via sensor fusion. For more information about current implementations the reader is referred to [280, 27].

2.1.2 Pose Graph Optimization

A pose graph is a reduced form of a factor graph representing camera motion and corresponding transformation constraints only. PGO concentrates on solving global camera poses $[R_k, \mathbf{t}_k]$ from given relative motion constraints $[R_{kl}, \mathbf{t}_{kl}]$ with respect to

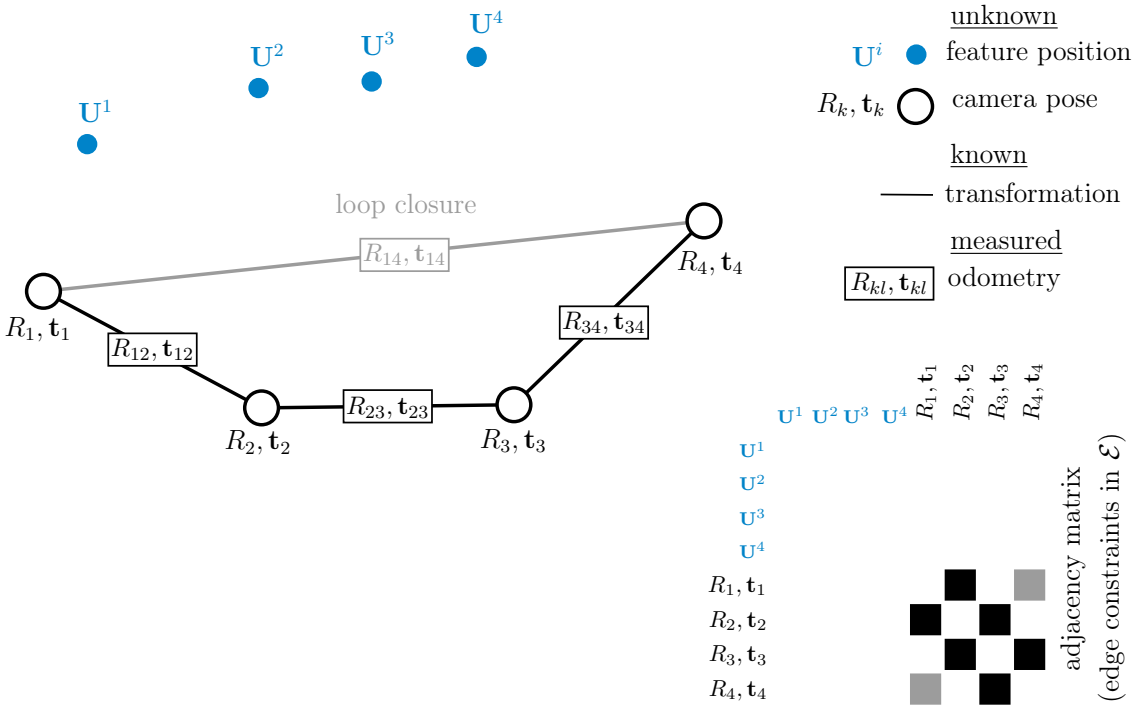

Fig. 2.5: Example case as PGO problem illustrated as factor graph.

corresponding uncertainties Ω_{kl} in least squares sense

$$\underset{R_k,\mathbf{t}_k}{\operatorname{argmin}} \sum_{(kl)\in\mathcal{E}} \left\| [R_{kl}, \mathbf{t}_{kl}] - [R_k^T R_l, R_k^T(\mathbf{t}_k - \mathbf{t}_l)] \right\|_{\Omega_{kl}}^2 .$$

It is used as back-end in Visual SLAM to optimize camera poses in case of detected loop closures to reduce motion drift. The task of constructing the actual pose graph is delegated to the front-end of the Visual SLAM implementation, which has access to the available sensor information [245]. Consequently PGO has no information about measured image features as Fig. 2.5 shows. The number of unknown optimization parameters thus depends only on the number of cameras but not on the number of image features as Fig. 2.6 indicates. The size of the problem slowly increases with the number of unkown camera poses, such that the problem is still solvable even in large-scale scenarios.

For a more comprehensive description the reader is referred to Chapter 8, page 139, which focuses in detail on pose graphs and PGO.

2.2 Structure from Motion

SfM (Structure from Motion) is a post-processing and accordingly an *offline* reconstruction technique used in photogrammetry to jointly obtain camera poses and 3d features from unordered image collections. It became popular for reconstructing large cityscapes and famous buildings from internet photos [234, 235, 4, 2, 106], for

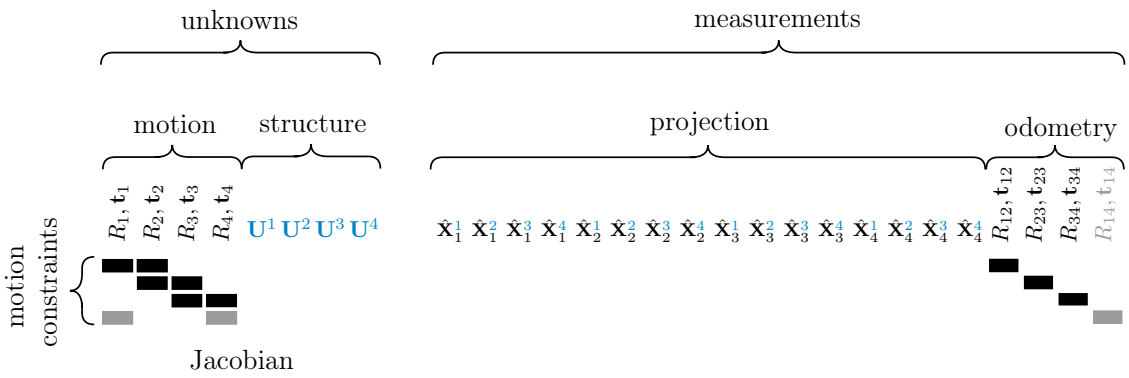

Fig. 2.6: Structure of the Jacobian related to PGO.

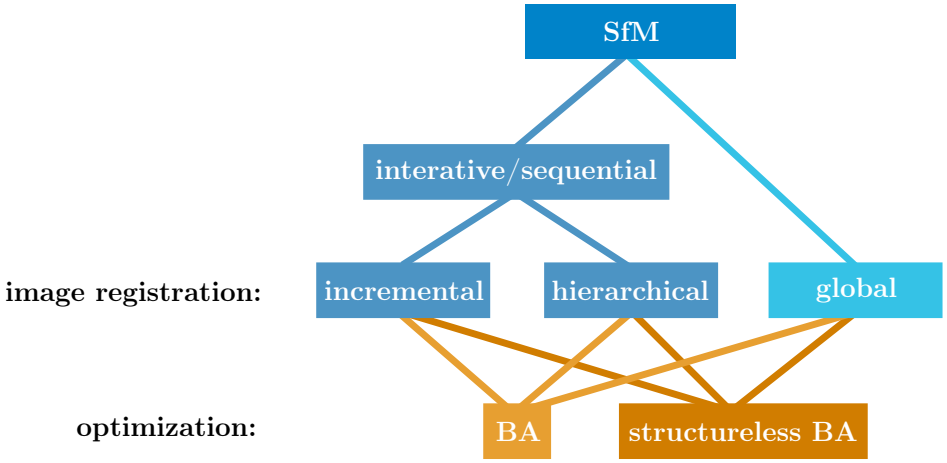

Fig. 2.7: SfM classification regarding image registration and optimization.

recovering camera motion from street view captures [141], for generating hyperlapse videos [143] or for image-based localization [120, 146, 247, 45, 111] in urban scenes, to name only a few applications. For more detailed information about the topic concerning SfM the reader is referred to [224, 194].

In recent years, various freeware impelementations have been released as closed source and open source:

- Bundler[25] [234, 235]
- Visual SfM[26] [276, 274]
- Colmap[27] [224, 225]
- OpenMVG[28] [188]
- Meshroom[29] [88]
- Apero/MicMac[30]
- Theia SfM[31] [251]
- Regard3D[32]

[25] https://www.cs.cornell.edu/~snavely/bundler/
[26] http://ccwu.me/vSfM/
[27] https://colmap.github.io
[28] https://github.com/openMVG/openMVG
[29] https://github.com/alicevision/meshroom
[30] https://micmac.ensg.eu/index.php/Accueil
[31] http://theia-SfM.org/
[32] https://www.regard3d.org

as well as commercial products:

- Agisoft Metashape[33]
- Reality Capture[35]
- SURE[aerial][37]
- 3DFlow Zephyr[34]
- PIX4Dmapper[36]
- PhotoModeler[38]

Commercial products usually provide an entire reconstruction pipeline, including SfM for camera registration, MVS (Multi-View Stereo) for dense point cloud reconstruction [70, 68, 69, 225], mesh generation and texturing. Most of them also allow to define extrinsic camera constraints in order to use multi-camera setups and to integrate laser scan data into the reconstruction process.
There is also a myriad of cloud-services for computer and smartphone devices, which are not considered here.

SLAM and accordingly SfM is a chicken-and-egg question [252]. A map is required to localize the cameras. Camera poses are needed to build a map. To tackle this problem different SfM approaches have been developed, which are classified in Fig. 2.7.
In general, SfM is a sequence of serveral techniques relating to feature detection and matching, two-view estimation, triangulation, pose estimation and non-linear optimization. It extracts features from given images and uses efficient matching strategies [236, 170, 102, 223] in order to select image pairs for feature matching. Derived feature matches between image pairs are geometrically verified and outliers are removed by determining the up-to-scale transformation using two-view geometry. However, due to the nature of noisy data ambiguous matches exist and unfortunatley there is no guarantee for complete outlier removal.

Based on two-view transformations a viewing/scene graph is built, which covers all derived image and feature relations. This graph is the basis for iterative approaches such as *hierarchical* [57, 76, 255] and *incremental* [234, 4, 275] reconstruction.

Hierarchical SfM approaches are less sensitive to initialization and drift error, however they are prone to insufficient feature matching such that reconstructed scenes are less detailed or incomplete [38].

Incremental SfM is the most popular strategy to recover camera poses and 3d feature information from unordered images. The following briefly describes a basic *incremental* SfM approach, which may vary depending on the implementation. Based on known two-view transformations and feature tracks (global feature correspon-

[33] https://www.agisoft.com
[34] https://www.3dflow.net/3df-zephyr-photogrammetry-software/
[35] https://www.capturingreality.com
[36] https://www.pix4d.com/product/pix4dmapper-photogrammetry-software
[37] https://www.nframes.com/products/sure-aerial/
[38] https://www.photomodeler.com

dences) *incremental* SfM selects an initial image pair and triangulates feature matches. Triangulation maps uncertainties from images to 3d space and hence is a topic of its own (Section 6.2, page 88), being extensively discussed in [101, 96, 238, 166, 134, 157, 277, 37]. The most important findings are the following. Under small parallax angles triangulation leads to large depth untertainties (as illustrated in Fig. 2.10, page 25) and those points are usually discarded [156] from the feature set. Triangulation from noisy or false feature observations must be also validated by cheirality (Section 6.4.1, page 97), which requires additional computational cost. Triangulation from multiple views increases accuracy, which is the reason why to build stable feature tracks in order to triangulate from all available views.

A new camera image is choosen from a *next best view selection* (e.g. the camera, that sees most triangulated points). This is a critical process, since a single bad decision leads to a cascade of camera misalignments and faulty triangulations [224]. The new camera is added to the scene based on 3d-2d point correspondences between triangulated features and image features of the new camera using a PnP algorithm, like the one presented in Section 4.4.1, page 60. This kind of algorithm obtains the camera pose with respect to the 3d features, some implementations are also able to recover camera intrinsics, if these are unknown. The newly registered image may also increase the scene coverage by extending the set of 3d features through triangulation. In the same way, all remaining cameras are added to the scene one by one.

Incremental SfM approaches register camera poses and 3d features in successive steps, which scales poorly as the image collection grows and can suffer from drift [44]. In order to reduce triangulation and pose estimation errors, an incremental/windowed BA (Section 2.2.1) is performed on the set of most-connected images after each image registration and feature triangulation. Since *incremental* SfM only affects the model locally, there is no need to perform global BA after each step. A global BA is performed after adding a certain amount of images in order to maintain appropriate run-times.

Incremental SfM adds new camera images in the same way as VO does, however it keeps track of matched features during the entire reconstruction process and does not forget about environmental features once they are out of view. *Incremental* SfM methods have been developed for constraint camera arrays [221, 141], for omnidirectional cameras [183], and for full $360° \times 180°$ cameras [196], which however are still poorly supported by available SfM applications.

Global SfM approaches [232, 9, 36, 122, 44, 187, 273, 46, 78] solve for all camera poses simultaneously using available up-to-scale two-view transformations from feature matching verification. These approaches obtain global camera rotations in a first step and perform translation averaging in a second step, which is a more difficult task, since each two-view transformation is up-to-scale and provides only the

direction of the translation. Based on results from *global* SfM approaches, BA can be performed for final optimization purpose, which is then less time consuming, since it only needs a few iterations. Up to now, there is no known implementation supporting full 360° × 180° cameras. Some implementations use multi-view constraints within camera triplets, however none of these approaches have proposed a global optimization that is solely based on multi-view constraints. However this idea paved the way for structureless BA [119].

2.2.1 Bundle Adjustment

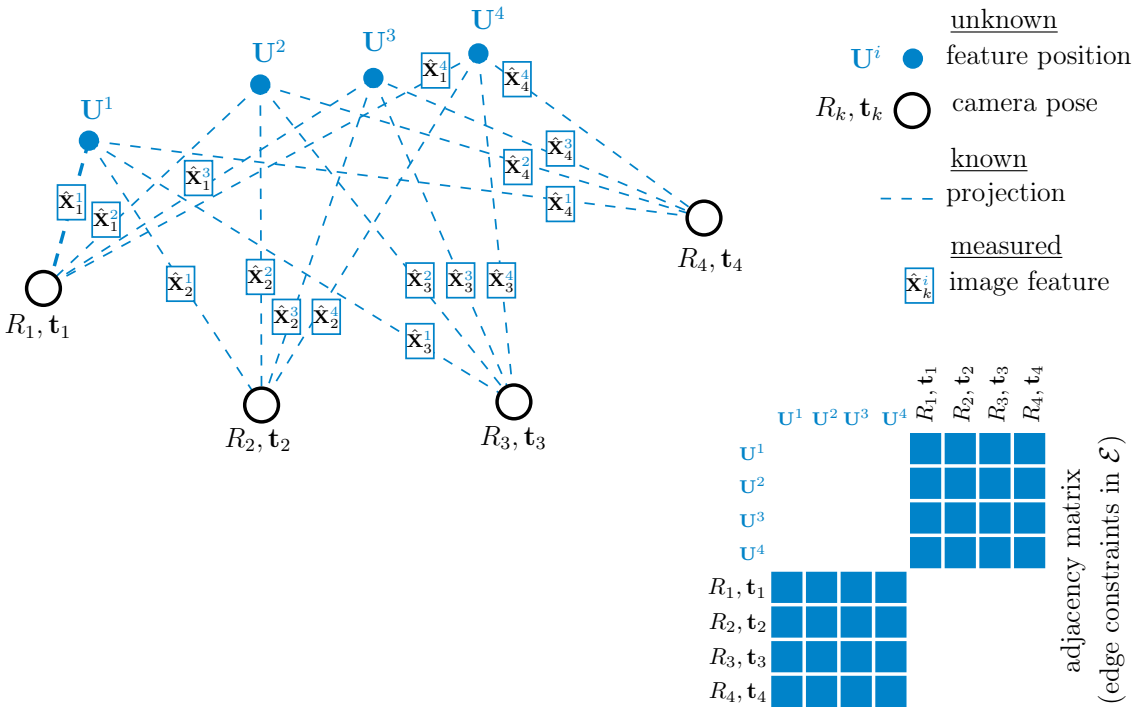

Fig. 2.8: Example case as BA problem illustrated as factor graph.

BA is a method to find the MAP estimate for both structure and motion by jointly optimizing 3d features and camera poses [259]. It represents the back-end of most SfM implementations and solves global as well as local registration tasks, as exemplary shown in Fig. 2.8. Popular standalone BA solvers like *SBA*[39] (Sparse BA) [172, 173], *MCBA*[40] (Multicore Bundle Adjustment) [276, 274] and *RPBA*[41] (Robust Parallel Bundle Adjustment) [180] are dedicated to SfM problems by exploiting the sparsity of the involved matrices in the optimization. Fig. 2.9 illustrates the mentioned sparsity at hand of the corresponding Jacobian. BA makes use of projective constraints and optimizes $[R_k, \mathbf{t}_k]$ and $\mathbf{U}^i$ by minimizing the sum of squared projection errors or squared

[39] http://users.ics.forth.gr/~lourakis/sba/
[40] http://grail.cs.washington.edu/projects/mcba/
[41] https://github.com/helmayer/RPBA

Mahalanobis distances (if measuring uncertainties Λ_k^i are known)

$$\underset{R_k,\mathbf{t}_k,\mathbf{U}^i}{\operatorname{argmin}} \sum_{(k,i)\in\mathcal{E}} \left\| \hat{\mathbf{x}}_k^i - \pi\left(\mathbf{U}^i, [R_k, \mathbf{t}_k]\right) \right\|_{\Lambda_k^i}^2$$

using robust LM or Dl [171] optimization. As can be seen, BA needs to optimize the entire set of 3d feature points $\mathbf{U}^i$ in order to refine all camera poses $[R_k, \mathbf{t}_k]$. In real world scenarios the number of 3d features is much larger compared to the number of cameras. As Fig. 2.9 explains, optimization becomes computationally expensive as the number of camera images and observed features increase. Thus the time complexity is quadratic related to number of images for most state-of-the-art algorithms [38].

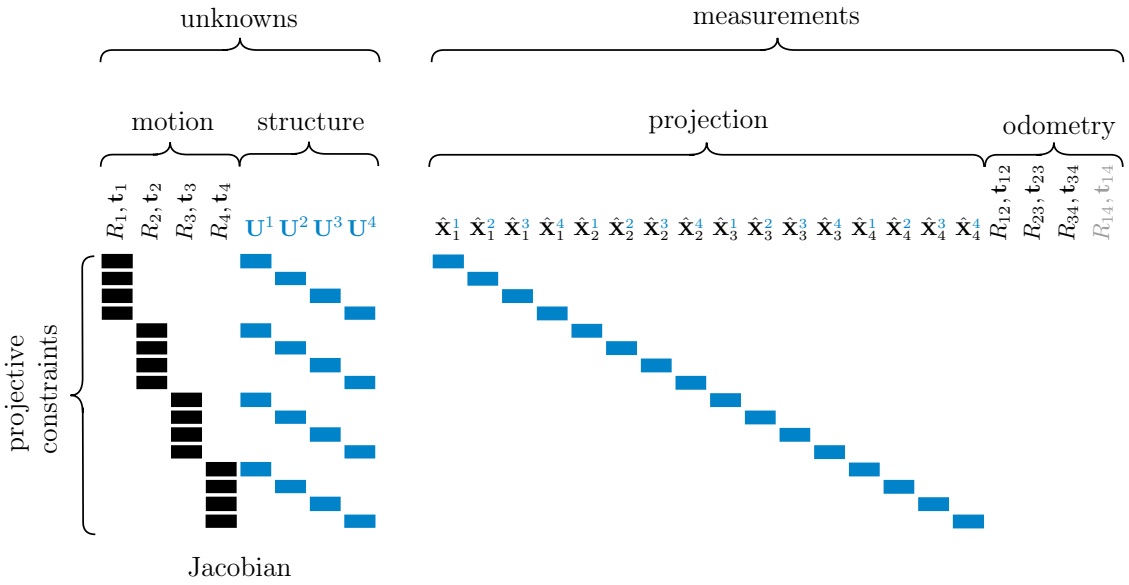

Fig. 2.9: Structure of the Jacobian related to BA optimization for solving an SfM problem.

In order to maintain computational cost managable and to keep time complexity linear to the number of images [38], some approaches have been made to solve large-scale SfM efficiently. They split the SfM problem into smaller, partially overlapping clusters/subsets, which are solved by parallel BA tasks [103, 104, 286, 16, 249, 39, 248, 291, 287, 55, 38]. However each sub-reconstruction relates to an individual scale. Consequently merging them is not straightforward and requires a combination of averaging and scaling techniques.

BA requires initial estimates, which should be sufficiently close to the optimal solution otherwise it can fall out of the convergence basin [207]. It does not make use of relative transformations $[R_{kl}, \mathbf{t}_{kl}]$ between camera poses, which are obtained during feature matching verification. These pieces of information are rejected such that camera poses are not directly connected by transformation constraints during optimization. Cameras are only linked via triangulated 3d feature points and their corresponding projections. What seems to be an disadvantage can be justified with

the fact, that projective constraints maintain scale and also allow BA to solve for unknown camera intrinsics, which is a tremendous advantage when working with image collections from internet or from unknown image sensors.

Projective constraints induce a reduction of dimension leading to the problem, that optimization of observed features is performed in 3d space with respect to an error distance in 2d space. In case of image noise this leads to triangulation uncertainties along the LoS (Line of Sight) as illustrated in Fig. 2.10. Points being observed under small parallax angle exhibit very large depth uncertainty, which is caused by small translatory camera motion or by large distances between 3d features and camera centers [218]. Thus large reconstruction uncertainties in 3d do not necessarily correspond to noticeable projection errors in the image and always depend on the number of camera views and their distribution.

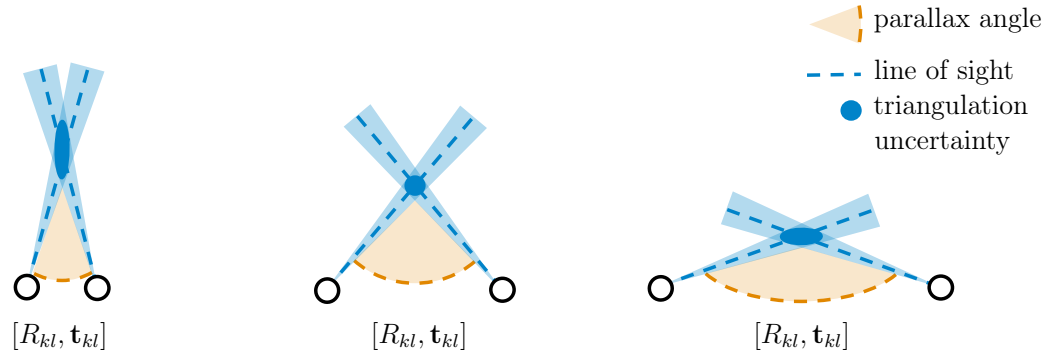

Fig. 2.10: Triangulation uncertainty over increasing parallax angle

To avoid the above problems, novel algorithms prevent triangulation and optimization under small parallax but also under large parallax, perform inverse depth weighting to reduce the influence of distant points, build feature tracks for robust triangulation from multiple views, validate cheirality (Section 6.4.1,page 97) to identify false feature matches (check if 3d features are in front of the camera) or split the reconstruction problem into smaller parts to maintain computational cost and runtime managable. Freely spoken, all these interventions do not change the fact, that a vast amount of 3d features needs to be reconstructed in order to obtain a handful camera poses. 3d features form a sparse representation of the observed structure, which can be used for real-time reconstruction monitoring purposes but they do not play an important role for further dense reconstruction using MVS. Based on known camera poses, image feature matchings and their corresponding triangulations can be recovered, if required.

2.2.2 Structureless Bundle Adjustment

Structureless BA techniques avoid 3d feature reconstruction in order to reduce the number of variables involved in the optimization. As illustrated in Fig. 2.11 they eliminate 3d features by replacing projective constraints with multi-view ones in order to

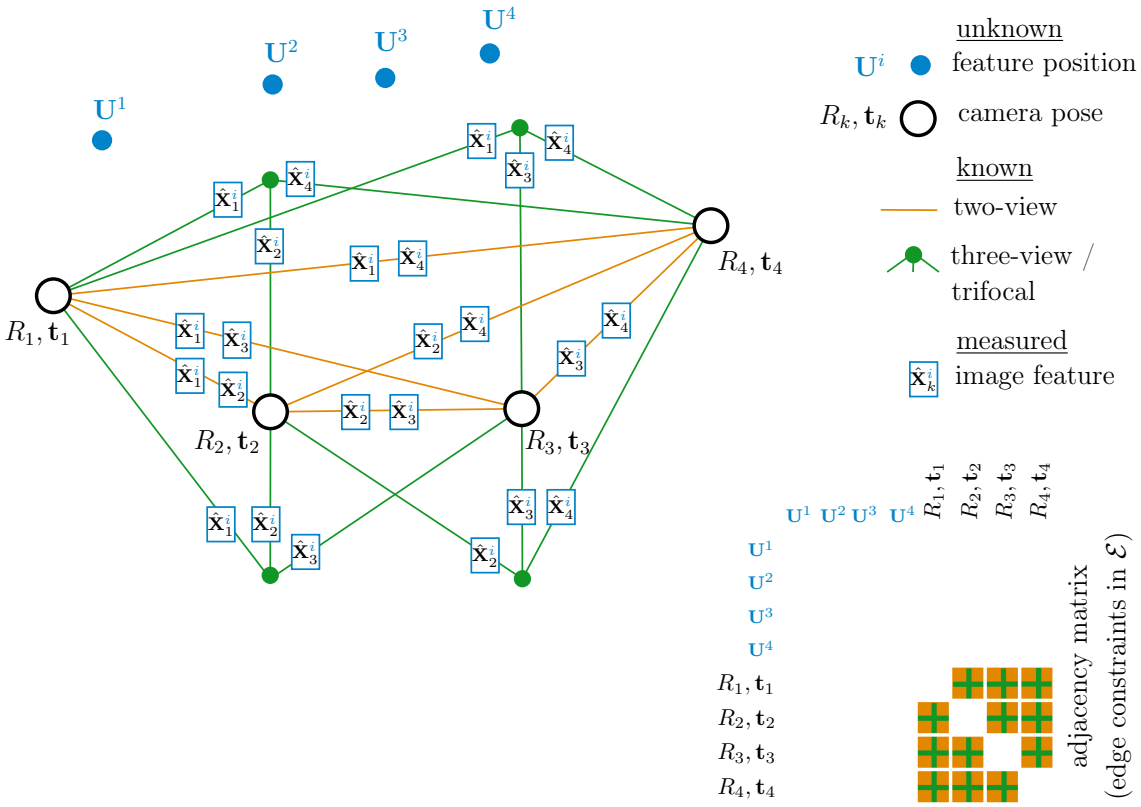

Fig. 2.11: Example case as structureless BA problem illustrated as factor graph.

obtain an MAP estimate of the camera poses. If required, structure is recovered by triangulating 3d features from optimized camera poses afterwards. There are different approaches using epipolar constraints [79, 207, 208], epipolar point transfer [250], combined epipolar and trifocal constraints [237] and combined epipolar and three-view constraints [115, 118, 117, 119]. In contrast to using only epipolar (two-view) constraints or epipolar point transfer, the use of three-view constraints as well as trifocal ones maintains a consistent scale even in collinear camera configurations (straight-line camera motion) [118, 119].

The combination of two-view and three-view/trifocal constraints replaces 3d features and thus reduces the optimization problem to the number of unknown motion parameters. As a consequence the number of constraints in structureless BA is usually much larger compared to the ones in BA [119], since each 3d feature may be observed by more than three camera views and hence is represented by a set of constraints between different cameras observing the same 3d feature. This circumstance can be seen by comparing the Jacobians from Fig. 2.9 and Fig. 2.12.

Structureless BA can be expressed as least squares optimization

$$\underset{R_{k,l,m},\mathbf{t}_{k,l,m}}{\operatorname{argmin}} \left\{ \sum_{(i,k,l)\in\mathcal{E}} \left\| \underbrace{g\big(\hat{\mathbf{X}}_k^i, \hat{\mathbf{X}}_l^i, [R_k,\mathbf{t}_k], [R_l,\mathbf{t}_l]\big)}_{\text{two-view constraint}} \right\|^2 + \sum_{(i,l,m)\in\mathcal{E}} \left\| \underbrace{g\big(\hat{\mathbf{X}}_l^i, \hat{\mathbf{X}}_m^i, [R_l,\mathbf{t}_l], [R_m,\mathbf{t}_m]\big)}_{\text{two-view constraint}} \right\|^2 + \sum_{(i,k,l,m)\in\mathcal{E}} \left\| \underbrace{h\big(\hat{\mathbf{X}}_k^i, \hat{\mathbf{X}}_l^i, \hat{\mathbf{X}}_m^i, [R_k,\mathbf{t}_k], [R_l,\mathbf{t}_l], [R_m,\mathbf{t}_m]\big)}_{\text{three-view/trifocal constraint}} \right\|^2 \right\},$$

where $g(\cdot,\cdot,\cdot,\cdot)$ denotes a two-view constraint between k^{th} and l^{th} camera as well as between l^{th} and m^{th} camera, and $h(\cdot,\cdot,\cdot,\cdot,\cdot,\cdot)$ denotes a three-view/trifocal constraint between k^{th}, l^{th} and m^{th} camera.

Besides an implementation for GTSAM named *iLBA*[42] (incremental light Bundle Adjustment) [118, 117, 119], there is no known SfM implementation/application incorporating structureless BA.

2.3 Corresponding Issues

Visual SLAM is limited to image sequences and novel developments have been made to reconstruct large-scale scenarios in real-time. They exploit the nature of sequential overlapping images, which can be registered one by one by using a combination of key-framing, local and global optimization techniques. Most visual SLAM implementations incorporate BA for image registration to reduce scale and local motion drift from VO, whereas PGO performs camera trajectory optimization over longer distances in case of loop closures to reduce global motion drift. However this strategy cannot be fully applied to unordered image collections. *Incremental* SfM is mostly used in that case and represents a robust registration strategy. It adds new cameras to an existing scene by using a combination of triangulation, pose estimation and optimization. It incorporates BA for local and global refinement in order to reduce drift. Thus the final result depends on each individual camera registration and a single decision can ruin the entire reconstruction results.

BA quickly becomes computationally expensive as more images are added since it solves for camera poses and 3d features. On the contrary structureless BA solves for camera poses only and replaces 3d features with multiview-constraints, which leads to an increased number of additional constraints. Both optimization techniques require initial estimates from image registration techniques, which should be in a sufficient range to the final solution. PGO initializes and optimizes global camera poses using two-view transformations (motion) between them. As can be seen in Fig. 2.13 PGO

[42] https://vindelman.net.technion.ac.il/software/

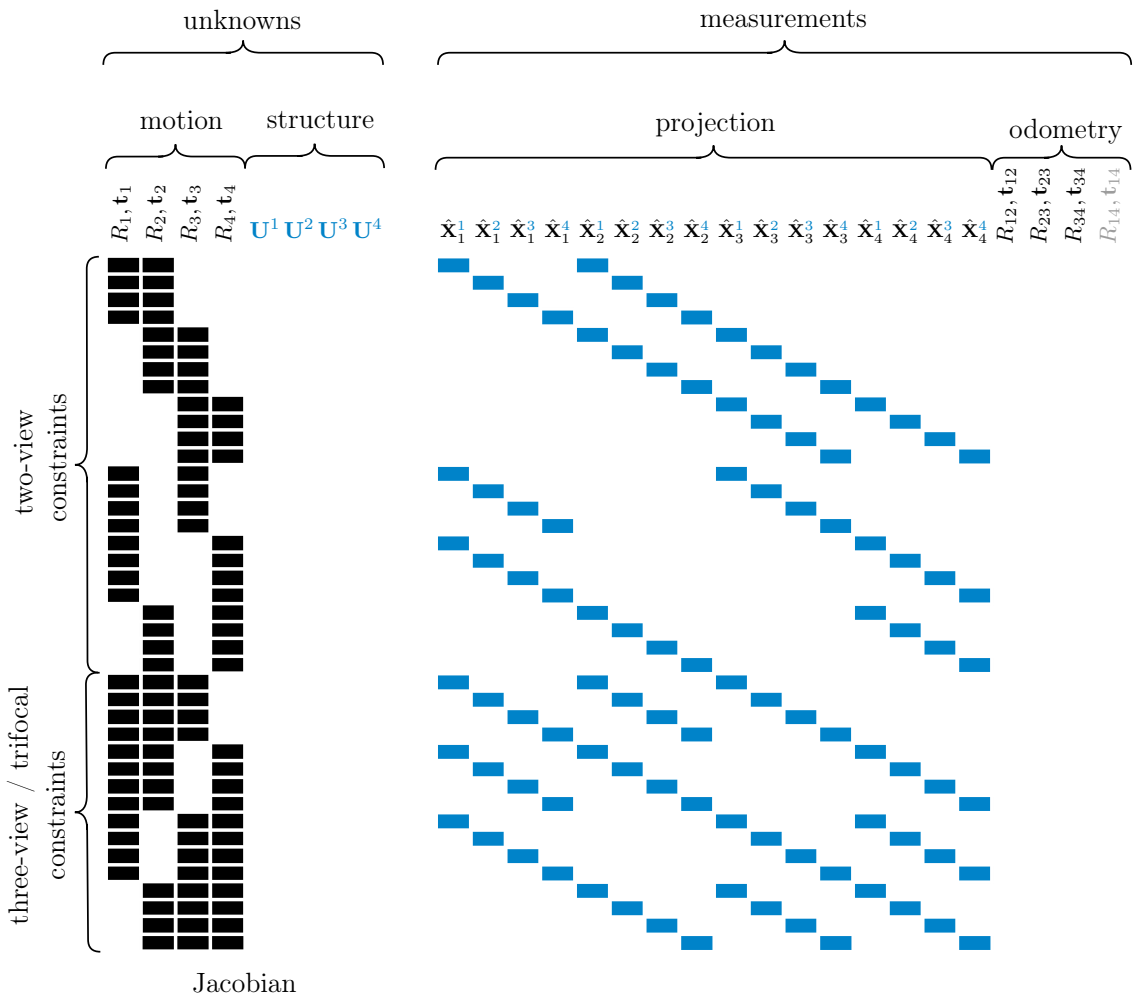

Fig. 2.12: Structure of the Jacobian related to structureless BA optimization for solving an SfM problem.

solves for the same number of unkown motion parameters as structureless BA does but is subjected to less constraints compared to BA. Hence is keeps the problem small and managable in both dimensions, unknowns and constraints, and thus is computationally more efficient. However PGO requires scaled two-view transformations and cannot be included into an SfM pipeline straightforwardly.

BA and structureless BA have a more complex problem structure as compared to PGO. During serveral tests in underground environments, *incremental* SfM ran into problems when loop closures over longer distances occured, as depicted in Fig. 2.14. *Incremental* SfM starts with an initial image pair and adds new cameras to one of either ends. Ideally, both ends should coincide in case of a loop closure, which due to drift in reconstruction never happenes. *Incremental* SfM uses local BA to reduce drift after each single image registration and accompanying feature triangulation leading to locally optimized reconstructions. Hence loop closing constraints produce comparativeley larger residuals. In order to align both ends of the loop closure BA tries to optimize the entire reconstruction. Due to complex problem structure, where

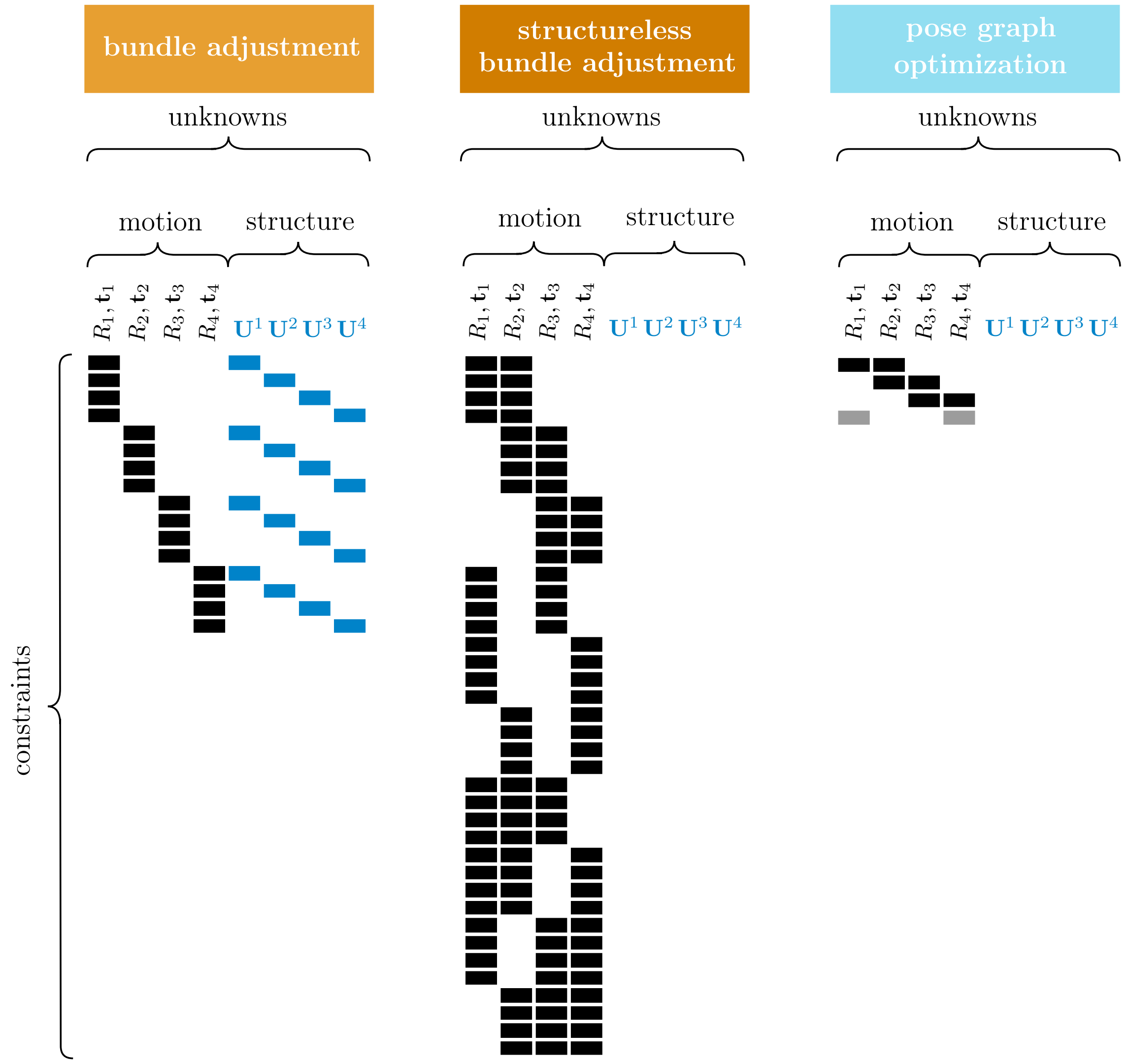

Fig. 2.13: Comparision of Jacobians in BA, structureless BA and PGO to solve for unknown motion (camera poses). The structure and dimensions of the Jacobian illustrate the size and complexity of the mathematical problem to be solved.

cameras are linked via 3d features, this is related to higher computational effort. In order to update camera poses, BA also needs to update corresponding 3d features. Hence larger parts of the reconstruction are influenced, which are already locally optimized. During optimization the sum of squared distances can increase under certain circumstances. BA is usually applied in combination with residual filtering, which removes constraints relating to large projection errors after a certain amount of iterations. This can also happen to loop closure constraints if BA is unable to improve reconstruction within a certain optimization regime. Generally speaking, in some cases it is computationally beneficial for the algorithm to remove small parts, namely cameras and 3d features related to large error constraints than trying to adjust the remaining parts of the reconstruction.

Fig. 2.15 illustrates the same scenario but reconstructed using a VO and PGO. As can be clearly seen, camera poses are directly linked, which leads to a simple

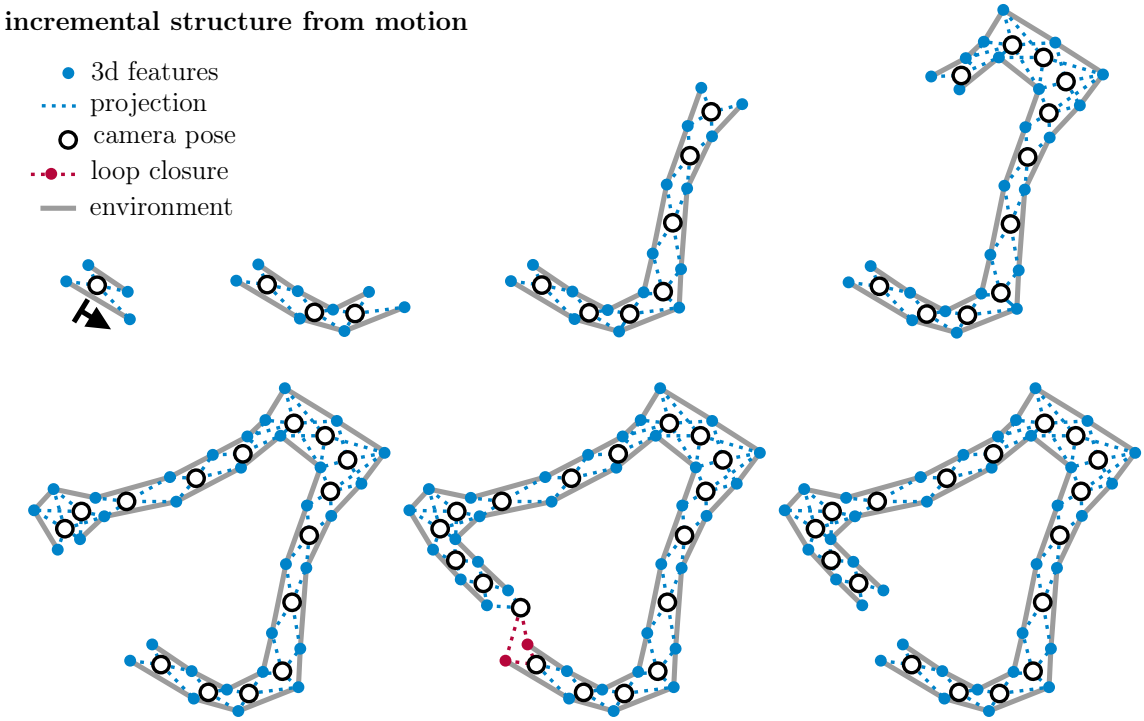

Fig. 2.14: *Incremental* SfM reconstructing camera poses and 3d features in a narrow underground environment. After each image registration and accompanying feature triangulation BA is performed to reduce drift. However loop closures after long distances might lead to the circumstance, that loop closing features as well as cameras are rejected through a filtering process, which leads to an incomplete and drift affected reconstruction.

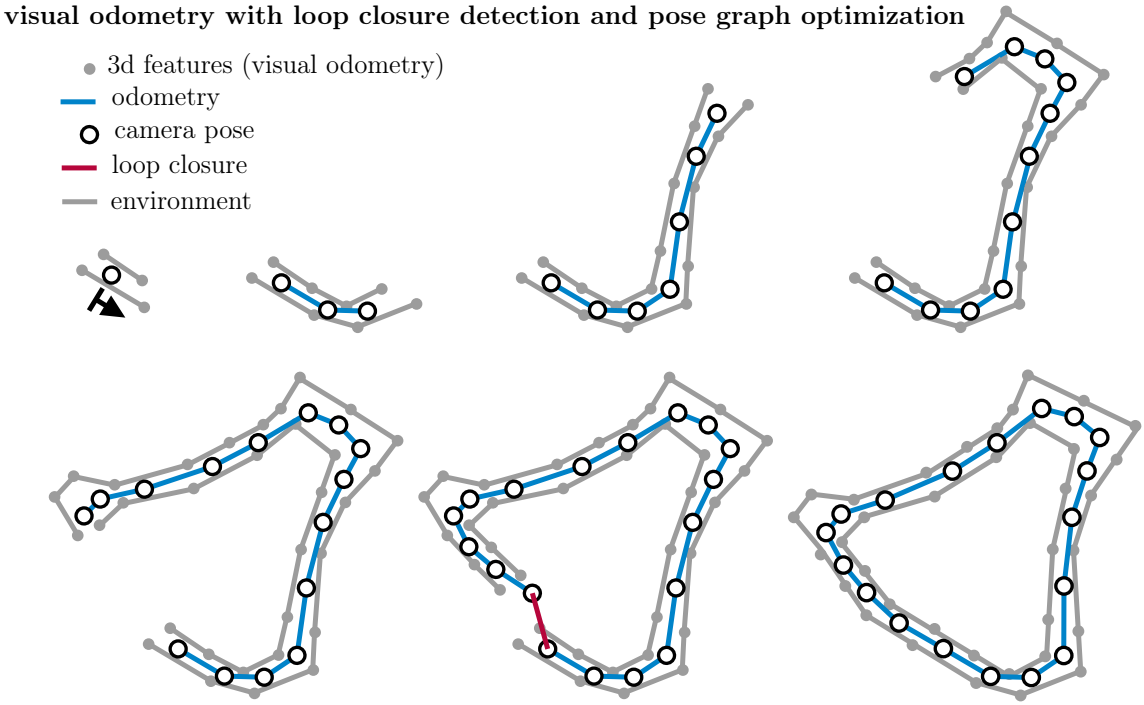

Fig. 2.15: VO in combination with loop closure detection and PGO reconstructing a camera trajectory in a narrow underground environment. Each time a loop closure is detected PGO compensates motion drift in the camera trajectory.

problem structure that enables compensating motion drift in case of a loop closure via PGO. Structure (3d features) can be recalculated from optimized camera poses, if required.

Considering all mentioned aspects, a camera motion estimation pipeline based on PGO is on the one hand computationally efficient and on the other hand more flexible in case of loop closures.

2.4 Proposed Reconstruction Pipeline

Based on presented findings from previous sections a new camera motion estimation pipeline is proposed that avoids structure (3d feature) estimation and solves for motion (camera poses) from unordered image collections and hence is called SCME (Structureless Camera Motion Estimation). Similar to SfM, SCME combines different image data processing algorithms like feature detection and matching as well as transformation estimation using two-view and three-view/trifocal geometry. The pipeline is inspired by *global* SfM (Section 2.2, page 19) approaches, structureless BA (Section 2.2.2, page 25) and PGO (Section 2.1.2, page 18).

Structureless BA - as it is presented in Figs. 2.11 and 2.12, pages 26 and 28 - uses two-view constraints to maintain rotation and translation direction within a two-view transformation in combination with three-view/trifocal constraints to maintain translation scale between two-view transformations. At each iteration step structureless BA uses point correspondences and current pose estimates in order to obtain residuals from two-view and three-view/trifocal constraints. Each residual represents an error distance between current camera configuration and an ideal camera configuration. Since this process is repeated for all iteration steps it leads to a significant computational cost.

SCME consideres the structureless BA problem from a different perspective. It uses up-to-scale two-view estimates from image matching and its geometric validation (Chapter 6), which are the best obtainable pose transformations between image pairs [218] and hence they are used as reference for optimization similiar to *global* SfM.
Three-view/trifocal constraints maintain the translation scale between two-view transformations. Consequently these constraints obtain a ratio factor between up-to-scale two-view transformations. Based on these ratio factors a global two-view translation scaling is be performed to obtain scaled two-view transformations, to be used a input data for efficient PGO (Chapter 8).

SCME introduces an intermediate step to obtain scaled two-view transformations from feature correspondences, neither using an incremental scaling approach nor triangulating features. As a consequence camera poses are not directly obtained from feature correspondences but from two-view transformations. Thus results represent

inital estimates, which may be refined via BA or structureless BA. SCME presents a fast and reliable pipeline to provide global camera pose estimates, without the need to reconstruct any single point in 3d. The proposed pipeline is especially designed for full $360° \times 180°$ cameras but also incorporates any directional camera, as long as its calibration can be converted into a P2S-map. Chapter 9 comprehensively explains the proposed SCME pipeline.

3 Cameras and Pixel-to-Sphere Mappings with P2S-Maps

Brief Chapter Overview

This chapter covers intrinsic camera modelling and introduces a novel strategy to store the obtained camera metric. Section 3.1 gives a brief summary about camera types, their classification and technical design. Section 3.2 concentrates on camera models and describes a geometric camera model in Section 3.2.1 and an analytic camera model in Section 3.2.2. Both are suitable for a wide range of central camera types and are also able to map pixels onto unit sphere, which paves the way to use a general spherical camera model, which is described in Section 3.2.3. Instead of saving the derived camera metric as parameters of a mapping function, Section 3.3 explains the idea of using a lookup mapping, which stores each pixel's corresponding position on the unit sphere. Consequently by using a lookup the camera geometry becomes independent of the underlying camera calibration model. Section 3.3.1 introduces a color-coding scheme to save lookup mappings as image file, which is called P2S-map and represents an equation-free calibration format. Section 3.3.2 explains how to obtain lookup values for intermediate pixels from a P2S-map using a linear interpolation technique. As a last point Section 3.3.3 illustrates the conversion of depth map data from RGBD-sensors in order to be used in conjunction with a P2S-map for back-projection purposes and scaling of two-view geometries (Section 6.7, page 113).

3.1 Types

This section shortly introduces a classification of different camera types as shown in Fig. 3.1. Cameras are classified into central and non-central according the way how light passes through the optical system.

Central cameras (or single effective viewpoint cameras) have a unique point, where all optical rays intersect. Each ray passes through this viewpoint in one particular direction. If a camera is calibrated, each pixel's direction can be recovered by unprojection the image onto the sphere centered at the single viewpoint. The single viewpoint constraint is the base for the well-known epipolar geometry (Section 6.3, page 94) that holds for any central camera system.

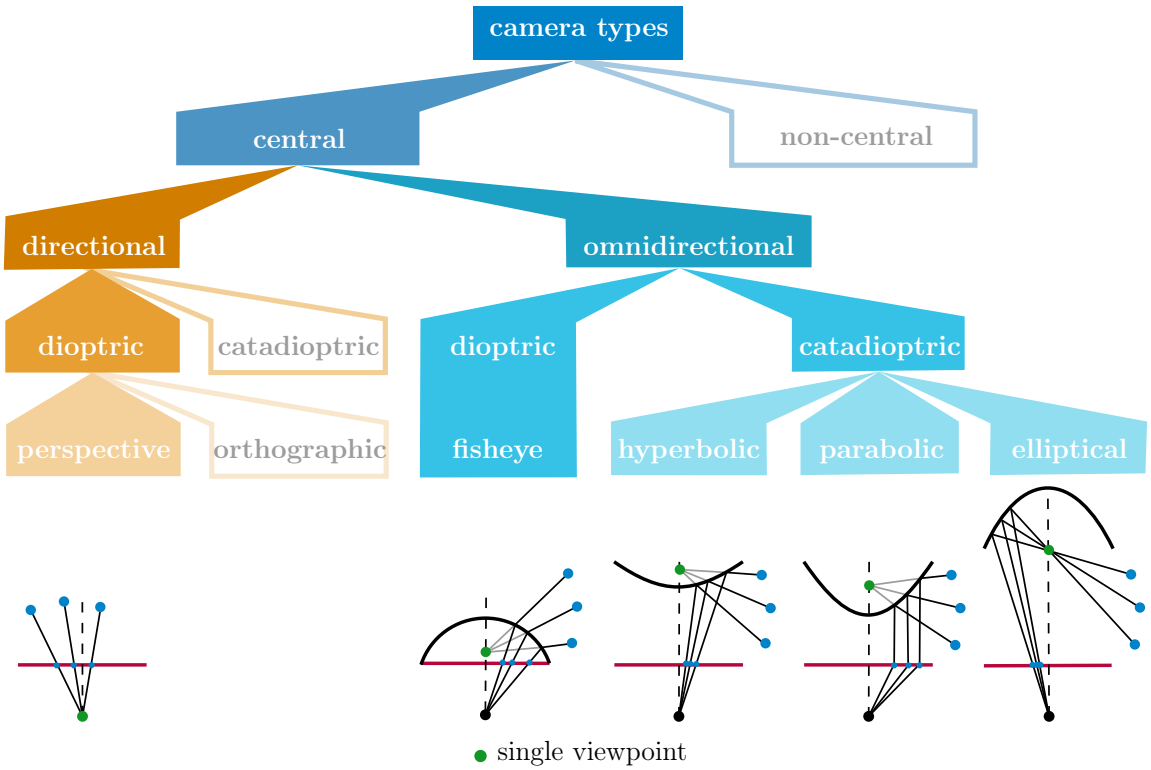

Fig. 3.1: Classification of camera types

Non-central cameras (or non-single effective viewpoint cameras), do not have a unique intersection point and project scenes into images along a general set of lines. This type of camera is not considered in this work.

Central directional cameras are the most common type in video and image applications and produce a limited FoV. They are further classified into **dioptric** (light refraction using lenses) and **catadioptric**, which is a combination of lenses (**dioptric**) and mirrors (**catoptric:** light reflection using curved mirrors).

The majority of cameras are called **central directional dioptric cameras** and produce a **perspective** view. Telecentric lenses produce an **orthographic** view, however they are very expensive and play an overall minor role since the are only used in specific research fields like fluid dynamics [203, 202].

There are also **central directional catadioptric cameras** such as many kinds of telescopes and mirror-based telephoto lenses. They are not used in robotics for mapping or navigation purposes.

Central omnidirectional cameras (**omni:** meaning all) have a 360° FoV in the horizontal plane. They combine a single projection center with a wide FoV. They became famous in different areas like surveillance for tracking and especially in robotics for visual navigation, localization and mapping. Similar to **central**

directional cameras, they are also further classified into **dioptric** and **catadioptric**.

Central omnidirectional dioptric cameras use a combination of shaped lenses. Fisheye lenses typically represent this camera type, which can cover a vertical FoV up to 250°[43]. It should be noted, not all fisheye lenses are central. As mentioned in [279] fisheye lenses do not necessarily have a single projection center but a locus of projection centers. In many computer vision and robotics literature this locus is assumed to be very small and approximated as single viewpoint. During the last years design and manufacturing of fisheye lenses have been improved. Modern fisheye lenses are nowadays classified as central.

Central Omnidirectional Catadioptric Cameras are a combination of a perspective camera with a hyperbolic or eliptical mirror or an orthographic camera with a parabolic mirror [10]. They were widely used in the robotics community, however manufacturing of these lenses is difficult, due to the fact that lense and mirror need to be precisely aligned.

Convention

Since the work focuses on **central camera** systems the term **central** will not be mentioned always. The terms **dioptric** and **catadioptric** will relate to **omnidirectional cameras**. Furthermore the indication **directional cameras** will always mean **dioptric perspective cameras**.

Omnidirectional cameras covering an entire sphere (360° × 180°) are referred as **full omnidirectional cameras**.

3.2 Models

This section focuses on two popular camera models, which allow to calibrate a wide range of central dioptric and catadioptric cameras. A third model assumes the camera as sphere, which is used as general representation of cameras in this work.

3.2.1 Unified Camera Model

The Unified Camera Model (UCM) - as it is presented here - was introduced by Mei [181] and is a modification of [75, 11]. It covers a wide range of directional and omnidirecional camera types such as catadioptric cameras with parabolic, hyperbolic, ellipsoidal and planar mirros as well as dioptric cameras with fisheye lenses [279, 43, 264].

[43] https://products.entaniya.co.jp/en/products/hal-250200/fisheyehal250/

Projection

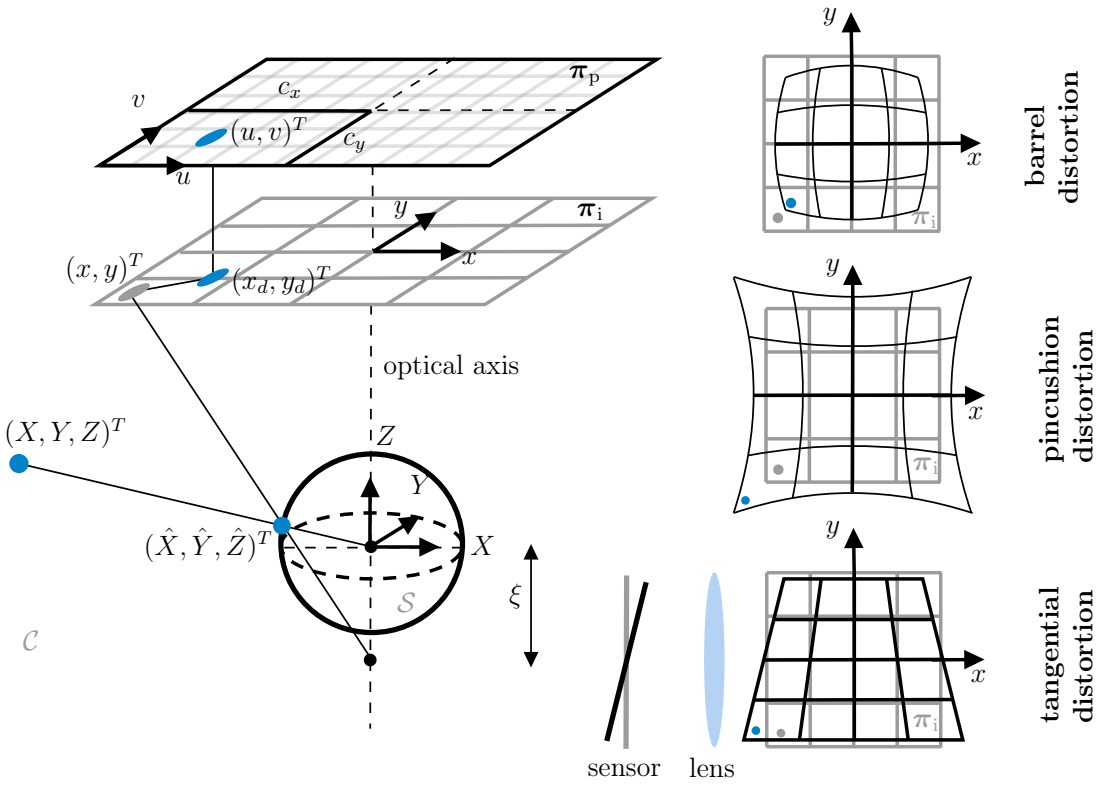

Fig. 3.2: Forward projection of the UCM with encountered main distortion types (barrel, pincushion, tangential).

A point $\mathbf{U} = (U,V,W)^T$ is transferred from world frame $\mathcal{W}$ to $\mathbf{X} = (X,Y,Z)^T$ in camera frame $\mathcal{C}$

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = R^T \begin{pmatrix} U \\ V \\ W \end{pmatrix} - R^T\mathbf{t} \tag{3.1}$$

using the extrinsic camera paramaters $[R,\mathbf{t}]$ (or simply extrinsics), which denote the camera pose.

As shown in Fig. 3.2, the intrinsic camera parameters (or simply intrinsics) describe the transformation of $(X,Y,Z)^T$ from camera frame $\mathcal{C}$ to $(u,v)^T$ on pixel plane $\boldsymbol{\pi}_\mathrm{p}$. The UCM is a sequence of successive projections. The first one projects $(X,Y,Z)^T$ from camera frame $\mathcal{C}$ onto the camera's unit sphere $\mathcal{S}$

$$\hat{\mathbf{X}} = \begin{pmatrix} \hat{X} \\ \hat{Y} \\ \hat{Z} \end{pmatrix} = \frac{(X,Y,Z)^T}{\|(X,Y,Z)^T\|}\,. \tag{3.2}$$

Then, the second one projects $(\hat{X},\hat{Y},\hat{Z})^T$ from unit sphere $\mathcal{S}$ to $(u,v)^T$ on the (normalized) image plane $\boldsymbol{\pi}_\mathrm{i}$ by shifting the center of projection from $(0,0,0)^T$ to $(0,0,-\xi)^T$

along Z-axis

$$\begin{pmatrix} x \\ y \end{pmatrix} = \frac{1}{\hat{Z} + \xi} \begin{pmatrix} \hat{X} \\ \hat{Y} \end{pmatrix} . \tag{3.3}$$

Sidenote: In some of the rich calibration literature the normalized image plane is simply referred as image plane ([216, 135]). Furthermore, there exist different denotations for pixel plane such as sensor plane [216, 135] or image plane [19]. In order to prevent confusion and to establish a uniform denotation, $\boldsymbol{\pi}_{\mathrm{i}}$ denotes the image plane and $\boldsymbol{\pi}_{\mathrm{p}}$ denotes the pixel plane since it describes point coordinates in pixel dimension and represents the projection plane of sensors and map projections (Section 5.2, page 78).

ξ represents the parameter that models the distortion. Depending on that value the projection transits from gnomonic ($\xi = 0$) to stereographic ($\xi > 0$) on the one hand and on the orther hand the projection enlarges the FoV ($-1 < \xi < 0$) by shifting the projection center towards the image plane $\boldsymbol{\pi}_{\mathrm{i}}$. This is possible for cameras with a smaller FoV ($<< 180°$). The smaller the FoV, the larger $\|\xi\|$ might become, depending on the underlying distortion. In some cases, one parameter does not fit distortion perfectly, that's why an additional distortion model [23] is added, which is referred as *Plumb Bob* or *Brown-Conrady* model. As schematically drawn in Fig. 3.2, it applies radial $\boldsymbol{\Pi}_{\mathrm{r}}$ and tangential distortion $\boldsymbol{\Pi}_{\mathrm{t}}$ to the initial (distortion-free) projected image points $(x, y)^T$:

$$\begin{pmatrix} x_{\mathrm{d}} \\ y_{\mathrm{d}} \end{pmatrix} = \boldsymbol{\Pi}_{\mathrm{r}} \begin{pmatrix} x \\ y \end{pmatrix} + \boldsymbol{\Pi}_{\mathrm{t}} \tag{3.4}$$

with

$$\begin{aligned} \boldsymbol{\Pi}_{\mathrm{r}} &= 1 + k_1 r^2 + k_2 r^4 + k_3 r^6 \\ \boldsymbol{\Pi}_{\mathrm{t}} &= \begin{pmatrix} 2p_1 xy + p_2\left(r^2 + 2x^2\right) \\ 2p_2 xy + p_1\left(r^2 + 2y^2\right) \end{pmatrix} \\ r^2 &= x^2 + y^2 . \end{aligned}$$

Radial distortion (k_1, k_2, k_3) depends on the optical characteristics of the lens and is point-symmetric at the optical center. This type of distortion can be devided into barrel distortion ($k_1, k_2, k_3 < 0$), which causes and outward shift of the image points and pincushion distortion ($k_1, k_2, k_3 > 0$) causing an inward shift of the image points. Tangential distortion (p_1, p_2) or *Thin Prism Distortion* [41, 22] is caused by improper lens and camera assembly such as imperfect centering and aligning of the lens components (e.g optical axis is not orthogonal to the sensor plane).
Finally, the homogeneous transformation from distorted image coordinates $(x_{\mathrm{d}}, y_{\mathrm{d}}, 1)^T$

on image plane $\boldsymbol{\pi}_\mathrm{i}$ to pixel coordinates $(u, v, 1)^T$ on pixel plane $\boldsymbol{\pi}_\mathrm{p}$ is given by

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{bmatrix} f_x & \alpha_\mathrm{s} & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x_\mathrm{d} \\ y_\mathrm{d} \\ 1 \end{pmatrix} . \tag{3.5}$$

Here, f_x and f_y denote the focal length (in pixel units) in horizontal (x-direction) and vertical (y-direction). $(c_x, c_y)^T$ denotes the principal point, which is the intersection between optical axis and pixel plane $\boldsymbol{\pi}_\mathrm{p}$. The coefficient α_s encodes the angles between horizontal and vertical sensor axes and is usually 0.
As can be seen, if $\xi = 0$ the UCM becomes the pinhole camera model with *Plumb Bob* distortion [19].

The Extended UCM [138] is a further development of the UCM. It is not considered in this work since UCM already achieved satisfying calibration results as shown in Fig. 4.4, page 57. Accordingly there was no need to additionaly implement this camera model.

Unprojection

The unprojection function transforms $(u, v)^T$ from pixel plane $\boldsymbol{\pi}_\mathrm{p}$ to $(\hat{X}, \hat{Y}, \hat{Z})^T$ on unit sphere $\mathcal{S}$. The transformation from pixel plane $\boldsymbol{\pi}_\mathrm{p}$ to image plane $\boldsymbol{\pi}_\mathrm{i}$ is the inverse of Eq. (3.5)

$$\begin{pmatrix} x_\mathrm{d} \\ y_\mathrm{d} \\ 1 \end{pmatrix} = \begin{bmatrix} f_x & \alpha_s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} . \tag{3.6}$$

In case of no distortion $(x, y)^T = (x_\mathrm{d}, y_\mathrm{d})^T$, the analytic solution to unproject $(x, y)^T$ from image plane $\boldsymbol{\pi}_\mathrm{i}$ to $(\hat{X}, \hat{Y}, \hat{Z})^T$ on unit sphere $\mathcal{S}$ is given by

$$\begin{pmatrix} \hat{X} \\ \hat{Y} \\ \hat{Z} \end{pmatrix} = \frac{\xi + \sqrt{1 + (1 - \xi^2)(x^2 + y^2)}}{x^2 + y^2 + 1} \begin{pmatrix} x \\ y \\ 1 - \xi \end{pmatrix} . \tag{3.7}$$

Undistortion

The applied *Plumb Bob* distortion model has no closed-form solution as undistortion function $(x_\mathrm{d}, y_\mathrm{d})^T \rightarrow (x, y)^T$. In [105], the authors describe serveral distortion correction approaches. The here presented one is an interative solution, based on the implementation from [146]

$$\begin{pmatrix} x \\ y \end{pmatrix}^{i+1} = \frac{1}{\boldsymbol{\Pi}^i_\mathrm{r}} \left(\begin{pmatrix} x_\mathrm{d} \\ y_\mathrm{d} \end{pmatrix} - \boldsymbol{\Pi}^i_\mathrm{t} \right) , \tag{3.8}$$

which is suitable for radial distortions up to two degrees ($p_1 \neq 0$, $p_2 \neq 0$, $p_3 = 0$). As Fig. 3.3 shows, the iterative solution from Eq. (3.8) for a radial distortion of three degrees obtains good undistortion results for points being close to the optical axis, since radial distortion increases with the distance to the optical center. In order to undistort the entire point set, a least squares non-linear optimization based on LM (Levenberg-Marquardt (non-linear solver)) is proposed

$$\underset{x,y}{\operatorname{argmin}} \sum \left\| \left(x_\mathrm{d}, y_\mathrm{d}\right)^T - \boldsymbol{\Pi}_\mathrm{r} \left(x, y\right)^T - \boldsymbol{\Pi}_\mathrm{t} \right\|^2 \tag{3.9}$$

with the corresponding Jacobian:

$$\begin{aligned} J &= \begin{bmatrix} \frac{\partial \boldsymbol{\Pi}_\mathrm{r}(x,y)^T}{\partial x} & \frac{\partial \boldsymbol{\Pi}_\mathrm{r}(x,y)^T}{\partial y} \\ \frac{\partial \boldsymbol{\Pi}_\mathrm{t}}{\partial x} & \frac{\partial \boldsymbol{\Pi}_\mathrm{t}}{\partial y} \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{\Pi}_\mathrm{r} + x^2 w & y^2 w \\ x^2 w & \boldsymbol{\Pi}_\mathrm{r} + y^2 w \\ 6p_2 x + 2p_1 y & 2p_1 x + 2p_2 y \\ 2p_1 x + 2p_2 y & 2p_2 x + 6p_1 y \end{bmatrix} \\ w &= 2k_1 + 4k_2 r^2 + 6k_3 r^4 \,, \end{aligned}$$

to further improve undistortion.

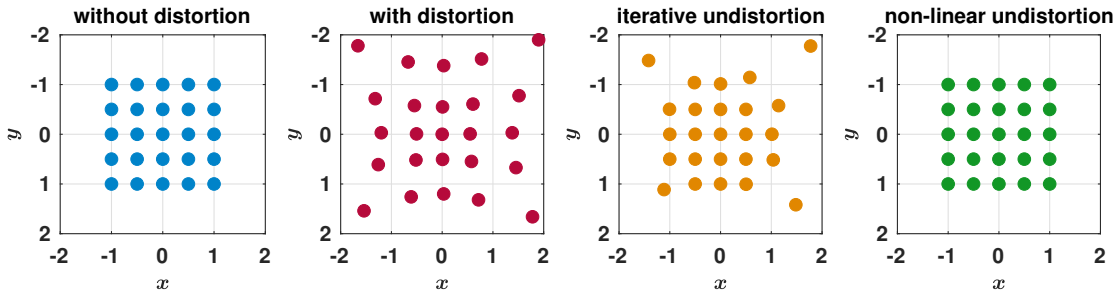

Fig. 3.3: Example of a *Plumb Bob* undistortion ($k_1 = 0.2, k_2 = 0.1, k_3 = -0.01, p_1 = -0.03, p_2 = 0.03$) illustrating the solutions after 10 linear iterations and after 7 non-linear LM optimization steps. As can be seen, the proposed non-linear optimization yields accurate undistortion results, whereas the iterative method obtains only satisfying results for points beeing closer to the optical axis $(x, y)^T = (0, 0)^T$.

3.2.2 Polynomal Camera Model

The Polynomal Camera Model (PCM) was first published in [182] and further improved by Scaramuzza in [215, 216, 219, 220]. In contrast to the UCM, which describes the projection $(X, Y, Z)^T$ from camera frame $\mathcal{C}$ to $(u, v)^T$ on pixel plane $\boldsymbol{\pi}_\mathrm{p}$ as mapping function, the PCM describes the unprojection from $(u, v)^T$ on pixel plane $\boldsymbol{\pi}_\mathrm{p}$ to a vector $(x, y, f(x, y))^T$ in camera frame $\mathcal{C}$, that equals the direction of the 3d point $(X, Y, Z)^T$. It is a generalized parametric model, which covers different kinds of omnidirectional dioptric and catadioptric cameras.

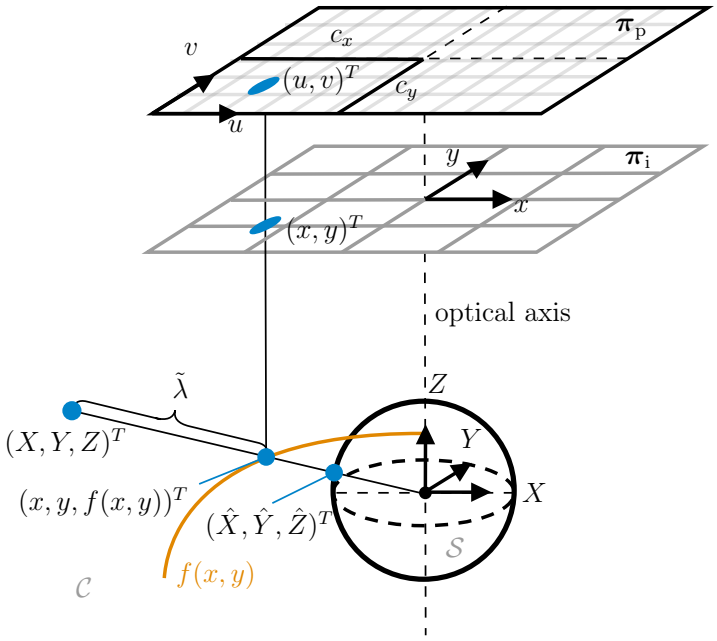

Fig. 3.4: Backward projection of the PCM

Unprojection

The inverse of an affine transformation converts a point from pixel coordinates $(u,v)^T$ on pixel plane $\boldsymbol{\pi}_\mathrm{p}$ to image coordinates $(x,y)^T$ on image plane $\boldsymbol{\pi}_\mathrm{i}$. This transformation is given in homogeneous form

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} c & d & c_x \\ e & 1 & c_y \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix}, \tag{3.10}$$

where c, d, e, c_x, c_y denote the affine parameters, incorporating the digitizing process and small axes misalignments. Similar to the UCM, the principal point $(c_x, c_y)^T$ indicates the intersection between optical axis and pixel plane $\boldsymbol{\pi}_\mathrm{p}$. The direction $(x, y, f(x,y)^T)$ of the 3d point (X,Y,Z) is calculated, with $f(x,y)$ being a rotationally symmetric, non-linear function based on a Taylor polynom of n degree:

$$\begin{aligned} f(x,y) &= a_0 + a_1 r + a_2 r^2 + a_3 r^3 + a_4 r^4 \ldots + a_n r^n \\ r &= \sqrt{x^2 + y^2}\,. \end{aligned} \tag{3.11}$$

$f(x,y)$ models the distortion of dioptric cameras (e.g. fisheye) and approximates the mirror shape of catadioptric cameras. As mentioned above, $(x, y, f(x,y)^T)$ is the di-

rection of (X, Y, Z) such that

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \tilde{\lambda} \begin{pmatrix} x \\ y \\ f(x,y) \end{pmatrix}, \tag{3.12}$$

with $\tilde{\lambda}$ being a positive scaling value (unscaled depth). Normalizing the obtained direction yields the point on unit sphere

$$\hat{\mathbf{X}} = \begin{pmatrix} \hat{X} \\ \hat{Y} \\ \hat{Z} \end{pmatrix} = \frac{(x, y, f(x,y))^T}{\|(x, y, f(x,y))^T\|}. \tag{3.13}$$

Projection

Projection is based on the inversion of $f(x, y)$. As there is no analytic solution for a polynom of degree n, the inversion is approximated by another Taylor polynom of degree m ($m > n$).

3.2.3 Spherical Camera Model

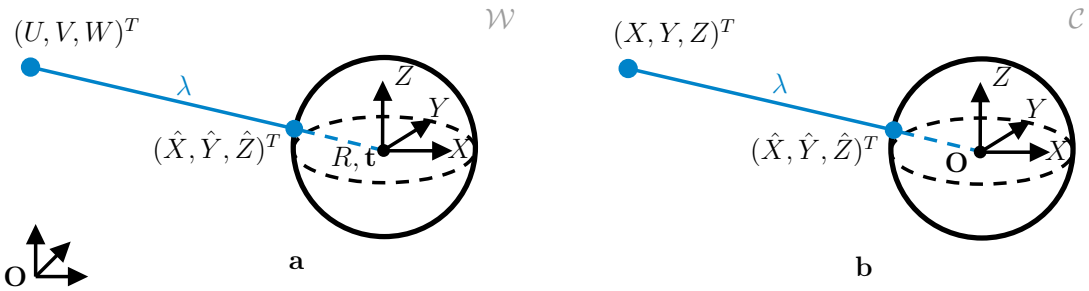

Fig. 3.5: Projection geometry on unit sphere: **a)** in world frame $\mathcal{W}$, **b)** in camera frame $\mathcal{C}$.

The Spherical Camera Model (SCM) considers a central camera as a unit sphere, where the surface of the sphere presents the image domain. This assumption enables a full omnidirectional projection since all directions a handled equally: There is no real front or back, up (zenit) and down (nadir). The SCM does not need prior knowledge about the underlying physical imaging system and can also work with world map projections from cartography (Section 5.2, page 78). Furthermore, the location of a projected point on unit sphere also represents its direction (in contrast to UCM and PCM, which need an unprojection function in order to obtain the corresponding direction).

The camera pose $[R, \mathbf{t}]$ describes the transformation from camera frame $\mathcal{C}$ to world frame $\mathcal{W}$. The inverse transformation from $\mathcal{W}$ to $\mathcal{C}$ is given by $\left[R^T, -R^T\mathbf{t}\right]$. The

forward projection (or simply projection) maps $\mathbf{U} \in \mathcal{W}$ to $\hat{\mathbf{X}} \in \mathcal{S}$ and is described by

$$\boxed{\hat{\mathbf{X}} = \frac{\mathbf{X}}{\|\mathbf{X}\|} = \frac{R^T(\mathbf{U}-\mathbf{t})}{\|R^T(\mathbf{U}-\mathbf{t})\|}}. \tag{3.14}$$

The inverse mapping is called backward projection (or simply back-projection) given by

$$\boxed{\mathbf{U} = R\hat{\mathbf{X}}\lambda + \mathbf{t}}. \tag{3.15}$$

The unit sphere $\mathcal{S}$ is a subspace of the camera frame $\mathcal{C}$. Since $\|R\hat{\mathbf{X}}\| = 1$, λ represents the entire depth from the optical camera center $[R, \mathbf{t}]$ to the 3d world point $\mathbf{U}$. The depth value can be re-obtained from the correspondence between image point and 3d world point. Taking Eq. (6.2) and subtracting $\mathbf{t}$ leads to

$$\underbrace{\mathbf{U}-\mathbf{t}}_{\mathbf{b}} = \underbrace{R\hat{\mathbf{X}}}_{\mathbf{a}}\lambda\,,$$

which can be solved for λ using the least squares solution $\lambda = \mathbf{a}^T\mathbf{b}/\|\mathbf{a}\|^2$ (this is explained in detail in Appendix A.2). This solution leads to the following equation

$$\boxed{\lambda = R\hat{\mathbf{X}}^T(\mathbf{U}-\mathbf{t})}. \tag{3.16}$$

3.3 P2S-Maps - Mapping onto Unit Sphere via Lookup Table

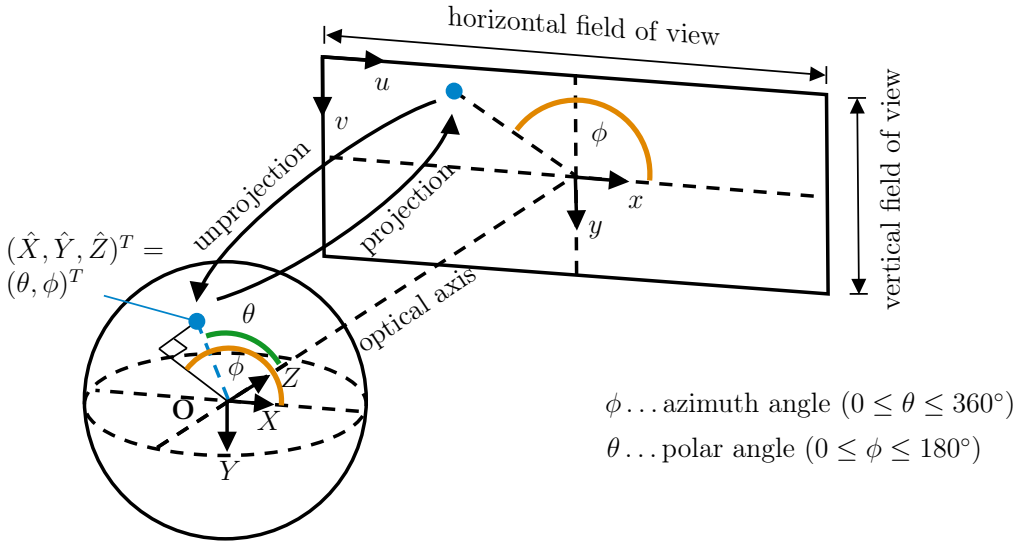

Fig. 3.6: Polar angle θ and azimuth angle ϕ explained at hand of a unit sphere and its corresponding perspective projection. Polar angle θ is around the optical axis $(0,0,1)^T$ and azimuth angle ϕ lies between otpical axis $(0,0,1)^T$ and $(\hat{X},\hat{Y},\hat{Z})^T$.

Both, UCM (Section 3.2.1) and PCM (Section 3.2.2) have a closed-form solution for pixel to unit sphere mapping, which is mandatory in order to use the generalized

SCM. However, here UCM is used with an additional *Plumb Bob* distortion model, which relates to higher computational effort for undistortion purposes. In order to reduce this load the unprojection (including undistortion) is performed for all image pixels and saved as lookup table, similar as recommended in [226]. This needs to be done only once and can be reused for later processing. The lookup table makes the mapping between pixel and unit sphere independent of the choosen camera calibration model as long as it provides an unprojection. This procedure preserves from additional implementation effort of specific projection functions into basic routines like PnP or BA. Camera calibration is replaced by a mapping between pixel and spherical coordinates as a lookup table and is no longer described as geometric (UCM) or analytic (PCM) mapping function.

Image coordinates on unit sphere may be expressed as either spherical coordinates $(\theta, \phi)^T$ or normalized Cartesian coordinates $(\hat{X}, \hat{Y}, \hat{Z})^T$ with the following conversion between them:

$$\theta(\hat{X}, \hat{Y}, \hat{Z}) = \operatorname{arccot}\left(\frac{\hat{Z}}{\sqrt{\hat{X}^2 + \hat{Y}^2}}\right) \cdot 180/\pi, \quad 0 \leq \theta \leq 360^\circ$$

$$\phi(\hat{X}, \hat{Y}, \hat{Z}) = \operatorname{atan2}\left(\frac{\hat{Y}}{\hat{X}}\right) \cdot 180/\pi + 90^\circ, \qquad 0 \leq \phi \leq 180^\circ\,,$$

as Fig. 3.6 explains. The polar angle θ is defined between the point $(\hat{X}, \hat{Y}, \hat{Z})^T$ and optical axis (Z-axis). ϕ denotes the azimuth angle around the optical axis between X-axis (and x-axis, respectively) and the orthogonal projection of the line segment between sphere center $\mathbf{O}$ and $(\hat{X}, \hat{Y}, \hat{Z})^T$ on X-Y-plane (and x-y-plane, respectively) in counterclockwise direction.
The lookup contains either $(u, v)^T \rightarrow (\theta, \phi)$ or $(u, v)^T \rightarrow (\hat{X}, \hat{Y}, \hat{Z})^T$ mapping correspondences. The first version needs less storage demand, however it may lead to confusion when converting back to Cartesian coordinates, since there exist different angle conventions. In order to prevent confusion, the lookup saves $(u, v)^T \rightarrow (\hat{X}, \hat{Y}, \hat{Z})^T$ mappings.

3.3.1 Lookup Table as Color Image

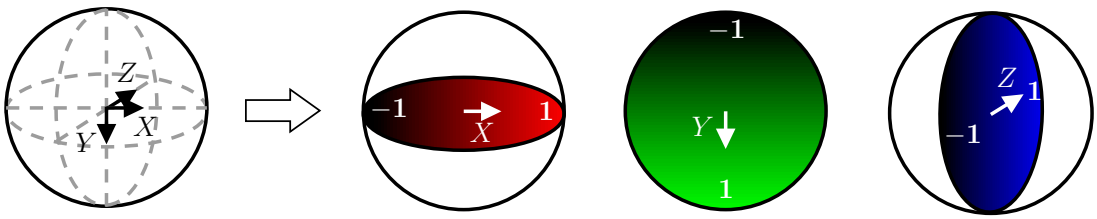

Fig. 3.7: Mapping of Cartesian coordiantes to color values. X is mapped to red, Y to green and Z to blue channel.

The proposed idea is to store the lookup as an image since these data type can be straightfowardly integrated into various applications and nearly all programming languages provide input/output functionality for most file formats. Each pixel in an image corresponds to RGB color values, that code the pixel's position on unit sphere and thus its direction. A similar approach is used for storing *object space normal maps*. However, storing location data as color values leads to precision loss, due to information limitation of each color channel.

The following describes an estimation of the maximum possible angular resolution $d\theta$ and $d\phi$ between neighboring points on unit sphere. Supposing θ and ϕ are stored as red and green color channel in a 48-bit image (3 color channels, 16-bit per channel). Consequently, the maximum angular resolution is:

$$\begin{aligned} d\theta &= 180°/2^{16} = 0.00275°/pix \\ d\phi &= 360°/2^{16} = 0.0055°/pix\,. \end{aligned}$$

As can be seen, the polar angle resolution $d\theta$ is twice the one of the azimuth angle $d\phi$. The blue channel stays empty thus this data allocation needs less storage demand, but does not use the entire data storage potential.

Supposing a linear mapping between Cartesian coordiantes on unit sphere and RGB color values

$$-1 \leq (\hat{X}, \hat{Y}, \hat{Z}) \leq 1 \leftrightarrow 0 \leq RGB \leq 2^{16} - 1$$

with

$$RGB = \frac{(\hat{X}, \hat{Y}, \hat{Z}) + 1}{2}(2^{16} - 1)$$

as depicted in Fig. 3.7. Each Cartesian dimension of the unit sphere ranges from -1 to 1 and is discretized by 2^{16} color values leading to a spatial Cartesian resolution $dX = dY = dZ = 2/2^{16} = 1/2^{15}$. Converting these values back to the corresponding angular resolutions $d\theta(\hat{X}, \hat{Y}, \hat{Z})$ and $d\phi(\hat{X}, \hat{Y}, \hat{Z})$ yields:

$$\begin{aligned} d\theta &= \left|\theta(0,1,0) - \theta(0,1,1/2^{15})\right| \\ &= \left|\text{arccot}(0) - arccot(1/1^{15})\right| \cdot 180/\pi \\ &= 0.001748°/pix \\ d\phi &= \left|\phi(1,0,0) - \phi(1,1/2^{15},0)\right| \\ &= \left|\text{atan2}(0) - atan2(1/1^{15})\right| \cdot 180/\pi \\ &= 0.001748°/pix\,. \end{aligned}$$

This allows to store higher angular resolutions $d\theta$ and $d\phi$, both at the same resolution level of $0.001748°/pix$. An equirectangular image with the size of

$205,950pix \times 102,975pix$ would use the full potential of the provided angular resolution.

In order to assess this resolution value, it is brought in relation to a real camera. At the moment on market, the *PhaseOne XF IQ4*[44] is one of the cameras with the highest image resolution of $14204 \times 10652pix$ (151 megapixels with a sensor size of $53.4mm \times 40mm$). Combined with a *Schneider Kreuznach 150mm LS f/2.8 IF*[45] telelens (one of the longest focal length lenses at this high resolution with 25° angle of view) requires a polar resolution $d\theta \approx 0.00176°/pix$ and an azimuthal one $d\phi \approx 0.0081°/pix$. These vaules can be still discretized by the resolution spectrum of the color-coding. The majority of cameras used in robotics and computer vision have a much lower pixel resolution since image data is aimed to be processed in real-time and consequently must be kept manageable. An overview of derived angular resolutions from camera calibration is shown in Fig. 4.5, page 58.
With increasing performance the use of higher resolution cameras will become feasable in the future, such that the minimum angular resolution boundary needs to be reduced by extending the 16-bit depth to 32-bit depth of each color channel, which is already supported by some image file formats. The lookup table as color-coded image is named P2S-map (Pixel-to-Sphere-map).

Pixels without mapping information are black. They cannot be mapped onto unit sphere since RGB= $(0,0,0)$ leads to $(\hat{X},\hat{Y},\hat{Z}) = (-1,-1,-1)$, which does not represent a point on unit sphere. Black colored regions indicate masked areas or areas that are outside the lens' FoV and hence are not covered by the unprojection function. Using an image editing programm the user is able to mask certain image areas by painting them black in the P2S-map.
Fig. 3.8 depicts an example image mapped onto unit sphere via its corresponding P2S-map compared to traditional mapping on the virtual image plane using intrinsic camera parameters (here: $f_x = 519.22, f_y = 479.46, c_x = 522.29, c_y = 272.73$).
Viewing the P2S-map in an image viewer gives the user direct information about the camera's image dimensions, about masked or uncalibrated areas that indicate unavailable mappings and - if the user is familiar with the color-coding - about the FoV. This allows a better comparison between different cameras (Fig. 4.5, page 58) and projections (Fig. 4.4, page 57).

Image Format

The P2S-map is saved as TIFF[46] (or short TIF) with 48-bits per pixel (16-bits per color channel) using a LZW lossless compression.

[44] https://www.phaseone.com/de-DE/Photography/XF-Camera-System/Camera-Configurations/XF-IQ4-150MP-Camera-System
[45] https://www.phaseone.com/de-DE/Photography/XF-Camera-System/Lenses/Schneider-Kreuznach-150mm-Blue-Ring
[46] https://www.itu.int/itudoc/itu-t/com16/tiff-fx/docs/tiff6.pdf

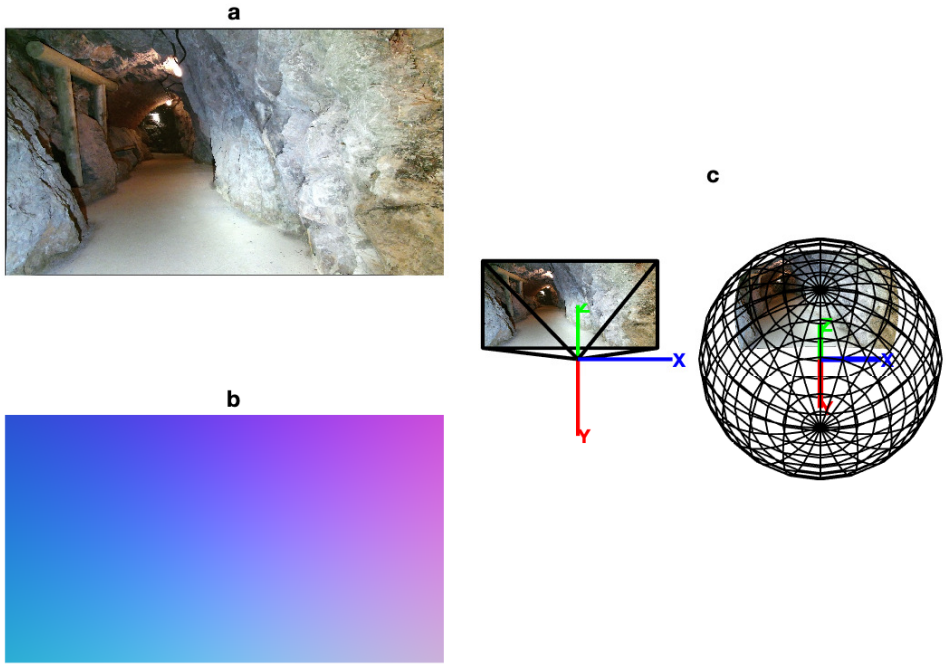

Fig. 3.8: Comparison between a standard pinhole projection and a corresponding sphere mapping using a P2S-map. The following is shown: **a)** color image, **b)** P2S-map as equation-free calibration format, **c)** image mapped on the virtual image plane as it is done in pinhole projection and image mapped onto unit sphere via P2S-map.

3.3.2 Lookup Interpolation

Image processing e.g. feature detection uses subpixel precision. Storing subpixels in a lookup at different precision levels would lead to a huge data amount, which cannot be managed efficiently. The proposed method maps subpixels from pixel plane $\boldsymbol{\pi}_\mathrm{p}$ onto unit sphere $\mathcal{S}$ by interpolating the lookup values of the surrounding pixels. Assuming $(u, v)^T$ to be a subpixel on pixel plane $\boldsymbol{\pi}_\mathrm{p}$ with unknown position $(\hat{X}, \hat{Y}, \hat{Z})^T$ on unit sphere $\mathcal{S}$ and $(u^i, v^j)^T$, $i, j = 1, 2$ to be the surrounding pixels with corresponding mappings $(\hat{X}^{i,j}, \hat{Y}^{i,j}, \hat{Z}^{i,j})^T$ as Fig. 3.9 depicts. Each $(u^i, v^i)^T$ lies on a pixel grid and has three corresponding function values $\hat{X}^{i,j}, \hat{Y}^{i,j}$ and $\hat{X}^{i,j}$. The four surrounding pixels are neighboring points on unit sphere, which means, the unknown mapping must lie between them. Supposing the geometry between these four points on unit sphere to be very small such that it can be approximated as plane. Then the relations between pixel distances equal relations between distances on the unit sphere. This assumption

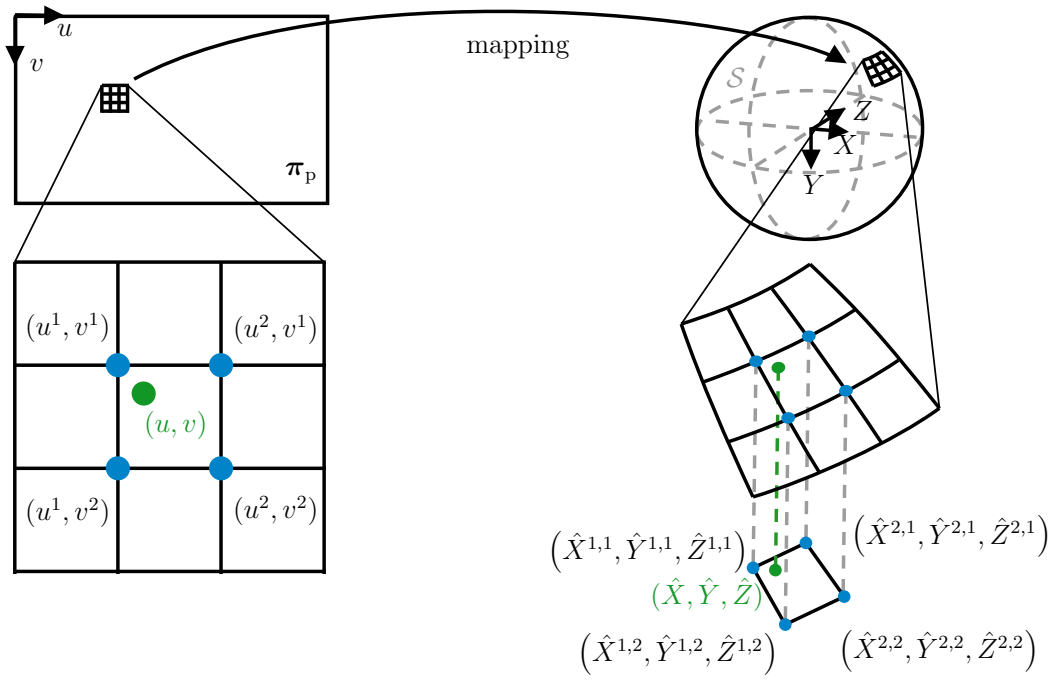

Fig. 3.9: Interpolation between lookup values in a P2S-map. Neighboring pixels are mapped onto unit sphere. Since the surrounded area between them is very small, it can be approximated as plane allowing a bilinear interpolation method.

allows to use a bilinear interpolation:

$$X = \frac{1}{(u^2-u^1)(v^2-v^1)} \begin{bmatrix} u^2-u & u-u^1 \end{bmatrix} \begin{bmatrix} \hat{X}^{1,1} & \hat{X}^{2,1} \\ \hat{X}^{1,2} & \hat{X}^{2,2} \end{bmatrix} \begin{pmatrix} v^2-v \\ v-v^1 \end{pmatrix}$$

$$Y = \frac{1}{(u^2-u^1)(v^2-v^1)} \begin{bmatrix} u^2-u & u-u^1 \end{bmatrix} \begin{bmatrix} \hat{Y}^{1,1} & \hat{Y}^{2,1} \\ \hat{Y}^{1,2} & \hat{Y}^{2,2} \end{bmatrix} \begin{pmatrix} v^2-v \\ v-v^1 \end{pmatrix}$$

$$Z = \frac{1}{(u^2-u^1)(v^2-v^1)} \begin{bmatrix} u^2-u & u-u^1 \end{bmatrix} \begin{bmatrix} \hat{Z}^{1,1} & \hat{Z}^{2,1} \\ \hat{Z}^{1,2} & \hat{Z}^{2,2} \end{bmatrix} \begin{pmatrix} v^2-v \\ v-v^1 \end{pmatrix}.$$

Since each direction value of (X, Y, Z) is obtained separately, the interpolated point does not necessarily lie on the sphere's surface and must be finally projected onto this $(\hat{X}, \hat{Y}, \hat{Z})^T = (X, Y, Z)^T / \left\|(X, Y, Z)^T\right\|$.

3.3.3 Depth Data Conversion

This section describes the conversion of image depth information to be directly used with a P2S-map. Image depth data from RGBD-cameras as well as from stereo cameras are usually stored as depth map using a 16-bit grayscale image in PNG file format. Each grayscale intensity relates to a depth value, enabling to store a depth range from $0mm$ to $65,535mm$ with a precision of $1mm$. This data format has become a quasi standard through *OpenNI / OpenNI2*. It is widely used in ROS[47]

[47] http://wiki.ros.org/depth_image_proc

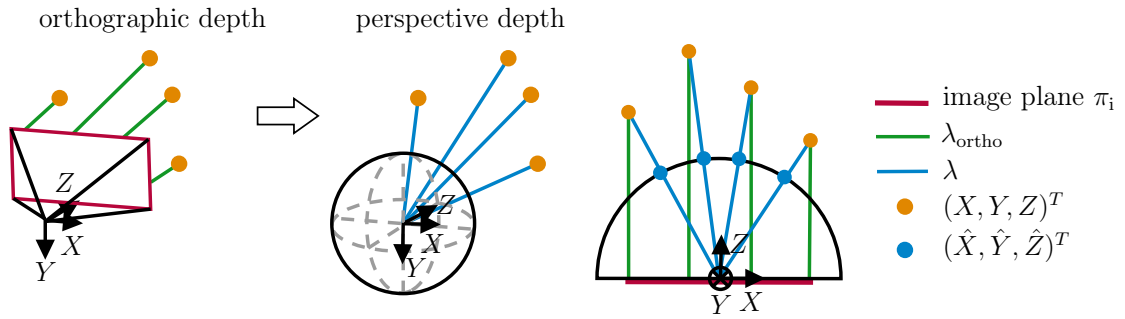

Fig. 3.10: Depth data conversion in order to use depth information e.g. from RGBD cameras with P2S-maps. RGBD cameras usually provide orthograpic depth data λ_{ortho}, whereas a camera sphere uses perspective depth data λ (similar to range data from laser scans).

and *OpenCV*[48] in combination with RGBD-cameras from *Kinect*, *Xtion* or *RealSense* product families, to name only a few popular ones.

Summarizing Eqs. (3.2) and (3.3), pages 36 and 37 for pinhole cameras ($\xi = 0$) without distortion leads to the projection function

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \frac{1}{Z} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

and its inversion, the back-projection function

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = Z \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} . \tag{3.17}$$

The depth map saves Z for each pixel directly, which leads to an orthographic depth representation $\lambda_{\text{ortho}} = Z$ as shown in Fig. 3.10. Storing orthographic depth information enables to directly compare depth values (especially in case of unkown intrinsics), since Z-values do not need camera intrinsics to be restored.

The back-projection for a camera sphere is given with

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \lambda \begin{pmatrix} \hat{X} \\ \hat{Y} \\ \hat{Z} \end{pmatrix} . \tag{3.18}$$

[48] https://docs.opencv.org/master/d2/d3a/group__rgbd.html

Here, λ denotes the perspetive depth, which is also designated as range data as this term is used for laser scan data. Equalizing Eqs. (3.17) and (3.18) leads to

$$\lambda_{\text{ortho}} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \lambda \begin{pmatrix} \hat{X} \\ \hat{Y} \\ \hat{Z} \end{pmatrix} . \tag{3.19}$$

Taking a closer look at the third row of Eq. (3.19) gives the relation between orthographic depth and perspective one

$$\lambda = \frac{\lambda_{\text{ortho}}}{\hat{Z}} . \tag{3.20}$$

Knowing this simple conversion allows to apply P2S-map on depth maps in order to recover 3d information.
Fig. 3.11 depicts a real world example using a depth map to back-project image points using a standard pinhole camera model and the genralized camera sphere model with P2S-map.

Brief Chapter Summary

A brief overview concerning camera types and their classifications was given in this chapter. Existing popular camera models were explained that are suitable for a wide range of camera types. The chapter introduced the idea to store the unprojection of each camera model as lookup, which maps image pixels onto unit sphere. Lookup mappings are further converted to color space in order to store be stored as color image that is called P2S-map. As a consequence of this lookup approach, the camera geometry becomes independent of the underlying camera calibration model and enables the usage of a general spherical camera model in order to process images from different camera types and projections models in one framework. Furthermore, a linear interpolation method was explained that allows to obtain mappings for intermediate pixels from a P2S-map, which is required for feature detection. As a last point the chapter described the conversion of depth data from e.g. RGBD-cameras to be used in combination with a P2S-map for back-projection purpose.

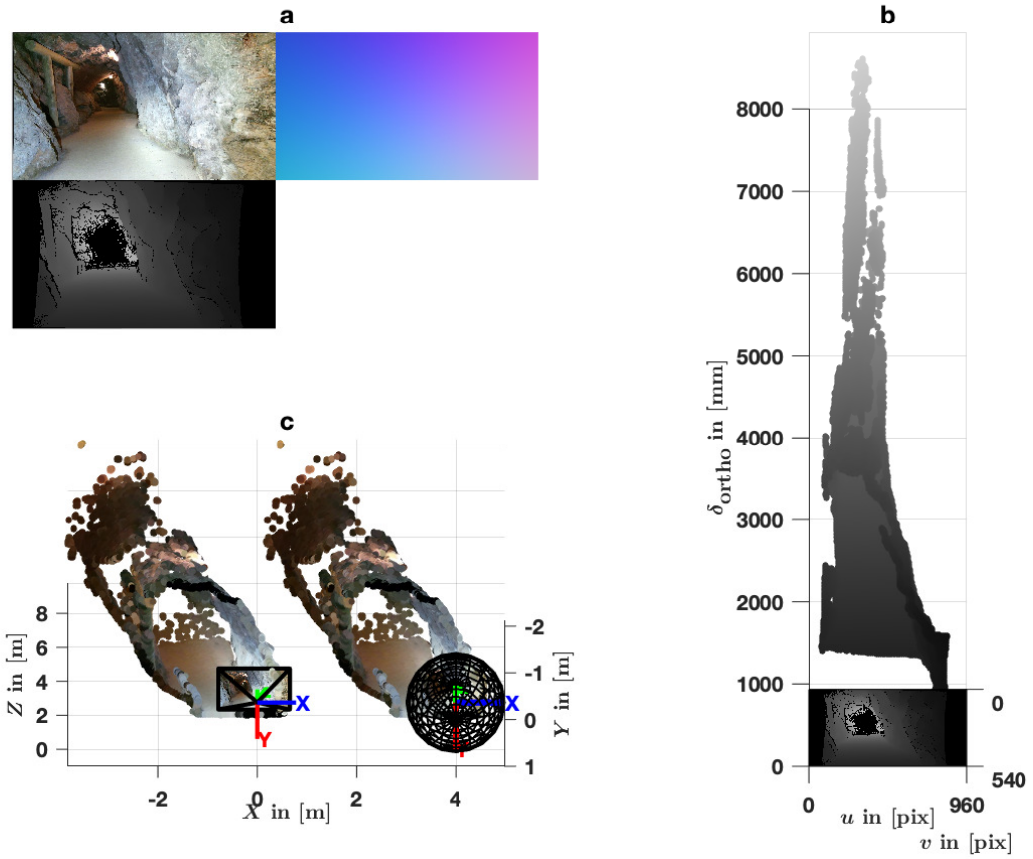

Fig. 3.11: Example conversion from orthographic depth to perspective depth. **a)** Visualization of color image, depth map and sphere-map (color-coded lookup), where the depth map is presented using a dynamic gray scaling. **b)** 3d visualization of the depth map with its corresponding depth information. The brightness of the pixel is proportional to orthographic depth δ_{ortho}, the brighter the pixel intensity, the further away. Black denotes missing depth information. **c)** Back-projection using the depth map with a pinhole camera model and converted to be used with a camera sphere (sphere is shifted by $(+4, 0, 0)^T$, to be placed beside the pinhole camera).

4 Calibration

Brief Chapter Overview

This chapter covers the topic of camera calibration. Section 4.1 gives an insight of the proposed processing structure, which is split into target detection, intrinsic calibration and extrinsic calibration. Section 4.2 describes a developed target detection toolbox based on availabe algorithms. It is followed by Section 4.3, which summarizes functions and algorithms from known toolboxes, which are incorporated in the presented intrinsic calibration toolbox. Intrinsic calibration is based on camera models from Chapter 3, page 33 and provides P2S-maps as output. Section 4.3.1 shows an example of an intrinsic calibration in detail and illustrates an overview concerning calibration results from different cameras.
Section 4.4 introduces a new extrinsic calibration worklflow for multi-camera systems based on P2S-maps. It describes the idea of adopting PGO to solve for camera extrinsics, which can be used to integrate multi-camera configurations into the SCME pipeline. Relative transformations between targets and camers are estimated by 3d-2d pose estimation as described in Section 4.4.1 or by 2d-2d pose estimation as described in Section 4.4.2. Section 4.4.3 explains the optimization of the derived transformation estimates and Section 4.4.4 reveals how to derive corresponding transformation uncertainties. Based on these information, Section 4.4.5 illustrates the generation of a pose graph, which is then optimized by a pose graph solver. The obtained extrinsics are further refined using BA as explained in Section 4.4.6. The chapter ends with Section 4.4.7 showing extrinsic calibration results from two challenging mutli-camera setups.

4.1 Overview of Proposed Calibration Pipeline

This chapter introduces a calibration routine consisting of three main parts. Most of the available calibration toolboxes use one specific camera model [19, 220, 181, 222, 264] and hence are limited to selected camera types. Their main goal is to intrinsically calibrate monocular camera systems. They also provide target poses from the calibration process itself, but these information are less important and mostly used for visualization purposes in order to validate the calibration results. There are also toolboxes providing an *all-in-one* solution [19, 161] by combining intrinsic and extrinsic calibration for multi-camera setups. However, this leads to problems if one of the included routine fails, making the entire calibration process cumbersome.

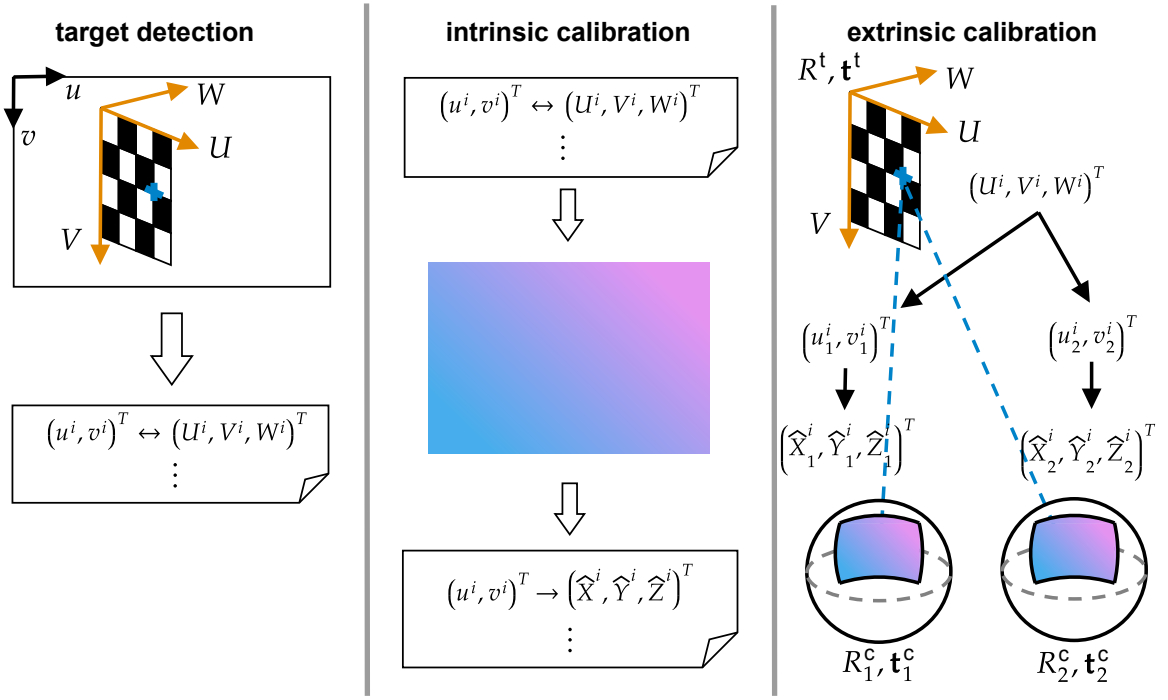

Fig. 4.1: Overview of the proposed camera calibration pipeline being split into three subroutines. **Target dectection** analyzes calibration images, detects checkerboard patterns and saves the relations between detected pattern points and real world coordinates $(u^i, v^i)^T \leftrightarrow (U^i, V^i, W^i)^T$. **Intrinsic calibration** uses these correspondences to obtain intrinsic parameters and to provide a corresponding P2S-map to unproject each image pixel onto unit sphere $(u^i, v^i)^T \rightarrow (\hat{X}^i, \hat{Y}^i, \hat{Z}^i)^T$. **Extrinsic calibration** obtains each camera pose in a multi-camera setup, where two or more cameras capture a calibration target at the same time. It requires point correspondences from **target dectection** and mapping information from **intrinsic calibration**.

The proposed processing pipeline splits the calibration routine into three separate subroutines: **target detection** (Section 4.2, page 53), **intrinsic calibration** (Section 4.3, page 54) and **extrinsic calibration** (Section 4.4, page 58) as Fig. 4.1 illustrates.
The following adjustments make the entire calibration process simpler, more flexible and applicable to various calibration scenarios:

- **Target detection** uses multiple detection algorithms (which are independent of the camera type) in order to decrease the number of false/failed detections. The user able to interfere e.g. to correct false detection or to manually select target points if required.

- **Intrinsic calibration** incorporates UCM and PCM covering a wide range of camera types in order to obtain intrinsic parameters using pattern point correspondences from **target detection**. Based on these parameters it provides lookup mappings from pixel to unit sphere (P2S-map).

- **Extrinsic calibration** obtains extrinsic parameters (rotation and translation) of each camera and target in a multi-camera setup and is not limitied to a certain number of cameras. It is based on SCM and thus enables to calibrate setups con-

sisting of mixed camera types. The subroutine requires pattern point correspondences from **target detection** as well as P2S-maps from **intrinisc calibration** as input data.

The modular structure also allows to replace or adapt certain subroutines if needed and enables to extend the toolbox' functionality.

In contrast to [19, 161], the presented pipeline seperates intrinsic from extrinsic calibration in order to work with different calibration image sets as well as patterns, which may differ in size (e.g. number of pattern points and grid spacing) and dimensionality (2d or 3d). Intrinsic calibration requires a set of images at different viewpoints and distances covering the entire FoV of the camera as Fig. 4.3 **d)**, page 56 shows. Extrinsic calibration is based on overlapping FoVs, requiring the pattern the be seen in at least two cameras. Depending on the camera setup, this requires to place target poses further away form the camera center or to use a smaller pattern compared to intrinsic calibration.

4.2 Target Detection

Checkerboards are mostly used as target pattern since they are straightforward to manufacture, cheap to employ and their corners can be detected with sup-pixel accuracy even under strong lens distortion [73], which makes them suitable to be used with a wide range of omnidirectional cameras.

The traget detection uses geometric knowledge of the pattern and assigns each detected point of the checkerboard to a real world coordinate $(u^i, v^i)^T \leftrightarrow (U^i, V^i, W^i)^T$, $i = 1, \ldots, p$. Fig. 4.2 presents the toolbox analyzing a set of calibration images from a fisheye camera *Kodak SP360 4K*. The presented traget detection incorporates three main algorithms, which do not need additional knowledge about the underlying lens distortion:

- **Rufli's automatic corner finder** software[49] [213] for blurred and distorted images, which is used by *OcamCalib*-toolbox [216, 220].
- **Geiger's corner and checkerboard detection** [73] called *LIBCDDETECT*[50], which detects multiple checkerboards in a single shot and is suitable for different camera types.
- **Bouguet's corner finder** [19], which is based on *Harris Corner Detection* and refines corner estimates from automatic and manual detection.

[49] https://sites.google.com/site/scarabotix/ocamcalib-toolbox/ocamcalib-toolbox-download-page
[50] http://www.cvlibs.net/software/libcbdetect/

Furthermore, the toolbox analyzes a set of calibration images using different detection algorithms in order to reduce the number of incorrect or aborted detections, if one algorithms fails. The toolbox also allows manual intervention by the user e.g. to select target points manually.

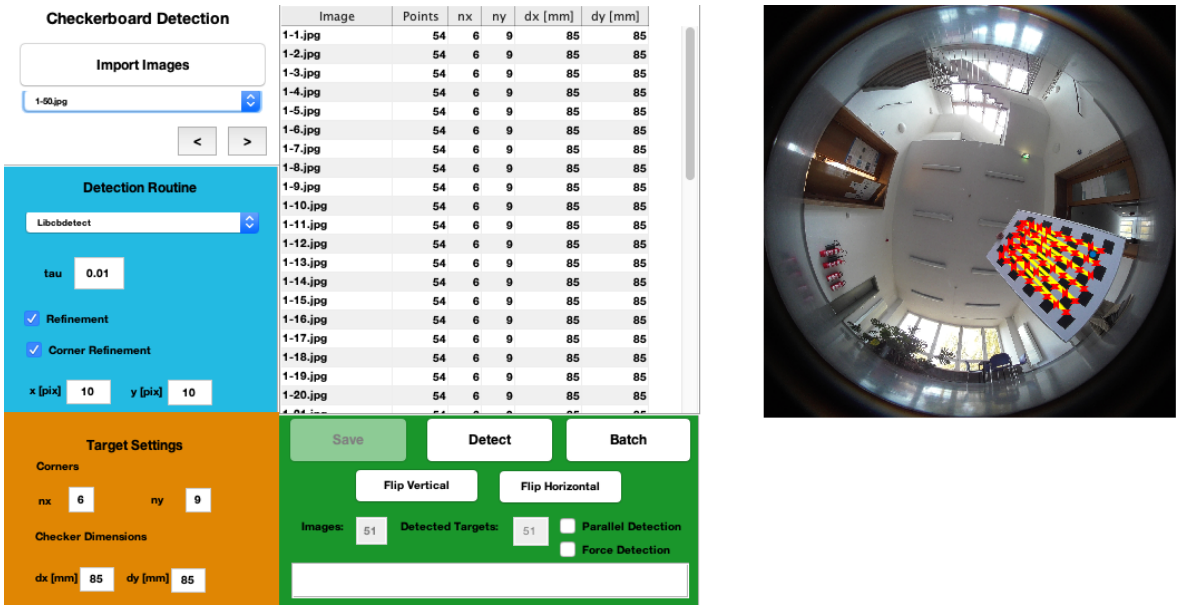

Fig. 4.2: Checkerboard detection GUI showing the parameter settings panel on the left and a preview of the detected pattern points in the camera image on the right.

4.3 Intrinsic Calibration

The developed intrinsic calibration toolbox uses UCM (Section 3.2.1, page 35) and PCM (Section 3.2.2, page 39) and incorporates functions/routines from the following existing toolboxes:

- **Bouguet's Camera Calibration Toolbox for Matlab** [19], which is probably the most popular calibration software (according to [73]) originally written in *Matlab*. It was later implemented in *OpenCV*[51] with automatic checkerboard detection. The software provides a complete calibration pipeline including semi-manual checkerboard detection, intrinsic calibration and extrinsic calibration for stereo-setups. It is mainly inspired by Heikkilä's method [105] (describing a whole calibration process in four steps, including image correction due to distortion), Zhang's method [283, 282] (using a planar calibration pattern in order to describe the relation between world and image coordinates as homography) and Tsai's method [260] (describing a stepwise procedure to incrementally solve for unknown parameters which also cooperates with non-planar calibration patterns). Bouguet's toolbox comes with a GUI for better user interaction. It is based on a pinhole model in conjunction with *Plumb Bob* / *Brown-Conrady*-distortion [23],

[51] https://docs.opencv.org/3.4/d4/d94/tutorial_camera_calibration.html

modelling radial and tangential (*Thin-Prism*) distortion [41, 22]. In this work obtained intrinsic parameters are converted to UCM.

- **Mei's Omnidirectional Calibration Toolbox**[52] [181] introducing UCM as Section 3.2.1, page 35 describes in detail.
- **Scaramuzza's OCamCalib (Omnidirectional Camera Calibration) for Matlab**[53] [220] proposing PCM as Section 3.2.2, page 39 explains. In this work it is used with an improved non-linear optimization routine[54] from [263].
- **Li's Multiple-Camera System Calibration Toolbox for Matlab**[55] [161], which is also based on UCM, but uses a different approach to obtain initial parameters compared to Mei.

Mei's as well as Scaramuzza's toolbox are inspired by Bouguet's work. The presented toolbox allows to interchange obtained parameters between algorithms from existing toolboxes in order to make use of each algorithm's strength. Results from one algorithm can be used as initial estimates in an other one (as long as they share the same parameters) in order to achieve robust calibration results. The developed toolbox provides intrinsic camera parameters, target poses, P2S-map with a corresponding mesh, which is required for projection conversion as described in Fig. 5.5, page 82. Fig. 4.3 depicts detailed information about the calibration results of the *Kodak SP360 4K* as an example.

4.3.1 Selected Examples

The proposed calibration toolbox is used to obtain intrinsic parameters from different cameras. Fig. 4.4 presents an overview of selected cameras, their intrinsic parameters and the corresponding P2S-maps, indicating the calibrated FoVs.

Comparing calibration results from various cameras is not straightforward. A P2S-map augments metric information to an image, such that each pixel's ray direction can be recovered directly. The distances between neighboring pixels in polar ($d\theta$) and azimuthal ($d\phi$) direction are calculated in order to obtain the cameras angular resolution. This enables a better camera comparison oppertunity, since the angular resolution is scale invariant and accordingly independent of the real pixel size. The mean angular resolutions $\bar{d\theta}$ and $\bar{d\phi}$ are obtained from P2S-maps and illustrated in Fig. 4.5. As can be seen, the mean polar resolution $\bar{d\theta}$ is higher (the smaller the value, the higher the resolution) than the azimuthal one $\bar{d\phi}$ for all cameras. This is caused due to the circumstance that ϕ represents the angle around the optical axis (Fig. 3.6, page 42) and hence always covers 360°, whereas θ ranges up to 180° and

[52] https://www.robots.ox.ac.uk/~cmei/Toolbox.html
[53] https://sites.google.com/site/scarabotix/ocamcalib-toolbox
[54] https://github.com/urbste/ImprovedOcamCalib
[55] https://github.com/prclibo/calibration-toolbox

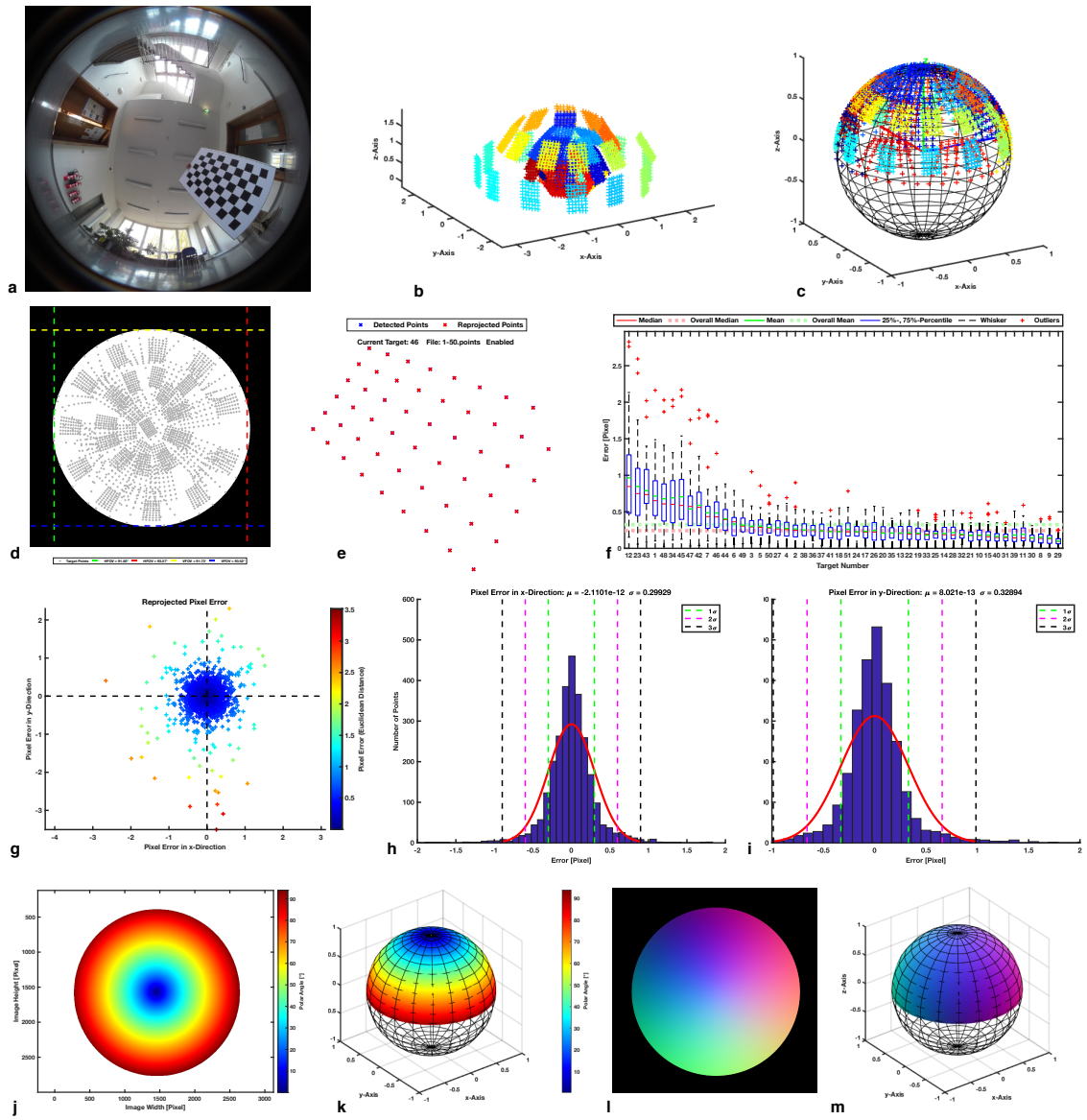

Fig. 4.3: Example of the *Kodak SP360 4K* calibration results. The intrinsic camera calibration provides an overview of the following information: **a)** shows the reference calibration image. **b)** obtained target poses in relation to the camera center, **c)** unprojected pattern points on camera sphere, **d)** detected pattern points of all calibration images indicating the calibrated FoV, **e)** detected (blue) and projected (red) pattern points of the reference calibration image, **f)** statistical overview of the projection error, **g)** projection error in x- and y-direction, **h)** gaussian distribution of the projection error in x-direction, **i)** gaussian distribution of the projection error in y-direction, **j)** obtained polar angle θ map of the calibrated FoV on image plane, angular resolution decreases towards the image boarder, due to increasing image distortion **k)** polar angle θ map on unit sphere, **l)** P2S-map and **m)** P2S-map on unit sphere.

depends on the camera's horizontal and vertical FoV. The mean polar resolution $\bar{d\theta}$ ranges from $0.09°/pix$ (*Ricoh Theta S*) to $0.015°/pix$ (*Fuji X-T2 + Samyang 8mm*) and the mean azimuthal resolution $\bar{d\phi}$ ranges from $0.12°/pix$ (*Ricoh Theta S*) to $0.02°/pix$ (*Fuji X-T2 + Samyang 8mm*).

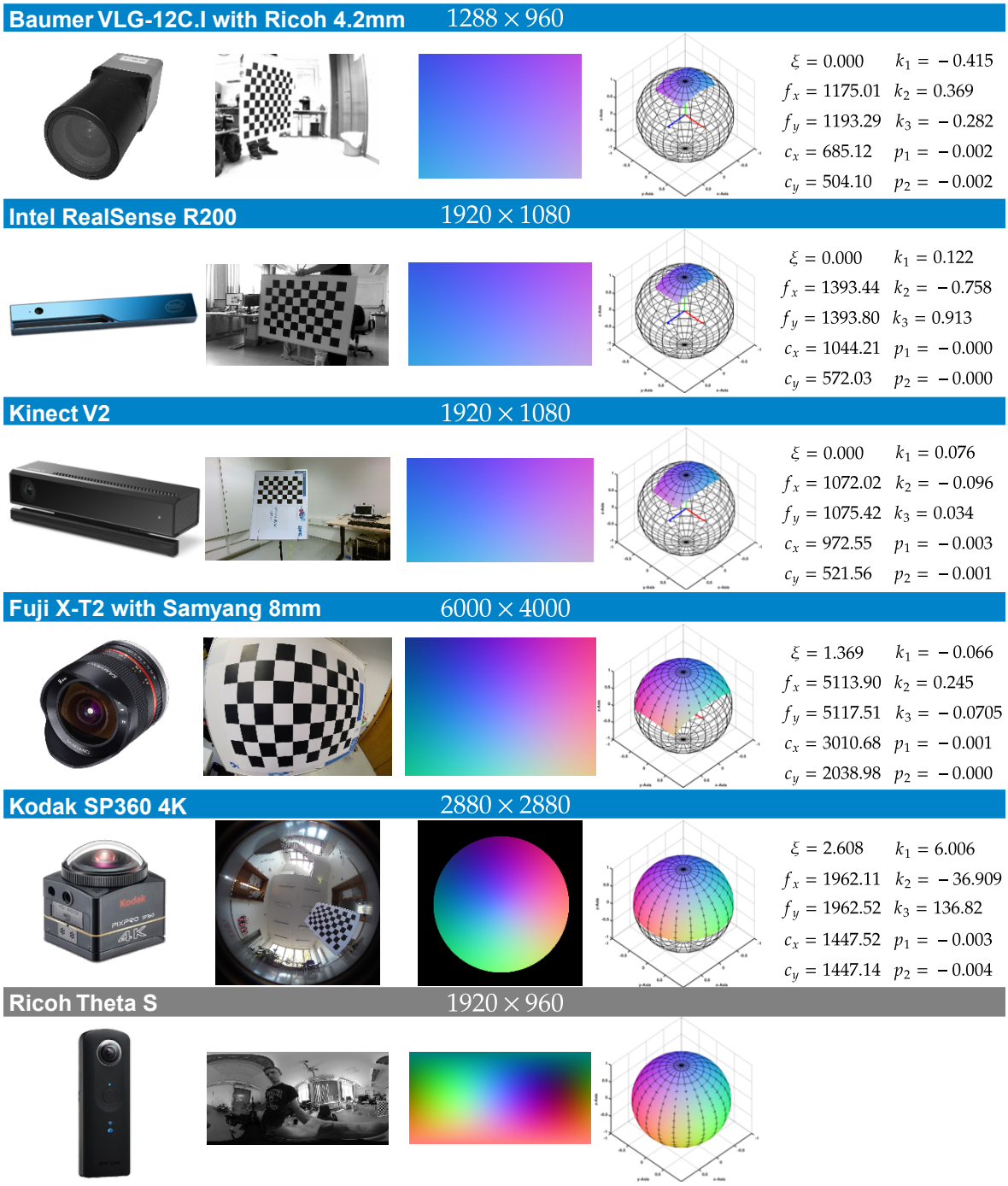

Fig. 4.4: Intrinsic camera calibration results of selected cameras using UCM. *Ricoh Theta S* is a stitching camera based on internal calibration and deals as reference for a full omnidirectional camera. The 1st column depicts the cameras and the 2nd one shows corresponding calibration images. The 3rd column illustrates the resulting unprojections as P2S-map and their pixel dimensions (width × height). The 4th column visualizes each camera's calibrated FoV. The corresponding intrinsic parameters for UCM are listed in the 5th column.

Note on Image Noise Representation

In literature image noise as well as uncertainties are usually expressed as pixel deviation. Comparing distance measures in pixel units is only suitable for cameras with similiar specifications e.g. pixel size, sensor resolution and FoV. Especially pixel size

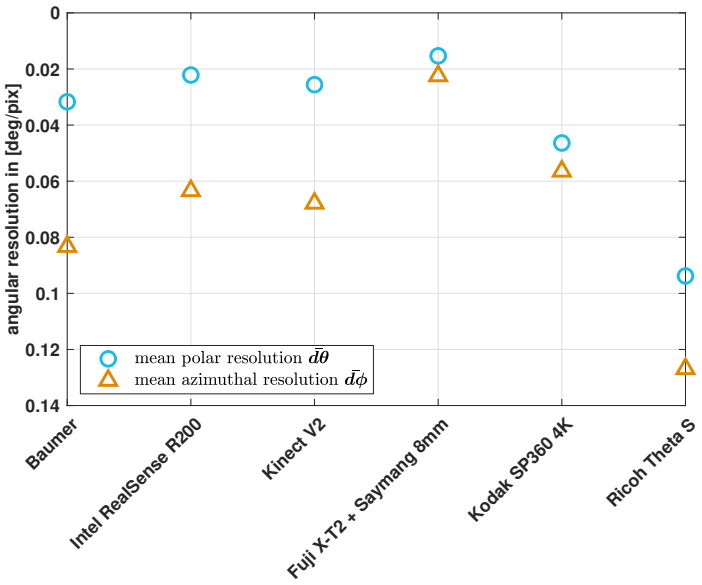

Fig. 4.5: Overview over the mean polar resolution $\bar{d\theta}$ and mean azimuthal resolution $\bar{d\phi}$ obtained from camera calibration.

information is not always given. Pixel units do not represent metric information, since a pixel's real world dimension depends on the pixel size and may vary from sensor to sensor.
Describing image noise as deviation in polar and azimuthal direction allows to define a metric distance, which is independent of the choosen camera system. Evaluation results in angular dimensions can be applied to every camera by back-converting angular distances to pixel units using the inverse of the camera's (mean) angular resolutions, and thus allowing to compare different camera types.
In this work the described image noise representation as angular distances in polar and azimuthal direction is used for evalution purposes.

4.4 Extrinsic Calibration

This work introduces an extrinsic multi-camera calibration, that is not restricted to stereo setups as it is the case for most calibration toolboxes [19, 35, 222], being known to the author. It is based on SCM and enables to calibrate different camera types even in a mixed setup as already shown in [34, 258]. It uses pose estimation techniques (Sections 4.4.1 and 4.4.2, pages 60 and 62) in conjunction with pose optimization (Section 4.4.3, page 66) to determine the relative transformations between targets and cameras as well as its corresponsing transformation uncertainties (Section 4.4.4, page 67) in order to describe the calibration scenario as pose graph (Section 4.4.5, page 67). This pose graph is solved by a pose graph solver (Section 4.4.5, page 67) obtaining camera and target poses, which may be refined in an additional BA step (Section 4.4.6, page 69).

The proposed routine is inspired by Li's multi-camera calibration toolbox [161] describing the extrinsic camera scenario as pose graph, where nodes represent cameras as well as targets and edges respresent the relative transformations between them. One camera node is assigned to be the reference frame and represents the root for an extracted spanning-tree. The remaining nodes are transferred into the gobal reference frame by traversing the spanning-tree in order to calculate initial camera and target poses. Li's toolbox is based on UCM and uses a BA-like optimization routine to refine camera and target poses by minimizing the projection error of the pattern points. Also to be mentioned as side note, Li's calibration process uses a SURF pattern instead of a checkerboard, which is matched across all images. This approach makes the calibration process more convenient, since only a part of the calbration pattern must be captured. However tests showed, that feature detection as well as feature matching failed under strong distortions.
This work adopts Li's idea to reformulate the extrinsic calibration scenario as pose graph but uses an external pose graph solver (Chapter 8, page 139) to initialize pose estimates and to perfom PGO in order to obtain optimized camera and target poses. Additional refinement is achieved through BA based on SCM, which is adopted from the described methods in [161, 167].

The multi-camera setup consists of rigidly connected and synchronized cameras to capture a calibration pattern, which is moved around the camera system. This also requires that each camera's FoV partly overlaps with at least one from another camera.

The proposed extrinsic calibration method is best described by means of an example as Fig. 4.6 illustrates. $[R_k^{\mathrm{c}}, \mathbf{t}_k^{\mathrm{c}}]$ denotes the k^{th} unknown camera pose and $[R_l^{\mathrm{t}}, \mathbf{t}_l^{\mathrm{t}}]$ denotes the l^{th} unknown target pose. $[R_{kl}, \mathbf{t}_{kl}]$ indicates the obtained relative transformation from target pose $[R_l^{\mathrm{t}}, \mathbf{t}_l^{\mathrm{t}}]$ to camera pose $[R_k^{\mathrm{c}}, \mathbf{t}_k^{\mathrm{c}}]$. The presented example consists of four cameras ($k = 1, \ldots, 4$), that are connected via four target poses ($l = 1, \ldots, 4$), which outlines a minimum case. More target poses are recommended as Fig. 4.6 implies.

The target pattern consists of p calibration points $\mathbf{U}^i$, $i = 1, \ldots, p$ in real world dimensions. Their corresponding image projections $(u_{kl}^i, v_{kl}^i)^T$ are obtained by target detection (Section 4.2, page 53). Here, $(u_{kl}^i, v_{kl}^i)^T$ denotes the projection of $\mathbf{U}^i$ from the l^{th} target pose into the k^{th} camera, which is further mapped onto unit sphere using a P2S-map from intrinsic camera calibration (Section 4.3, page 54) to obtain $\hat{\mathbf{X}}_{kl}^i$. The proposed extrinsic calibration uses $\mathbf{U}^i \leftrightarrow \hat{\mathbf{X}}_{kl}^i$ as input data.

For each target l that is captured by camera k a pose estimation (either from 3d-2d or 2d-2d correspondences) is processed in order to obtain an initial relative transformation $[R_{kl}, \mathbf{t}_{kl}]$.

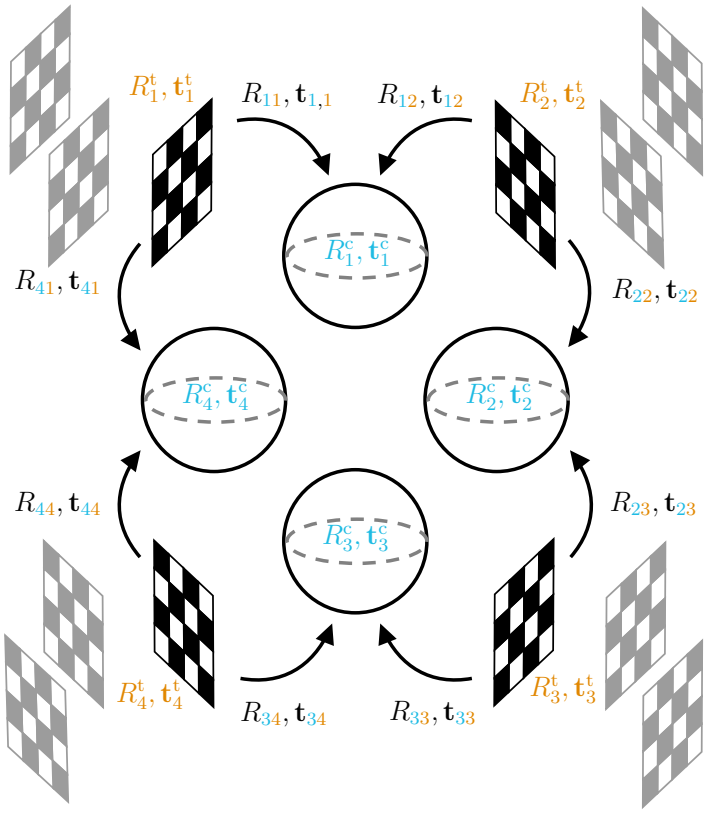

Fig. 4.6: Extrinsic camera calibration example with a sensor setup consisting of four partly overlapping cameras connected by four calibration target poses to solve the extrinsic calibration as pose graph. Additional target poses are recommended to improve calibration results. In this example they are greyed out, since they are not mandatory to explain the basic concepts of the proposed method.

4.4.1 3D-2D Pose Estimation

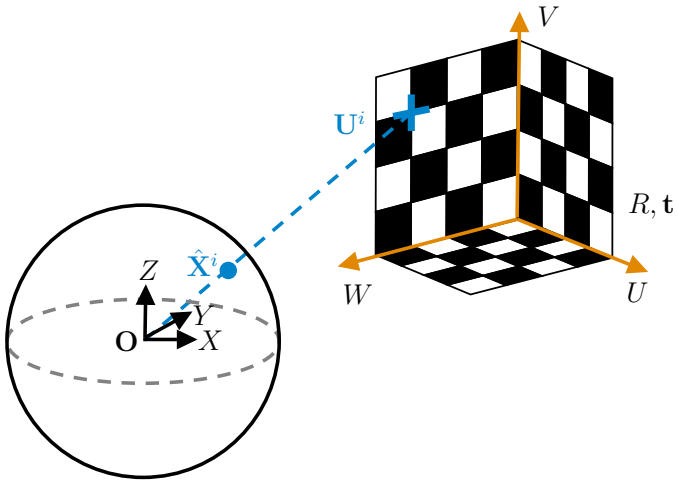

Fig. 4.7: Pose estimation from known 3d objects.

This section describes a pose estimation approach to obtain the relative transformation $[R_{kl}, \mathbf{t}_{kl}]$ between k^{th} camera and l^{th} target. PnP algorithms using 3d-2d feature correspondences play an important role in a SfM (Section 2.2, page 19) and VO (Section 2.1.1, page 17) pipeline to register a camera image to a known 3d structure. It

is also used in calibration scenarios to obtain relative pose estimates from 3d targets. To the best of the author's knowledge this topic hasn't been explained for spherical cameras in detail by literature so far and as part of the extrinisc calibration pipeline it is describe in the following.
For the sake of better readability the indication i, j is omitted, thus describing the proposed pose estimation method for a general case.
Given a set of correspondences $\mathbf{U}^i \leftrightarrow \hat{\mathbf{X}}^i$, $i = 1, \ldots, p$ between known 3d points $\mathbf{U}^i$ and their projections on unit sphere $\hat{\mathbf{X}}^i$ such that

$$\hat{\mathbf{X}}^i \times \left(R\mathbf{U}^i + \mathbf{t}\right) = \mathbf{0}\,, \tag{4.1}$$

as mentioned in [147] and illustrated in Fig. 4.7. This relation is brought into a DLT-form

$$B^i\mathbf{x} = \mathbf{0} \tag{4.2}$$

with

$$B^i = \left[\ \left([\hat{\mathbf{X}}^i]_\times \otimes \mathbf{U}^i\right)^T \ \middle|\ [\hat{\mathbf{X}}^i]_\times\ \right] \in \mathbb{R}^{3\times 12} \tag{4.3}$$

and $\mathbf{x}$ containing the unknown transformation elements

$$\mathbf{x} = \left(r_{11}, r_{12}, r_{13}, r_{21}, r_{22}, r_{23}, r_{31}, r_{32}, r_{33}, t_1, t_2, t_3\right)^T \in \mathbb{R}^{12\times 1}\,. \tag{4.4}$$

The third row of B^i is linearly dependent [196] such that $\operatorname{rank}(B^i) = 2$. This means at least six point correspondences are needed in order to solve the linear system of equations for $\mathbf{x}$, which is achieved by stacking all available point correspondences

$$B = \begin{bmatrix} B^1_{(1,:)} \\ B^1_{(2,:)} \\ \vdots \\ B^p_{(1,:)} \\ B^p_{(2,:)} \end{bmatrix} \in \mathbb{R}^{2p\times 12}\,. \tag{4.5}$$

Applying an SVD to B with

$$B = U\Sigma V^T \tag{4.6}$$

obtains the right unitary matrix $V \in \mathbb{R}^{12\times 12}$. The last column $V_{(:,12)}$ corresponds to the smallest singular value in $\Sigma \in \mathbb{R}^{12\times 12}$, which represents the solution $\mathbf{x} = V_{(:,12)}$. Before applying the SVD, normalizing each row of B leads to a more stable numerical solution. $\tilde{R}$ and $\tilde{\mathbf{t}}$ are extracted from $\mathbf{x}$ such that:

$$\tilde{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{4.7}$$

$$\tilde{\mathbf{t}} = \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix}\,. \tag{4.8}$$

The elements of $\tilde{R}$ are obtained in least squares sense without considering rotation matrix properties. Forcing $\tilde{R}$ to be orthogonal using an SVD

$$\tilde{R} = U_{\tilde{R}} \Sigma_{\tilde{R}} V_{\tilde{R}}^T \tag{4.9}$$

obtains $U_{\tilde{R}} \in \mathbb{R}^{3\times3}$ and $V_{\tilde{R}} \in \mathbb{R}^{3\times3}$ forming an orthogonal rotation matrix

$$R = U_{\tilde{R}} V_{\tilde{R}}^T, \tag{4.10}$$

subjecting $\|R\|_{\mathrm{F}} = 1$. However, this requires a rescaling of the translation

$$\mathbf{t} = \frac{\tilde{\mathbf{t}}}{\mathrm{diag}(\Sigma_{\tilde{R}})} \tag{4.11}$$

using the three singular values $\sigma_1, \sigma_2, \sigma_3$ embedded in the diagonal matrix $\Sigma_{\tilde{R}} \in \mathbb{R}^{3\times3}$.

Resolve Ambiguity from 3D-2D Pose Estimation

Since $\mathbf{x}$ is the solution based on collinearity, its sign is arbitrary yielding two solutions $[R, \mathbf{t}]$ and $[-R, -\mathbf{t}]$. Validating cheirality (Section 6.4.1, page 97) by obtaining the viewing direction of the first correspondence pair $\mathbf{U}^1 \leftrightarrow \hat{\mathbf{X}}^1$ as recommended in [196] resolves the correct transformation solution:

$$[R, \mathbf{t}] = \begin{cases} [R, \mathbf{t}], & \text{if } \hat{\mathbf{X}}^1 \bullet \left(R\mathbf{U}^1 + \mathbf{t}\right) > 0 \\ [-R, -\mathbf{t}], & \text{otherwise}\,. \end{cases}$$

However, tests turned out in case of noisy data it is recommended to check the viewing direction of each point correspondence $\mathbf{U}^i \leftrightarrow \hat{\mathbf{X}}^i$. The correct transformation is then chosen according the quantity of positive and negative directions.

4.4.2 2D-2D Pose Estimation

The following proposes a newly developed pose estimation method from 2d-2d correspondences as it is used for pose estimation from planar targets. As already noted in the previous section the indication k, l is omitted in order to describe a general case. Planar checkerboards are convenient for manufacturing and hence are widely used for camera calibration. Given a set of correspondences $\mathbf{U}^i \leftrightarrow \hat{\mathbf{X}}^i$, $i = 1, \ldots, p$ between known points from a planar pattern $\mathbf{U}^i$ and their corresponding projections on camera sphere $\hat{\mathbf{X}}^i = (\hat{X}^i, \hat{Y}^i, \hat{Z}^i)^T$ as Fig. 4.8 illustrates.

Planar objects - also referred as degenerate structure - cause a rank-deficiency when using the pose estimation method from Section 4.4.1, page 60. The following proposes a reliable method to obtain pose estimates from planar objects. Planar objects are always observed under $FoV < 180°$ (in both, horizontal and vertical direction), which equals the projection of a directional camera. This allows the usage of state-of-the-art PnP-algorithms, which are developed for calibrated perspective

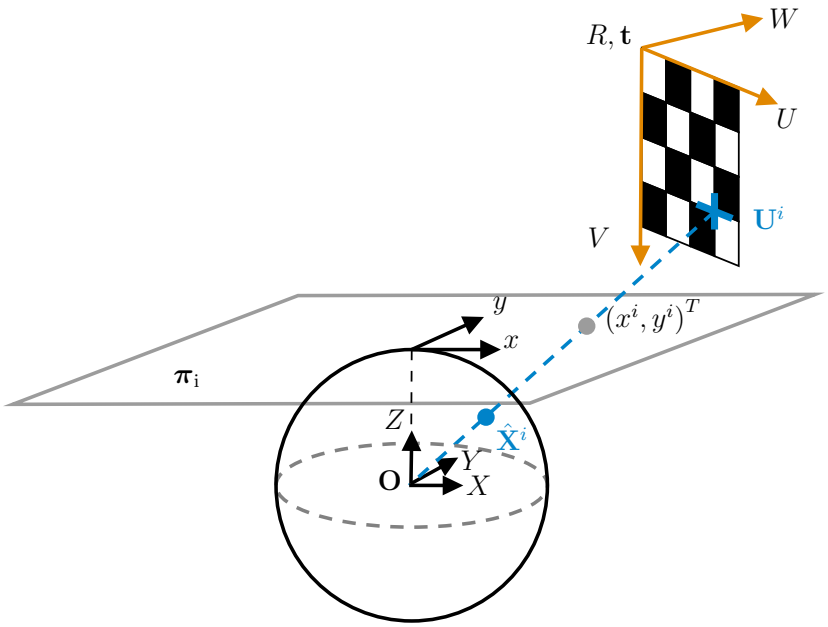

Fig. 4.8: Pose estimation from known planar objects.

cameras. Image points on unit sphere are projected onto an image plane $\boldsymbol{\pi}_\mathrm{i}$ placed at $Z = 1$

$$\begin{pmatrix} x^i \\ y^i \\ 1 \end{pmatrix} = \frac{1}{\hat{Z}^i} \begin{pmatrix} \hat{X}^i \\ \hat{Y}^i \\ \hat{Z}^i \end{pmatrix}$$

to be used as input for the EPnP-algorithm[56] presented in [186, 158]. This algorithm has proved to be a reliable rotation estimator, which requires at least three point correspondences. Rotation estimation is scale invariant and thus less prone to image noise in contrast to translation estimation, which is noticably disturbed, due to the projection onto image plane $\boldsymbol{\pi}_\mathrm{i}$. In order to overcome this circumstance translation is estimated in a separate step by taking advantage of the collinearity between $\hat{\mathbf{X}}^i$ and $\mathbf{U}^i$ from Eq. (4.1), page 61

$$\hat{\mathbf{X}}^i \times \left(R\mathbf{U}^i + \mathbf{t} \right) = \mathbf{0}\,. \tag{4.12}$$

Substituting $\mathring{\mathbf{U}}^i = R\mathbf{U}^i$ simplifies the collinearity constraint:

$$\hat{\mathbf{X}}^i \times \left(\mathring{\mathbf{U}}^i + \mathbf{t} \right) = \mathbf{0} \tag{4.13}$$

$$[\hat{\mathbf{X}}^i]_\times \mathbf{t} + \hat{\mathbf{X}}^i \times \mathring{\mathbf{U}}^i = \mathbf{0} \tag{4.14}$$

[56] https://github.com/cvlab-epfl/EPnP

to be brought into a DLT-form $B^i\mathbf{x} = \mathbf{0}$ with

$$\underbrace{\left[\ [\hat{\mathbf{X}}^i]_\times \ \middle|\ \hat{\mathbf{X}}^i \times \mathring{\mathbf{U}}^i\ \right]}_{B^i} \underbrace{\begin{pmatrix}\mathbf{t}\\ 1\end{pmatrix}}_{\mathbf{x}} = \mathbf{0}\,. \tag{4.15}$$

This is similar to the 3d-2d PnP principle already explained in Section 4.4.1, page 60, however in this case $\mathbf{t}$ is the only unknown. B^i is also rank-deficienct with $\operatorname{rank}(B^i) = 2$, which leads to a reduced number of matrix rows

$$B^i = \begin{bmatrix} 0 & -\hat{\mathbf{X}}^i_{(3)} & \hat{\mathbf{X}}^i_{(2)} & \hat{\mathbf{X}}^i_{(2)}\mathring{\mathbf{U}}^i_{(3)} - \hat{\mathbf{X}}^i_{(3)}\mathring{\mathbf{U}}^i_{(2)} \\ \hat{\mathbf{X}}^i_{(3)} & 0 & -\hat{\mathbf{X}}^i_{(1)} & \hat{\mathbf{X}}^i_{(3)}\mathring{\mathbf{U}}^i_{(1)} - \hat{\mathbf{X}}^i_{(1)}\mathring{\mathbf{U}}^i_{(3)} \end{bmatrix} \in \mathbb{R}^{2\times 4}\,. \tag{4.16}$$

All point correspondences - at least two are necessary to solve the system of linear equations - are stacked into B

$$B = \begin{bmatrix} 0 & -\hat{\mathbf{X}}^1_{(3)} & \hat{\mathbf{X}}^1_{(2)} & \hat{\mathbf{X}}^1_{(2)}\mathring{\mathbf{U}}^1_{(3)} - \hat{\mathbf{X}}^1_{(3)}\mathring{\mathbf{U}}^1_{(2)} \\ \hat{\mathbf{X}}^1_{(3)} & 0 & -\hat{\mathbf{X}}^1_{(1)} & \hat{\mathbf{X}}^1_{(3)}\mathring{\mathbf{U}}^1_{(1)} - \hat{\mathbf{X}}^1_{(1)}\mathring{\mathbf{U}}^1_{(3)} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & -\hat{\mathbf{X}}^p_{(3)} & \hat{\mathbf{X}}^p_{(2)} & \hat{\mathbf{X}}^p_{(2)}\mathring{\mathbf{U}}^p_{(3)} - \hat{\mathbf{X}}^p_{(3)}\mathring{\mathbf{U}}^p_{(2)} \\ \hat{\mathbf{X}}^p_{(3)} & 0 & -\hat{\mathbf{X}}^p_{(1)} & \hat{\mathbf{X}}^p_{(3)}\mathring{\mathbf{U}}^p_{(1)} - \hat{\mathbf{X}}^p_{(1)}\mathring{\mathbf{U}}^p_{(3)} \end{bmatrix} \in \mathbb{R}^{2p\times 4} \tag{4.17}$$

to be decomposed $B = U\Sigma V^T$ using an SVD. Again, normalizing each row of B before applying an SVD leads to more stable numberical solution. The last column $V_{(:,4)}$ of the right unitary matrix $V \in \mathbb{R}^{4\times 4}$ corresponds to the smalles singular value in the diagonal matrix $\Sigma \in \mathbb{R}^{4\times 4}$ and represents the solution for $\mathbf{x} = V_{(:,4)}$. In order to extract $\mathbf{t} = \mathbf{x}_{(1:3)}$, a normalization is required such that

$$\mathbf{t} = \frac{\mathbf{x}_{(1:3)}}{\mathbf{x}_{(4)}} \tag{4.18}$$

as originally stated in Eq. (4.15). By doing so the sign of $\mathbf{t}$ is not arbitrary and hence correctly obtained for the input rotation R. However, there exists a rotation ambiguty that also influcences the sign of the translation as explained in the following.

Resolve Ambiguity from 3D-2D Pose Estimation

Ambiguity is caused by the projection onto image plane $\boldsymbol{\pi}_\mathrm{i}$, which is independent of the sign of $\hat{\mathbf{X}}^i$. This means a point on the opposite hemisphere yields the same image coordinates

$$\begin{pmatrix} x^i \\ y^i \\ 1 \end{pmatrix} = \frac{1}{-\hat{Z}^i} \begin{pmatrix} -\hat{X}^i \\ -\hat{Y}^i \\ -\hat{Z}^i \end{pmatrix},$$

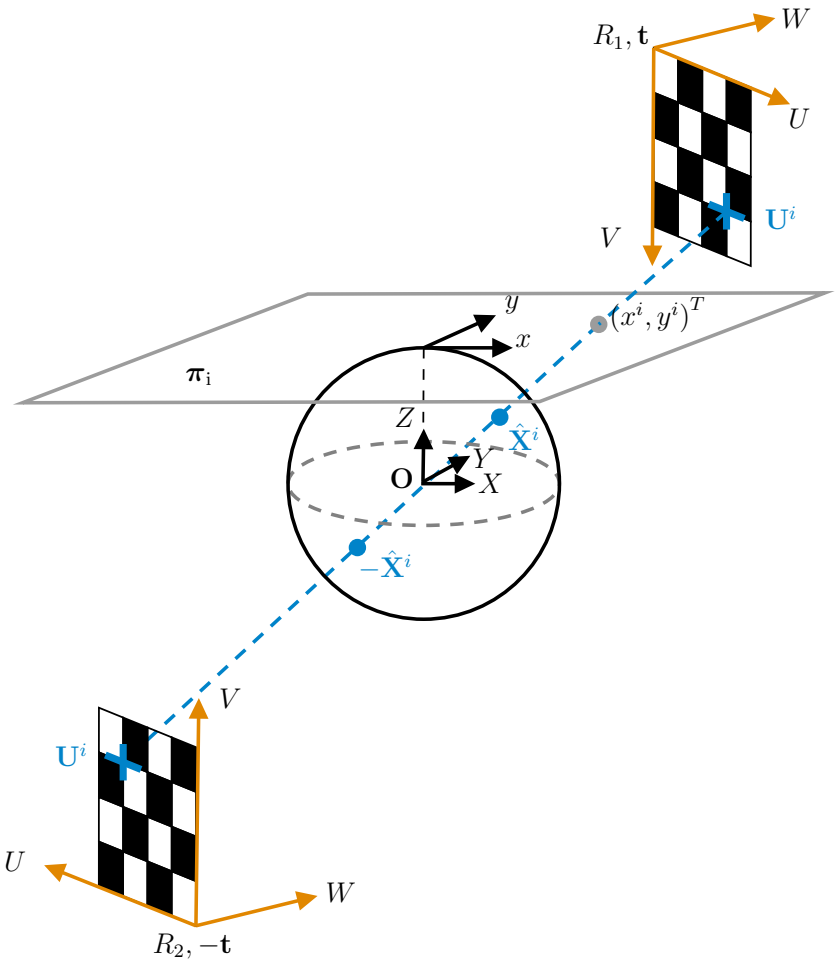

Fig. 4.9: Pose estimation from planar objects leads to two solutions caused by the projection onto image plane $\boldsymbol{\pi}_\mathrm{i}$. Thus point $\hat{\mathbf{X}}^i$ and its counterpart $-\hat{\mathbf{X}}^i$ yield the same projection $(x^i, y^i)^T$.

as Fig. 4.9 illustrates. The EPnP-algorithm uses the relations $(x^i, y^i)^T \leftrightarrow (U^i, V^i, W^i)^T$ as input correspondences in order to obtain the rotation and hence has no direct relation to the real image coordinates on unit sphere $(\hat{X}^i, \hat{Y}^i, \hat{Z}^i)^T$. There are two possible solutions $[R_1, \mathbf{t}]$ and $[R_2, -\mathbf{t}]$, where R_1 is directly obtained via EPnP. $[R_2, -\mathbf{t}]$ represents the counterpart that correspondes to $-\hat{\mathbf{X}}^i$ with

$$R_2 = R_1 \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.19}$$

which is flipped along U- and V-direction. The correct transformation is choosen by validating cheirality (Section 6.4.1, page 97). Tests turned out that the projection error is a more reliable criterion under image noise than the viewing direction as it is used in Section 4.4.1, page 60 for cheirality validation. Furthermore it is recommended to compare the projection errors af all point correspondences instead of taking only the first point pair or an arbitrary choosen one. The sum of squared projection errors

corresponding to $[R_1, \mathbf{t}]$ is given by

$$\epsilon_1 = \sum_{i=1}^{p} \left\| \hat{\mathbf{X}}^i - \hat{\mathbf{Y}}^i \right\|^2 \tag{4.20}$$

with

$$\hat{\mathbf{Y}}^i = \frac{R_1 \mathbf{U}^i + \mathbf{t}}{\left\| R_1 \mathbf{U}^i + \mathbf{t} \right\|} . \tag{4.21}$$

Analogous ϵ_2 relating to $[R_1, -\mathbf{t}]$ is simply obtained by

$$\epsilon_2 = \sum_{i=1}^{p} \left\| \hat{\mathbf{X}}^i + \hat{\mathbf{Y}}^i \right\|^2 \tag{4.22}$$

and does not need a recalculation of $\hat{\mathbf{Y}}^i$. Finally, the correct transformation is resolved by:

$$[R, \mathbf{t}] = \begin{cases} [R_1, \mathbf{t}], & \text{if } \epsilon_1 < \epsilon_2 \\ [R_2, -\mathbf{t}], & \text{otherwise} . \end{cases}$$

4.4.3 Pose Optimization

Pose optimization is based on minimizing the Euclidean distance between measured $\hat{\mathbf{X}}^i_{kl}$ and projected target points $\mathbf{Y}^i_{kl}$ as recommended in [147]. It is suitable for 3d-2d as well as for 2d-2d point correspondences and refines inital pose estimates from Sections 4.4.1 and 4.4.2, pages 60 and 62. Target points $\mathbf{U}^i$ are transferred from target l (which is the world frame $\mathcal{W}$ in this case) into camera frame $\mathcal{C}_k$

$$\mathbf{Y}^i_{kl} = R_{kl} \mathbf{U}^i + \mathbf{t}_{kl}$$

and then projected onto the camera sphere $\mathcal{S}_k$

$$\hat{\mathbf{Y}}^i_{kl} = \frac{\mathbf{Y}^i_{kl}}{\| \mathbf{Y}^i_{kl} \|} .$$

The optimization function is given by

$$\underset{R_{kl}, \mathbf{t}_{kl}}{\operatorname{argmin}} \sum_{i=1}^{p} \left\| \hat{\mathbf{X}}^i_{kl} - \frac{R_{kl} \mathbf{U}^i + \mathbf{t}_{kl}}{\| R_{kl} \mathbf{U}^i + \mathbf{t}_{kl} \|} \right\|^2 \tag{4.23}$$

and non-linear refined using LM. The following partial derivatives:

$$\frac{\partial \hat{\mathbf{Y}}^i_{kl}}{\partial \mathbf{Y}^i_{kl}} = \frac{k - \hat{\mathbf{Y}}^i_{kl} \left(\hat{\mathbf{Y}}^i_{kl} \right)^T}{\| \mathbf{Y}^i_{kl} \|} \qquad \frac{\partial \mathbf{Y}^i_{kl}}{\partial R_{kl}} = \left(\mathbf{U}^i \otimes k \right)^T \qquad \frac{\partial \mathbf{Y}^i_{kl}}{\partial \mathbf{t}_{kl}} = 1$$

form the corresponding Jacobian matrix

$$J_{kl} = \begin{bmatrix} \frac{\partial \hat{\mathbf{Y}}^1_{kl}}{\partial \mathbf{Y}^1_{kl}} \frac{\partial \mathbf{Y}^1_{kl}}{\partial R_{kl}} & \frac{\partial \hat{\mathbf{Y}}^1_{kl}}{\partial \mathbf{Y}^1_{kl}} \frac{\partial \mathbf{Y}^1_{kl}}{\partial \mathbf{t}_{kl}} \\ \vdots & \vdots \\ \frac{\partial \hat{\mathbf{Y}}^p_{kl}}{\partial \mathbf{Y}^p_{kl}} \frac{\partial \mathbf{Y}^p_{kl}}{\partial R_{kl}} & \frac{\partial \hat{\mathbf{Y}}^p_{kl}}{\partial \mathbf{Y}^p_{kl}} \frac{\partial \mathbf{Y}^p_{kl}}{\partial \mathbf{t}_{kl}} \end{bmatrix} . \tag{4.24}$$

4.4.4 Uncertainty Estimation

Based on the assumption from Appendix A.7, page 180, the covariance can be approximated by

$$S_{kl} = (J_{kl}^T J_{kl})^{-1} \frac{1}{p-1} \sum_{i=1}^{p} \left\| \hat{\mathbf{X}}^i_{kl} - \hat{\mathbf{Y}}^i_{kl} \right\|^2 . \tag{4.25}$$

The calculation of $(J_{kl}^T J_{kl})^{-1}$ is not defined if J_{kl} is rank-deficient, which requires to calculate the pseudo-inverse. J_{kl} becomes rank-deficient if the three dimensional rotation is overparameterized as it is the case for a rotation matrix (nine parameters) or a quaternion (four parameters) representation. In order to prevent this circumstance it is recommended to use a rotation representation such as Euler angles or Rodrigues vector. Later the covariance matrix can be transferred into rotation matrix or quaternion representation as proposed in [17]. For extrinsic calibration only diagonal values of S_{kl} were used.

The information matrix $\Omega_{kl} = S_{kl}^{-1}$ is the inverse of the covariance matrix [133, 128]. It is another representation describing the uncertainties, where larger values correspond to higher confidence. The information matrix as uncertainty representation is often used by pose graph solvers as Section 8.1, page 139 explains.

4.4.5 Pose Graph Representation

The proposed extrinsic calibration routine uses PGO (Chapter 8, page 139) in order to obtain camera $[R^{\mathrm{c}}_k, \mathbf{t}^{\mathrm{c}}_k]$, $k = 1, \ldots, m$ and target poses $[R^{\mathrm{t}}_l, \mathbf{t}^{\mathrm{t}}_l]$, $l = 1, \ldots, n$, which both represent graph nodes. The relative transformation $[R_{kl}, \mathbf{t}_{kl}]$ links the l^{th} target node to the k^{th} camera node and denotes an edge in $\mathcal{E}$.

In a frist step the algorithm selects a subset of target nodes, which connects all camera nodes. Each target node connects two camera nodes as illustrated in Fig. 4.10**a**. In case of multiple solutions the algorithm chooses the target node with the smallest cumulative projection error in both camera nodes. In a second step (Fig. 4.10**b**), nodes are renumbered such that node numbers alternate between cameras and targets in ascending order. This is necessary since most pose graph solvers require a sequence (trajectory) of input data and hence need an ordered structure with an ascending node order. In a third step, the remaining target nodes are added to the pose graph (Fig. 4.10**c**) and are renumbered. They add additional links between

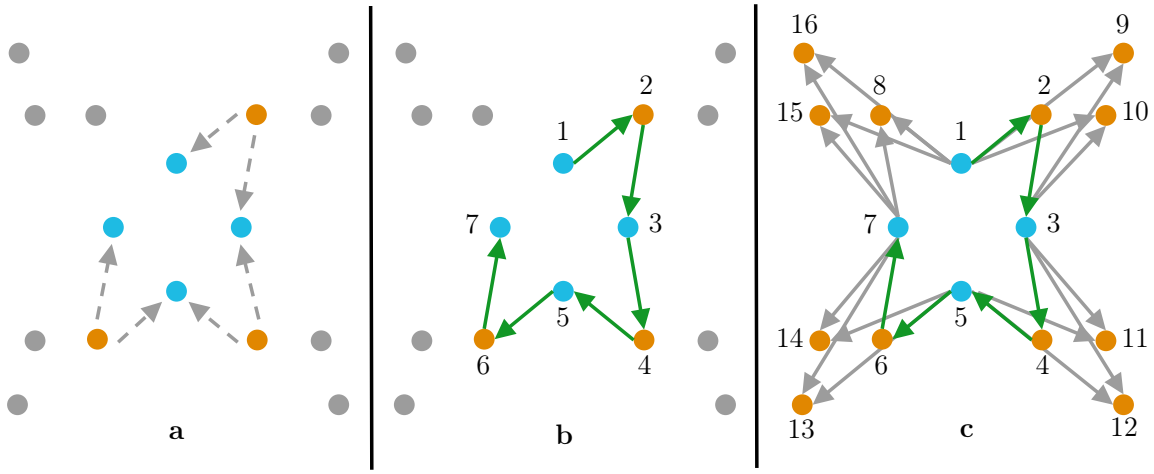

Fig. 4.10: Pose graph extraction from given relative transformations (⇢). **a)** Finding a subset of target nodes (●), which connects all camera nodes (●). **b)** Renumbering camera and target nodes and inverting certain relative transformations to form a directed sub-pose graph (→) with ascending node order $(1-7)$. **c)** Remaining target nodes are renumbered $(8-16)$ and added to the pose graph (→).

camera nodes. The pose graph solver is only able to perform optimization if either multiple target nodes link between camera nodes or a loop closure exists between the first and the last camera (as shown in Fig. 4.6, page 60, considering the four highlighted cameras only).
Renumbering is saved as list in order to restore the original node indication. A pose graph solver (Section 8.2, page 140) solves for the unknown camera and target nodes. The solver usually aligns the global coordinate system with the first camera node ($[R_1^c, \mathbf{t}_1^c] = [I|\mathbf{0}]$).

4.4.6 Bundle Adjustment

BA further improves extrinsics from PGO. It compensates alignment errors, which may be caused by erroneous uncertainty estimates from relative pose optimization (Section 4.4.3, page 66). Extrinsic optimization is achieved by minimizing the sum of squared projection errors

$$\underset{\tilde{R}_k^{\mathrm{c}},\tilde{\mathbf{t}}_k^{\mathrm{c}}\tilde{R}_l^{\mathrm{t}},\tilde{\mathbf{t}}_l^{\mathrm{t}}}{\operatorname{argmin}} \sum_{kl\in\mathcal{E}} \sum_{i=1}^{p} \left\| \hat{\mathbf{X}}_{kl}^{i} - \hat{\mathbf{Y}}_{kl}^{i} \right\|^2 , \tag{4.26}$$

which leads to a sparse BA problem that is solved by LM. Rotation optimization is performed by using a Rodrigues vector representation. The projection function follows an adopted transformation sequence from Section 4.4.3, page 66 with:

$$\begin{aligned} \mathbf{Y}_{kl}^{i} &= \tilde{R}_k^{\mathrm{c}} R_l^{\mathrm{t}} \mathbf{U}^{i} + \tilde{R}_k^{\mathrm{c}} \mathbf{t}_l^{\mathrm{t}} + \tilde{\mathbf{t}}_k^{\mathrm{c}} \\ \hat{\mathbf{Y}}_{kl}^{i} &= \frac{\mathbf{Y}_{kl}^{i}}{\|\mathbf{Y}_{kl}^{i}\|} , \end{aligned}$$

where $\left[\tilde{R}_k^{\mathrm{c}}, \tilde{\mathbf{t}}_k^{\mathrm{c}}\right] = \left[(R_k^{\mathrm{c}})^T, -(R_k^{\mathrm{c}})^T \mathbf{t}_k^{\mathrm{c}}\right]$ denotes the inverse of the camera pose, making jacobian calculation simpler. For the sake of better readability the Jacobian J is split into camera Jacobian J^{c} and target Jacobian J^{t}:

$$J_{kl}^{\mathrm{c}} = \begin{bmatrix} \frac{\partial \hat{\mathbf{Y}}_{kl}^{1}}{\partial \mathbf{Y}_{kl}^{1}} \frac{\partial \mathbf{Y}_{kl}^{1}}{\partial \tilde{R}_k^{\mathrm{c}}} & \frac{\partial \hat{\mathbf{Y}}_{kl}^{1}}{\partial \mathbf{Y}_{kl}^{1}} \frac{\partial \mathbf{Y}_{kl}^{1}}{\partial \tilde{\mathbf{t}}_k^{\mathrm{c}}} \\ \vdots & \vdots \\ \frac{\partial \hat{\mathbf{Y}}_{kl}^{p}}{\partial \mathbf{Y}_{kl}^{p}} \frac{\partial \mathbf{Y}_{kl}^{p}}{\partial \tilde{R}_k^{\mathrm{c}}} & \frac{\partial \hat{\mathbf{Y}}_{kl}^{p}}{\partial \mathbf{Y}_{kl}^{p}} \frac{\partial \mathbf{Y}_{kl}^{p}}{\partial \tilde{\mathbf{t}}_k^{\mathrm{c}}} \end{bmatrix} \qquad J_{kl}^{\mathrm{t}} = \begin{bmatrix} \frac{\partial \hat{\mathbf{Y}}_{kl}^{1}}{\partial \mathbf{Y}_{kl}^{1}} \frac{\partial \mathbf{Y}_{kl}^{1}}{\partial R_l^{\mathrm{t}}} & \frac{\partial \hat{\mathbf{Y}}_{kl}^{1}}{\partial \mathbf{Y}_{kl}^{1}} \frac{\partial \mathbf{Y}_{kl}^{1}}{\partial \mathbf{t}_l^{\mathrm{t}}} \\ \vdots & \vdots \\ \frac{\partial \hat{\mathbf{Y}}_{kl}^{p}}{\partial \mathbf{Y}_{kl}^{p}} \frac{\partial \mathbf{Y}_{kl}^{p}}{\partial \tilde{R}_k^{\mathrm{c}}} & \frac{\partial \hat{\mathbf{Y}}_{kl}^{p}}{\partial \mathbf{Y}_{kl}^{p}} \frac{\partial \mathbf{Y}_{kl}^{p}}{\partial \tilde{\mathbf{t}}_k^{\mathrm{c}}} \end{bmatrix} ,$$

which are formed by the following derivatives:

$$\frac{\partial \hat{\mathbf{Y}}_{kl}^{i}}{\partial \mathbf{Y}_{kl}^{i}} = \frac{k - \hat{\mathbf{Y}}_{kl}^{i} \left(\hat{\mathbf{Y}}_{kl}^{i}\right)^T}{\|\mathbf{Y}_{kl}^{i}\|} \qquad \frac{\partial \mathbf{Y}_{kl}^{i}}{\partial \tilde{R}_k^{\mathrm{c}}} = \left(\left(R_l^{\mathrm{t}} \mathbf{U}^{i} + \mathbf{t}_l^{\mathrm{t}}\right) \otimes k \right)^T \qquad \frac{\partial \mathbf{Y}_{kl}^{i}}{\partial R_l^{\mathrm{t}}} = \left(\mathbf{U}^{i}\right)^T \otimes \tilde{R}_k^{\mathrm{c}}$$

$$\frac{\partial \mathbf{Y}_{kl}^{i}}{\partial \tilde{\mathbf{t}}_k^{\mathrm{c}}} = 1 \qquad \frac{\partial \mathbf{Y}_{kl}^{i}}{\partial \mathbf{t}_l^{\mathrm{t}}} = \tilde{R}_k^{\mathrm{c}}$$

using chain rule. Stacking all J^{c}_{kl} and J^{t}_{kl} in J using the proposed scheme such that

$$J = \left[\begin{array}{ccc|ccc} J^{\mathrm{c}}_{11} & & & J^{\mathrm{t}}_{11} & & \\ \vdots & & & & \ddots & \\ J^{\mathrm{c}}_{1n} & & & & & J^{\mathrm{t}}_{1n} \\ & \ddots & & \vdots & & \\ & \ddots & & & \vdots & \\ & \ddots & & & & \vdots \\ & & J^{\mathrm{c}}_{m1} & J^{\mathrm{t}}_{m1} & & \\ & & \vdots & & \ddots & \\ & & J^{\mathrm{c}}_{mn} & & & J^{\mathrm{t}}_{mn} \end{array}\right] \tag{4.27}$$

yields a sparse Jacobian, whose non-zero elements depend on the visibility between cameras and targets. Zeros rows are rejected from the Jacobian structure, as shown at hand of the example case illustrated in Fig. 4.11**c**. The corresponding covariance matrix is calculated following the way as described in Section 4.4.4, page 67. Uncertainties of both, camera and target poses are extracted from diagonal entries of the derived covariance matrix.

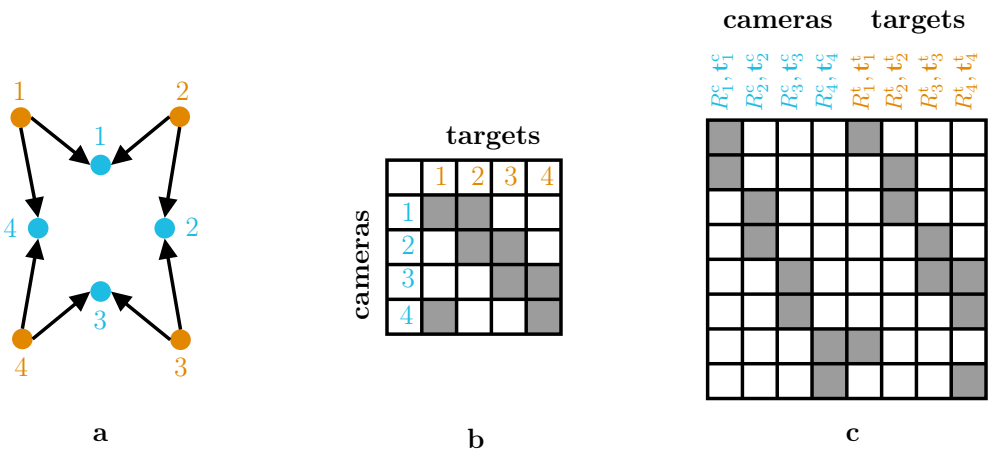

Fig. 4.11: BA to refine extrinsic camera calibration. **a)** Recap of the calibration scenario consisting of four partly overlapping cameras connected by four calibration target poses, where each target is observed by two cameras. **b)** Visibility map indicating the set $\mathcal{E}$ of available target-camera constraints. **c)** Corresponding sparse structure of the Jacobian used in BA.

4.4.7 Selected Examples

This section presents two challenging camera setups that are extrinsically calibrated with the proposed pipeline.

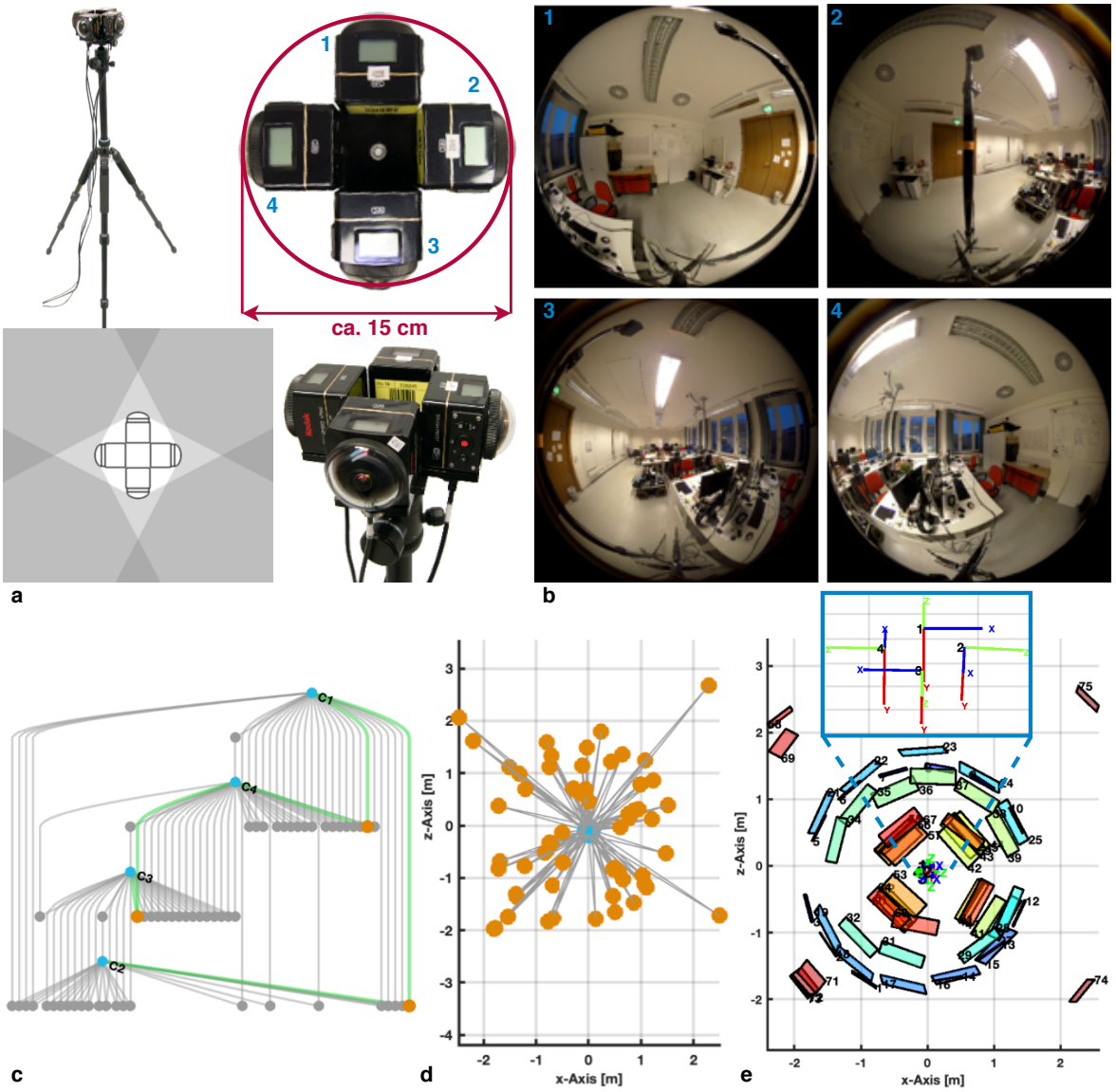

Fig. 4.12: Extrinsic calibration of a camera setup consisting of four *Kodak SP360 4K* cameras. The following is shown: **a)** final camera setup with corresponding FoVs, **b)** sample camera images, **c)** a graph view visualizes the connections between cameras and targets (▬) and highlights the sub-pose graph (▬), which connects the four cameras nodes (●, C1-C4) using three target nodes (●), **d)** optimized pose graph with all target nodes (top view), **e)** camera and target alignment from BA-refinement (top view).

Four Camera Setup

A camera setup was built, which consists of four *Kodak SP360 4K* cameras. This action camera model is equipped with an ultra-fisheye lens covering a FoV up to 220° (depending on the camera's capture mode) as shown in Fig. 4.4, page 57. In [258], the author calibrates a slimmed version of this camera setup (three cameras only) with Li's toolbox [161], which however yield unsatisfying results.

The system was designed to capture a full omnidirectional view (360° × 180°) and to recover depth information from overlapping FoVs. Even objects being close the camera setup are captured by up to three cameras as illustrated in Fig. 4.12**a**, which enables

improved depth reconstruction. This setup is suitable to be used in narrow environments, e.g. for visual shaft inspection.
Each of the four cameras is intrinsically calibrated with the proposed calibration toolbox from Section 4.3, page 54. The obtained extrinsic results are graphically illustrated in Fig. 4.12**e**.

Eight Camera Setup

The eight camera setup represents the final prototype, which was developed during the *iDeepMon* project to be used as an optical scanning device for vertical shaft inspection purposes. The system consists of eight *PCO edge* 5.5[57] cameras in conjunction with a *Voigtländer Nokton* 10.5*mm* *F*0.95[58] mounted in an elliptical shaped arrangement in order to cover a monoscopic 360° horizontal FoV.

The system was intrinsically and extrinsically calibrated by *i3main - Institut für raumbezogene Informations- und Messtechnik, University of Applied Science*[59] using a separate calibration room as well as the software tools *Triptop v6.2.0*[60], *Agisoft PhotoScan v1.4.3* and *Jag3D v20180714*[61]. The provided data are used as reference calibration, which is shown in Fig. 4.13**f, g**. As can be seen both calibration do not coincide. Especially cameras on the lefthand side show clear alignment differences. The reference calibration was accomplished by an external contractor using specialized equipment and laboratory space during several hours of calibration. Whereas the presented calibration was performed during a shaft inspection test on site, which restricted the image aquisition process in time and space. Images were taken within a short time span of a few minutes only, resulting in an insufficient number of calibration images on the lefthand side. As Fig. 4.13**b, c** clearly show, there is only one target pose that connects two neighboring cameras. This circumstance possibly causes the deviation to the reference calibration. It can be expected that more calibration images would yield better extrinsic results similiar to the ones from the reference calibration. However in order to verify this statement an additional calibration needs to be done under proper conditions in future work.

The presented extrinsic calibration routine obtained promising extrinsic results. Compared to a specialized calibration, it is a convenient method to calibrate the system on site within a shorter time frame and using one software toolbox only. On site calibration is especially necessary if the camera system is modified during operation process.

[57] https://www.pco.de/fileadmin/user_upload/pco-product_sheets/pco.edge_55_data_sheet.pdf
[58] https://www.voigtlaender.de/objektive/mft/105-mm-f-095-nokton/
[59] https://www.hs-mainz.de/forschung/forschung-transfer/institute-amtliche-pruefstelle/i3mainz/
[60] https://www.gom.com/en/products/3d-scanning/tritop
[61] https://github.com/applied-geodesy/jag3d
[62] image curtesy *DMT*

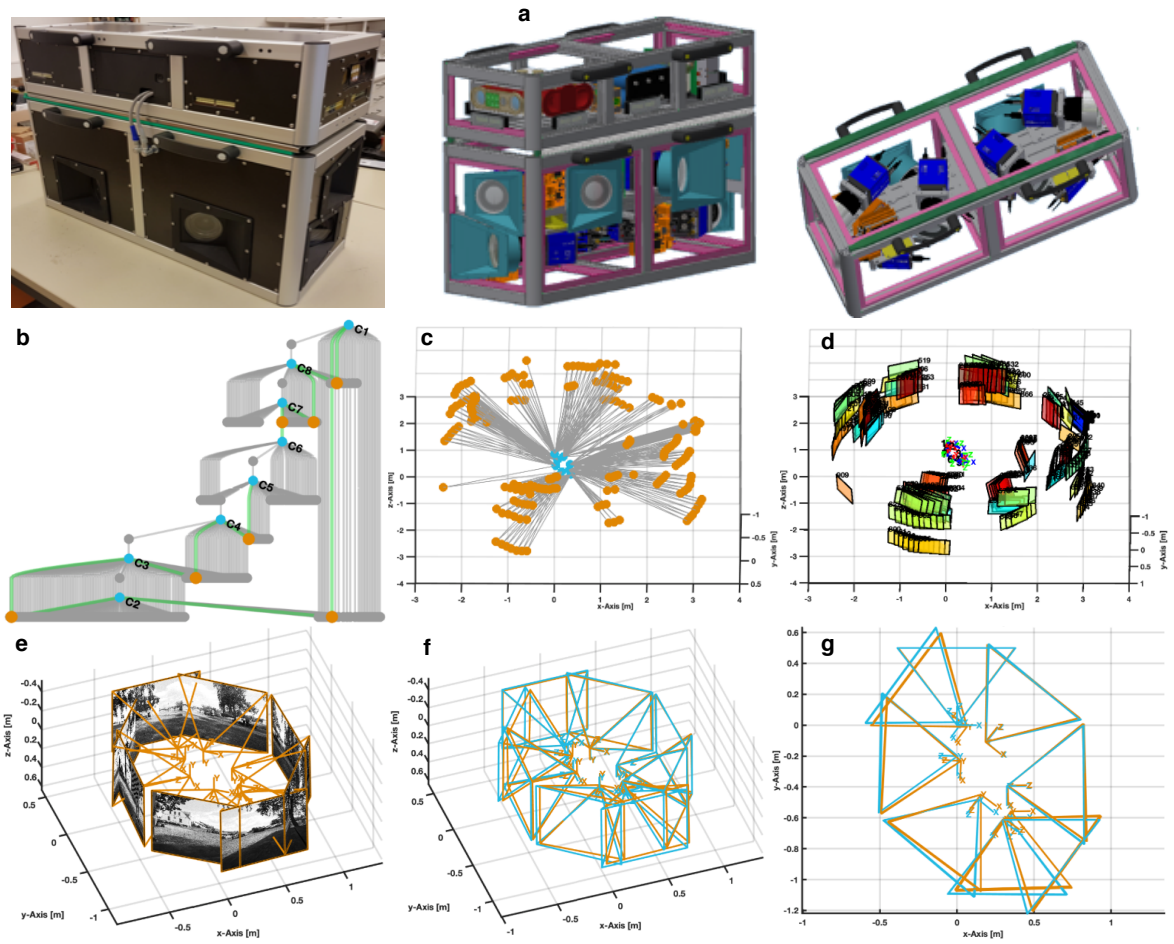

Fig. 4.13: Extrinsic calibration of an eight camera system used in the *iDeepMon* project. The following is shown: **a)** final shaft scanning prototype consisting of eight cameras[62], **b)** a graph view visualizes the connections between cameras and targets (▬) and highlights the sub-pose graph (▬), which connects the eight camera nodes (●, C1-C8) using seven target nodes (●), **c)** optimized pose graph with all target nodes, **d)** camera and target alignment from BA-refinement, **e)** camera extrinsics with corresponding images, **f, g)** obtained camera extrinsics (▬) overlayed with reference calibration (▬).

Additional calibration examples are presented in Appendix A.1, page 170.

Brief Chapter Summary

The chapter explained the proposed calibration pipeline consisting of target detection, intrinsic and extrinsic calibration. Target detection is a compilation of existing checkerboard detection algorithms, which do not requiere any prior knowledge about image distortion. Intrinsic calibration is based on popular calibration toolboxes integrating UCM as well as PCM and provides P2S-maps as calibration output. Extrinsic calibration incorporates P2S-maps to work with all types of central camera systems. PnP techniques dedicated to spherical cameras were explained to obtain target-camera transformation estimates from 3d-2d and 2d-2d point correspondences. Based on these estimates PGO solves for extrinsics, which can be further refined via BA. This workflow represents as flexible extrinsic calibration routine for multi-camera setups with over-

lapping FoVs, that is not limited to a certain number of cameras and achieves similar results compared to exsting implementations.

5 Full Omnidirectional Image Projections

Brief Chapter Overview

The generation as well as the conversion of full omnidirectional projections are in focus of this chapter. The generation is considered from two perspectives, namely the generation of source data in a standard projection format with large distortion and the generation of target projections with reduced distortion effects, which are suitable for further image processing. Section 5.1 describes the generation of an omnidirectional projection from overlapping input images using stitching techniques, which are implemented in modern 360° action cameras. Section 5.2 shows a selection of different world map projections with reduced image distortion effects as they are used in cartography. Section 5.3 explains the idea of generating custom P2S-maps from world map projections using a geographic library. It is followed by Section 5.4 giving insight into the conversion between image projections based on their corresponding P2S-maps using an interpolation scheme with weighted lookup values as comprehensively explained in Section 5.4.1. The conversion process is kept as general as possible and does not need any additional knowledge about the underlying projection functions. It can be applied to any projection described by a corresponding P2S-map and is used to convert stitched full omnidirectional images to a suitable world map projection in order to reduce distortion effects. Since this conversion depends on both projection geometries, the obtained conversion lookup between both projections can be saved for further image conversion as Section 5.4.2 describes. The chapter ends with Section 5.4.3 presenting an application example of the proposed conversion workflow.

5.1 Panoramic Image Stitching

Since full omnidirectional stitching cameras play an important role for underground image acquisition as already clarified in Section 1.1.3, page 4, this section describes the workflow how these cameras work. A stitched omnidirectional image that covers a 360° × 180° FoV represents an ideal central camera due to the circumstance, that the coverage of an entire sphere is technically impossible with one lens only. Image stichting (sometime also referred as image mosaicing) is a common technique to generate panoramas up to gigapixel size [144] from a collection of single overlapping (mostly directional) images. This is achieved by rotating a camera around the nodal

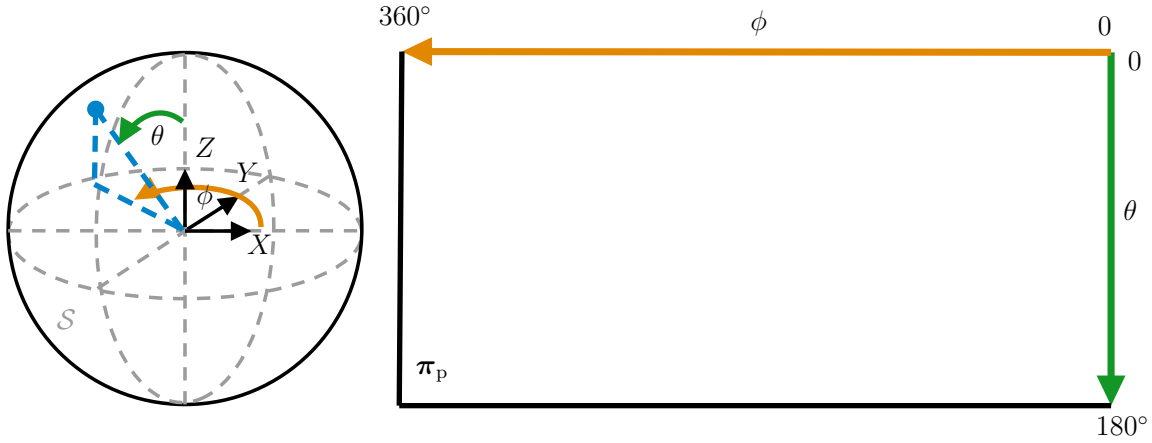

Fig. 5.1: Mapping between unit sphere and equirectangular image format.

point (optical camera center) using a tripod thus the captured images share a common viewpoint. They are combined using the following steps, which [77, 197] describe in detail.

Image registration obtains the transformation between overlapping images. There are different strategies like direct-based methods (using all available image information e.g. optical flow) or feature-based methods (using destinct or salient features). The latter ones are more robust against illumination variations and moving objects.

Warping maps each image onto a common manifold (surface) to align the input images. For flat mosaics a simple homography is used. Cylinderical or spherical surfaces are used for omnidirectional image composition, where each pixel's viewing direction of the composite image is mapped into the input images to obtain color information.

Blending is the most important step that reduces visible artifacts in order to generate seamless masaics. It compensates geometric and photometric errors, which lead to discontinuities between the images in the overlapping region. They are caused by registration inaccuracies and other factors like illumination variation, exposure difference or color imbalance. There are also different blending strategies such as transition smoothing, (a pixel in the transition region is replaced with a weighted average of the contributing images), optimal seam finding (finds an optimal seam in the overlapping region that produces least visible discrepancies) and hybrid blending (a combination of both forementioned strategies).

State-of-the-art stichting algorithms can even handle camera rotation and translation movements [281], which is suitable for handheld device like smartphones. This allows to capture panoramic images without the need of additional equipment like a tripot. However, camera translation leads to multiple viewpoints, which cause motion

parallax resulting into ghosting effects.

This motion parallax effect is especially desired to obtain omnidirectional stereoscopic images and videos (also known as stereoscopic 360°-videos or VR-videos). They became popular with increasing availability of improved VR-devices such as HMDs and provide a 3DoF (rotation around a fixed viewpoint) experience to the user. Images are captured at different viewpoints, which is achieved by mounting the camera on a rotating arm or by using overlapping synchronized cameras, which are distributed around a sphere in order to capture dynamic scenes. These camera setups are typical examples for non-central camera types. A stereoscopic panorama can then be created by stitching specific stripes from the input views [199, 204], however this may lead to distortion effects. In [89], the authors give a deep insight into different omnidirectional stereoscopic image acquisition techniques.

Fig. 5.1 depicts the equirectangular projection, which is widely used for full omnidirectional image and video data and has become a quasi standard. Cameras providing an equirectangular image are assumed to be calibrated, since each pixel's direction is known from the image specification as illustrated in Fig. 5.1. Stichting algorithms are aimed at compensating warping distortions, but even if a monoscopic or stereoscopic omnidirectional image looks appropriate to the user, it does not mean to be geometrically correct. This circumstance may cause camera alignment problems in image registration processes.

The equirectangular image format has strong distortions at the top and bottom, which is unsuitable for feature detection. Furthermore, this type of projection leads to increasing redundant pixel values towards the poles, which themselves become a horizontal line in the projection. In [65, 53] the authors focus on VR streaming and propose novel omnidirectional image representations based on cube map and rhombic dodecahedron map. The resulting projection maps are further split into tiles, which are rearranged in order to compose an image without black borders. However, none of these proposals have become a standard in industry yet. This work picks up the mentioned idea and converts equirectangular images to projections with less distortion effects (Section 5.4, page 81) as they are known in the field of carthography.

The *Ricoh Theta S*[63] is an omnidirectional action camera, which is used for underground capturing tests. The camera is spray water resistant and consists of two fisheye cameras with a small overlapping region. Both optical centers are very close to each other ($< 1cm$ [5]). This allows to warp both fisheye images onto a common unit sphere without causing noticable distortions [179]. Blending is only applied to the small overlapping area as shown in Fig. 5.2. The internal camera software stitches still

[63] https://theta360.com/en/about/theta/s.html

images, wheras a camera-specific computer software[64] or smartphone application[65] processes video data.

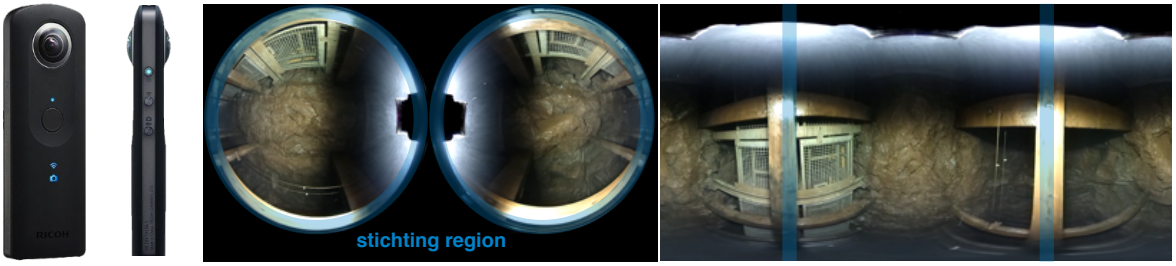

Fig. 5.2: Example illustration of an omnidirectional image stitching from fisheye cameras. *Ricoh Theta S* is shown with highlighted stitching regions in the raw image and corresponding stitched equirectangular image. The optical centers of both fisheye lenses are very close to each other, allowing to map both images onto a common unit sphere without causing strong distortions. Each fisheye lens provides a FoV $> 180°$, leading to a small stitching region (stripe).

5.2 World Map Projections

Map projections or world map projections describe the representation of the earth's surface on a flat map. The inverse of a map projection is an unprojection from pixel plane $\boldsymbol{\pi}_\mathrm{p}$ (map projection) to unit sphere $\mathcal{S}$ (earth's surface), similar to the unprojection from UCM (Section 3.2.1, page 35) and PCM (Section 3.2.2, page 39). The majority of map projections are designed to keep most distortions away from land. They project oceans or sparsely populated regions (e.g. arctic or antarctica) to stronger distorted map ares. Since these map projections make use of the earth's topography (distribution of water and land), they only work well under specific orientation. Fig. 5.3 depicts a selection of popular map projection from cartography. The Tissot's indicatrix illustrates the corresponding local distortions. Briefly explained, the Tissot's indicatrix characterizes local distortions by mapping a small circle from the unit sphere's surface to the map projection resulting in an ellipse. The size of the ellipse indicates areal distortions. Length and orientation of the semi-minor axes define angular distortions.

The idea is to convert a specific camera projection to an appropriate world map projection that equally distributes distortions, regardless of the image orientation. Making image representation independent of the specific camera projection and view orientation allows to compare/match images from different camera sources.

[64] https://theta360.com/de/about/application/pc.html
[65] https://theta360.com/de/about/application/basic.html

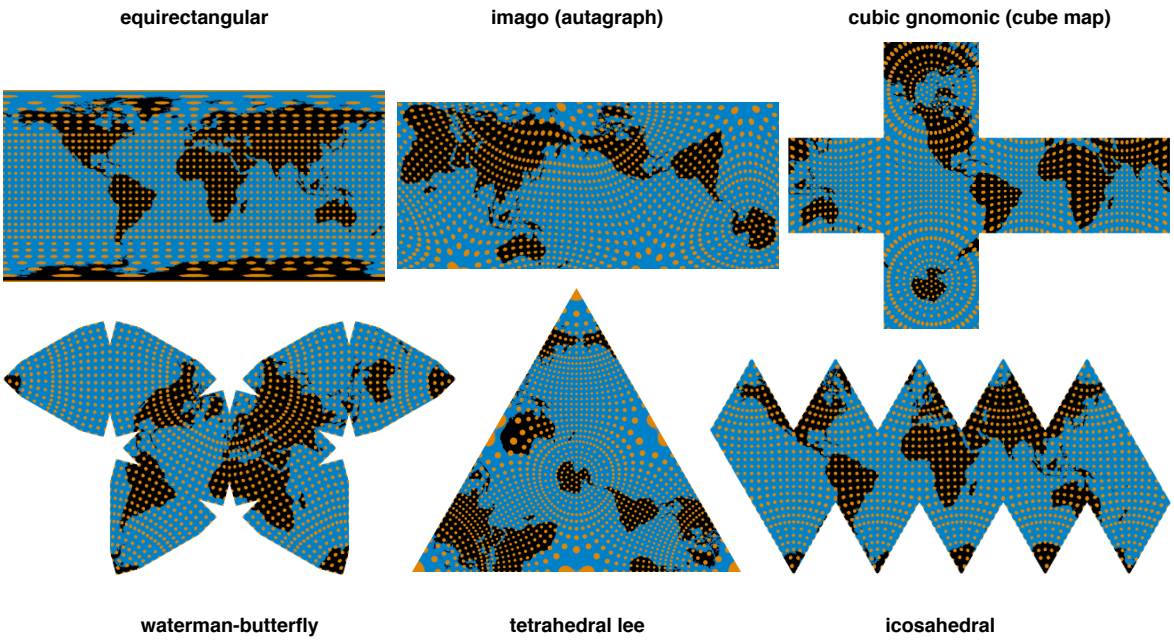

Fig. 5.3: Overview of selected world map projections with Tissot's indicatrix (•) visualizing the distortion distribution at specified locations. The **equirectangular** projection is one of the most widely known map projections in cartography and is also established as standard format for 360° images and videos. Having a closer look at the Tissot's indicatrix shows a strongly increasing distortion towards north and south pole. **Imago** and **tetrahedral lee** projections try to keep strong distortion away from land, which are less noticeable. The **cube map** projection has less destorted ares and distortion differences of **waterman-butterfly** and **icosahedral** projection are only noticable at higher zoom level leading to a more homogeneous distortion distribution.

5.3 World Map Projection Generator for P2S-Maps

There exist world map generation and conversion tools like *NASA's gprojector*[66], a *Java*-based program called *Map-Projections*[67] or a *Photoshop* plugin *Flexify 2*[68]. *NASA's gprojector* and *Map-Projections* are tested by the author. Each uses an equirectangular image as input. The user chooses from a list of target projections and adjusts rotation parameters of the sphere in order to obtain the desired projection view. However, these programs do not provide batch conversion for a serie of camera images.

In order to overcome this limitation, the basic idea was to use an equirectangular P2S-map as input image, to convert it into a desired target projection and to save the resulting image, which represents the desired P2S-map. The actual image conversion as batch processing would be performed using the workflow from Section 5.4, page 81. This strategy would allow to use forementioned map projection converters to generator new P2S-maps without prior knowledge about the world map projection itself. Unfortunately, until now the tested programs aren't able to output

[66] https://www.giss.nasa.gov/tools/gprojector/
[67] https://github.com/jkunimune15/Map-Projections
[68] http://www.flamingpear.com/flexify-output-modes.html

48-bit images (which is required to maintain the angular resolution as described in Section 3.3.1, page 43), such that this idea is postponed for future work.

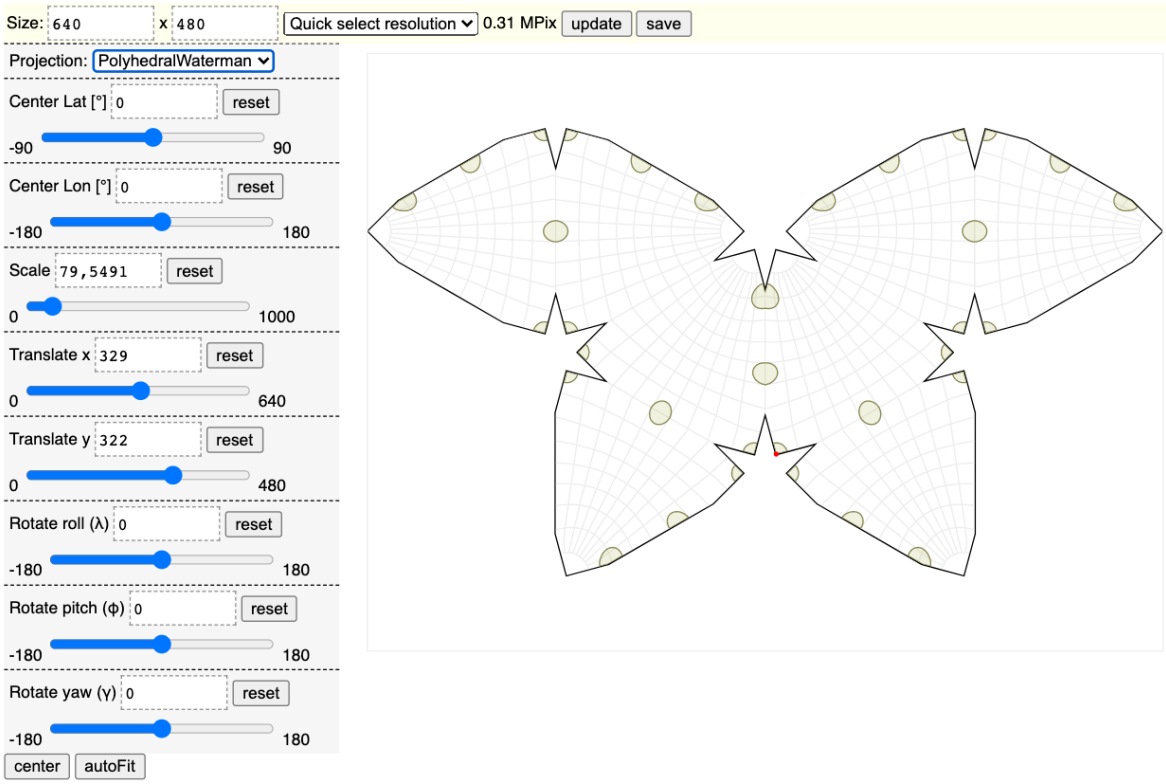

Fig. 5.4: Map projection generator GUI with parameter settings panel on the left and the corresponding world projection map with Tissot's indicatrices on the right.

In order to overcome the mentioned problem a custom map projection generator is developed. The generator is inspired by an online *projection explorer*[69], which is based on the *d3-geo*[70] library for *JavaScript*. Fig. 5.4 shows the generator's GUI in a webbrowser. The user selects a target world map projection from a drop-down list, specifies the image dimensions of the P2S-map and is able to change projection parameters in order to adjust the projection to the user's needs. The generator obtains the unprojection of the resulting world map projection by exploiting the library's functionality. For each pixel in the world map projection the library returns the corresponding spherical coordinates, which are collected, converted and saved as P2S-map.

[69] https://bl.ocks.org/d3indepth/f7ece0ab9a3df06a8cecd2c0e33e54ef
[70] https://github.com/d3/d3-geo

5.4 Conversion between Projections based on P2S-Maps

P2S-maps contain the unprojection mappings from pixel plane $\boldsymbol{\pi}_\mathrm{p}$ to unit sphere $\mathcal{S}$ as lookup instead of an analytic or geometric function. This circumstance prevents a direct conversion that maps pixels from projection A (${\boldsymbol{\pi}_\mathrm{p}}^A$) to projection B (${\boldsymbol{\pi}_\mathrm{p}}^B$) since both P2S-maps describe the mapping in the same direction. P2S-map B maps pixels from pixel plane ${\boldsymbol{\pi}_\mathrm{p}}^B$ onto unit sphere $\mathcal{S}$, however the mapping from unit sphere $\mathcal{S}$ to pixel plane ${\boldsymbol{\pi}_\mathrm{p}}^A$ is unknown. A simple inversion of P2S-map A obtains mappings from unit sphere $\mathcal{S}$ to pixel plane ${\boldsymbol{\pi}_\mathrm{p}}^A$ but would enforce lookup value interpolation (analogous to Section 3.3.2, page 46), which is not straightforward in this direction, due to the fact that neighboring points on unit sphere do not necessarily correspond to the same neighboring points on pixel plane ${\boldsymbol{\pi}_\mathrm{p}}^A$ as illustrated in Fig. 5.5. Instead of an interpolation, the inverse mapping could be generated from the known projection function A and could be also stored as lookup. However, this approach requires knowledge about the underlying model of projection A and also depends on the required angular resolution of P2S-map B.

The proposed conversion workflow is aimed to be as simple and general as possible in order to work with a wide range of P2S-maps from world map projections Section 5.2, page 78 and intrinsic camera calibrations Section 4.3, page 54. Additional information such as underlying projection and unprojection functions are not required.

5.4.1 Proposed Workflow

The follwing five step workflow is illustrated in Fig. 5.5 and describes the conversion from pojection A to projection B using their corresponding P2S-maps only.

Step 1

Each of the n pixels in projection A (${\boldsymbol{\pi}_\mathrm{p}}^A$) is mapped to a vertex on unit sphere $\mathcal{S}$ $(u_\mathrm{A}^j, v_\mathrm{A}^j)^T \rightarrow \hat{\mathbf{X}}_\mathrm{A}^j$, $j = 1, \ldots, n$ via corresponding P2S-map A.

Step 2

A triangle mesh is generated by connecting three neighboring vertices, which form a face. Since all vertices have the same distance to the center of the sphere, meshing guarantees to connect each vertex to the closest neighbor on the sphere surface. This step is especially mandatory when working with world map projections as they need cutting edges for unwrapping, e.g. equirectangular projection has one cutting edge near the date line. This circumstance leads to the forementioned problem, that neighboring vertices do not necessarily correspond to neighboring pixels, especially along cutting edges. The meshing process recovers the lost neighbor relations. Fig. 5.5 clarifies this probelm, $\hat{\mathbf{X}}_\mathrm{A}^k$, $\hat{\mathbf{X}}_\mathrm{A}^l$ and $\hat{\mathbf{X}}_\mathrm{A}^m$ are neighboring points on unit sphere and form a triangle.

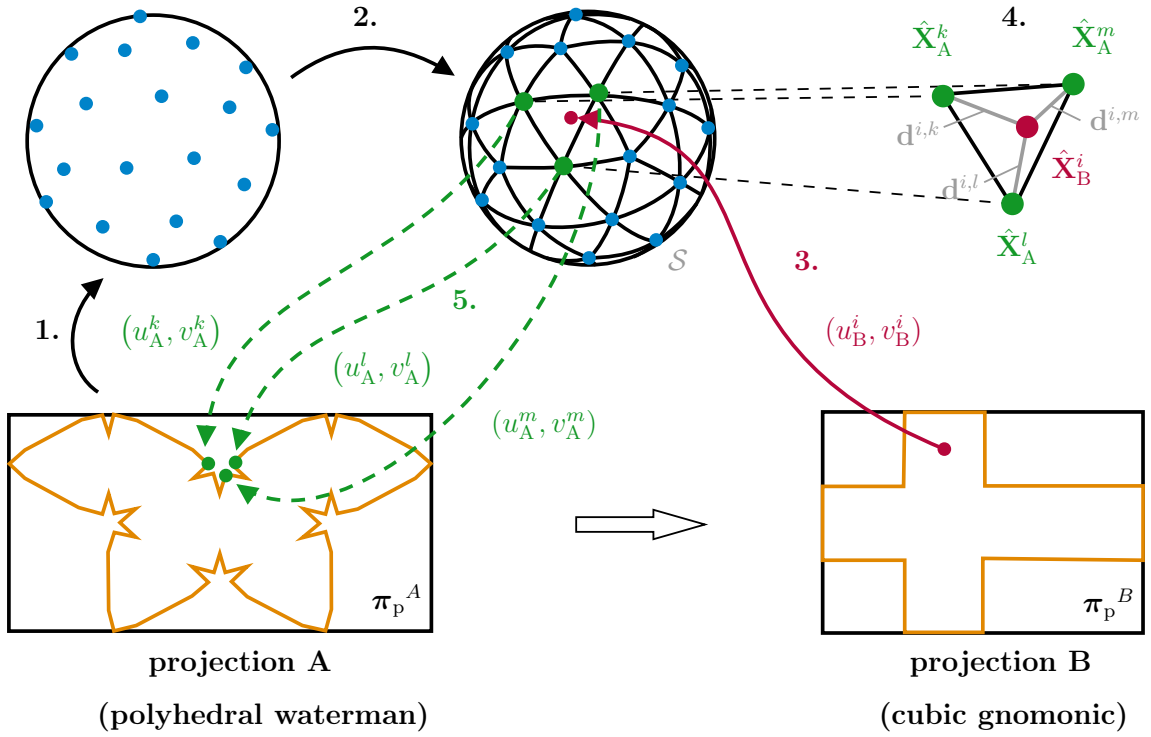

Fig. 5.5: Conversion from projection A to projection B in five steps.

On the contrary their correspondences $(u_A^k, v_A^k)^T$, $(u_A^l, v_A^l)^T$ and $(u_A^m, v_A^m)^T$ in projection A are located further apart. A direct interpolation between these pixel positions would lead to false results.

Step 3

Analogous to step 1, each of the m pixels in projection B ($\boldsymbol{\pi}_p{}^B$) is mapped to a vertex on unit sphere $\mathcal{S}$ $(u_B^i, v_B^i)^T \rightarrow \hat{\mathbf{X}}_B^i$, $i = 1, \ldots, m$ via corresponding P2S-map B.

Step 4

For each vertex $\hat{\mathbf{X}}_B^i$ the intersecting triangle face $\triangle(\hat{\mathbf{X}}_A^k, \hat{\mathbf{X}}_A^l, \hat{\mathbf{X}}_A^m)$ is obtained.

Step 5

Color value calculation is based on *inverse distance weighting*. The Euclidean distances $\mathbf{d}_{\mathbb{R}^3}(\cdot, \cdot)$ between $\hat{\mathbf{X}}_B^i$ and $\hat{\mathbf{X}}_A^k, \hat{\mathbf{X}}_A^l, \hat{\mathbf{X}}_A^m$ are calculated:

$$\begin{aligned} \mathbf{d}^{i,k} &= \left\| \hat{\mathbf{X}}_B^i - \hat{\mathbf{X}}_A^k \right\| \\ \mathbf{d}^{i,l} &= \left\| \hat{\mathbf{X}}_B^i - \hat{\mathbf{X}}_A^l \right\| \\ \mathbf{d}^{i,m} &= \left\| \hat{\mathbf{X}}_B^i - \hat{\mathbf{X}}_A^m \right\|, \end{aligned}$$

to further obtain the normalized weighting factors:

$$w^k = \frac{\frac{1}{\mathbf{d}^{i,k}}}{\frac{1}{\mathbf{d}^{i,k}} + \frac{1}{\mathbf{d}^{i,l}} + \frac{1}{\mathbf{d}^{i,m}}}$$
$$w^l = \frac{\frac{1}{\mathbf{d}^{i,l}}}{\frac{1}{\mathbf{d}^{i,k}} + \frac{1}{\mathbf{d}^{i,l}} + \frac{1}{\mathbf{d}^{i,m}}}$$
$$w^m = \frac{\frac{1}{\mathbf{d}^{i,m}}}{\frac{1}{\mathbf{d}^{i,k}} + \frac{1}{\mathbf{d}^{i,l}} + \frac{1}{\mathbf{d}^{i,m}}} ,$$

subjecting $w^k + w^l + w^m = 1$. Finally, the color values for each pixel $(u^i_{\mathrm{B}}, v^i_{\mathrm{B}})^T$ in projection B are obtained with

$$RGB(u^i_{\mathrm{B}}, v^i_{\mathrm{B}}) = RGB(u^k_{\mathrm{A}}, v^k_{\mathrm{A}})w^k + RGB(u^l_{\mathrm{A}}, v^l_{\mathrm{A}})w^l + RGB(u^m_{\mathrm{A}}, v^m_{\mathrm{A}})w^m .$$

Fig. 5.6 depicts a compilation of a scene in different world map projections. The original image is captured in equirectangular format and converted to other projections using the proposed five step workflow.

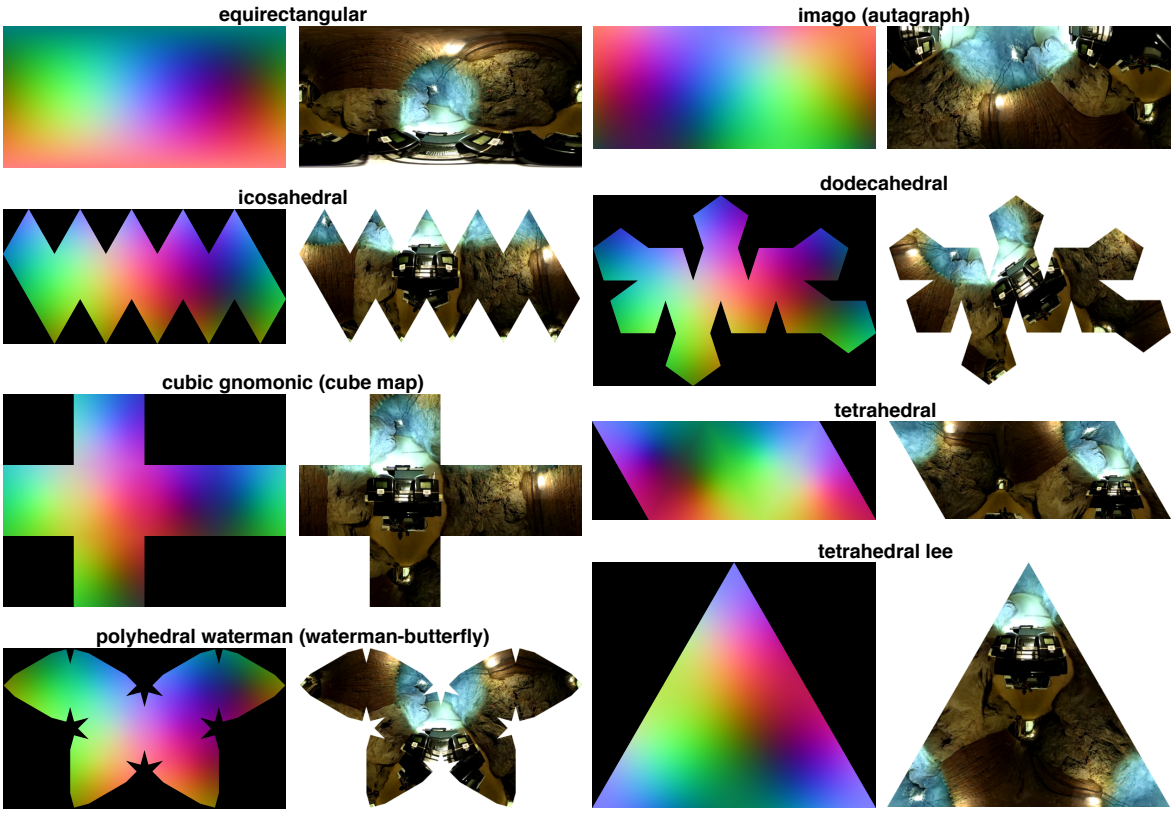

Fig. 5.6: Conversion of an equirectangular image from a *Ricoh Theta S* camera to different world map projections using corresponding P2S-maps.

5.4.2 Data Storage Format

Conversion from projection A to projection B is based on the lookup of pixel positions with additional interpolation using weighting factors. Each pixel $(u_{\mathrm{B}}^{i}, v_{\mathrm{B}}^{i})$ in projection B is assigned to three pixel positions $(u_{\mathrm{A}}^{k}, v_{\mathrm{A}}^{k}), (u_{\mathrm{A}}^{l}, v_{\mathrm{A}}^{l}), (u_{\mathrm{A}}^{m}, v_{\mathrm{A}}^{m})$ in projection A with corresponding weighting factors w^{k}, w^{l}, w^{m}. These lookup and interpolation values depend on geometry between both projections and hence must be obtained only once. However, in this case there are more lookup data correspondences as in

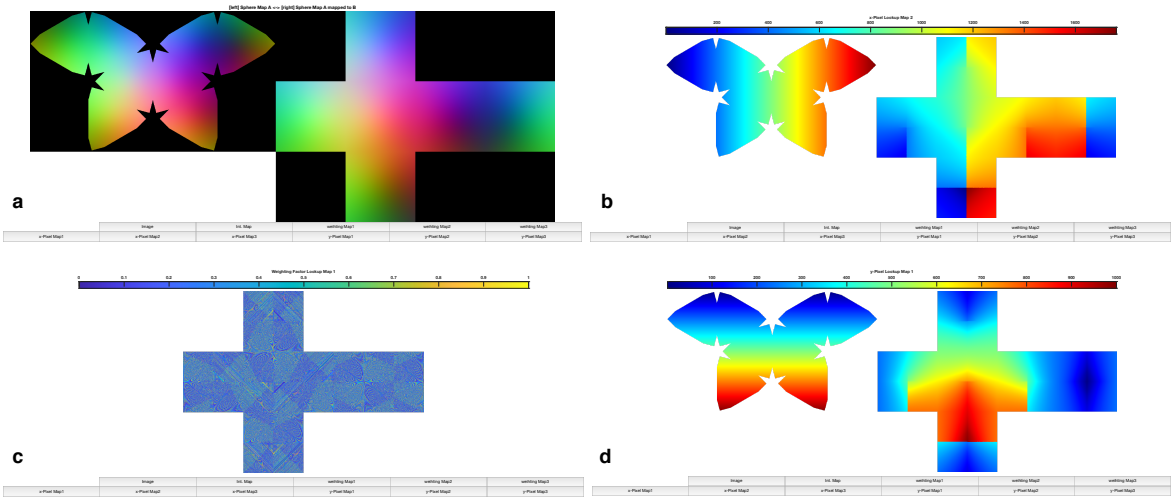

Fig. 5.7: Projection conversion GUI with an example conversion of a waterman-butterfly projection (A) to a cube map projection (B) showing the following: **a)** P2S-maps of source projection A and target projection B, **b)** the u_{A}^{k}-pixel lookup map where corresponding pixels between both projections have the same color, **c)** corresponding map of weighting factor w^{k}, **d)** v_{A}^{k}-pixel lookup map.

contrast to P2S-map. In order to store lookup data efficently, the HDF5[71] file format is chosen, its content can be visualized with the free Panoply[72] viewer. Each of the six lookup and three interpolation values is stored as matrix in shape of P2S-map B. Fig. 5.7 illustrates a P2S-map conversion example with additional information from the conversion toolbox.

Based on the generated mesh from **Step 2** (Section 5.4.1) and the known correspondences between pixels and mesh vertices an UV-map is generated. Mesh and and corresponding UV-map are additionally stored in a PLY file format. An image projection can then be handled as texture, which is mapped onto the unit sphere geometry in a 3d viewer as shown in Fig. 5.8.

[71] https://www.hdfgroup.org
[72] https://www.giss.nasa.gov/tools/panoply/download/

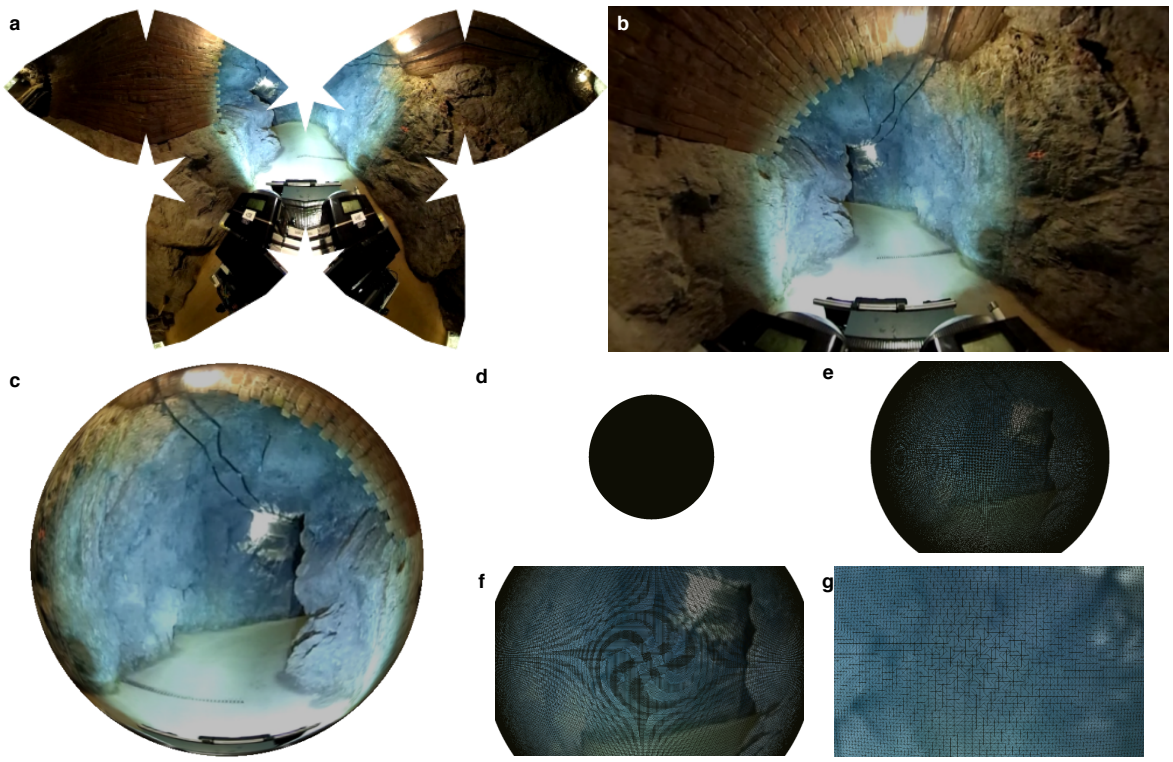

Fig. 5.8: Image projection as texture file mapped on a unit sphere mesh. The following is shown: **a)** waterman-butterfly projection as image texture file, **b)** image projection from the inside view of the sphere (same direction as image was captured), **c)** image projection mapped onto unit sphere geometry viewed from outside (opposite direction as image was captured), **d-g)** outside view of image projection with overlayed mesh at different zoom levels.

5.4.3 Real World Example

Within the *iDeepMon*[73] project an early protoype was built, that consist of a *Ricoh Theta S* (full omnidirectional stitching camera) and a lighting system to be mounted at the bottom of a cage for mineshaft inspection purposes as published in [15]. The proposed P2S-map conversion was successfully applied to the equirectangular camera images in order to extract seven overlapping directional pinhole cameras as shown in Fig. 5.9. This step was necessary, since the used SfM-pipeline wasn't able to handle full omnidirectional images.

Brief Chapter Summary

This chapter considered the generation of omnidirectional images from two perspectives. The first one describes the generation of omnidirectional source data from overlapping camera images using stitching techniques in a standard projection format (equirectangular). The second one describes the generation of suitable target projections with less distortion effects and their corresponding P2S-maps using a geo-

[73] https://www.ideepmon.eu/project/

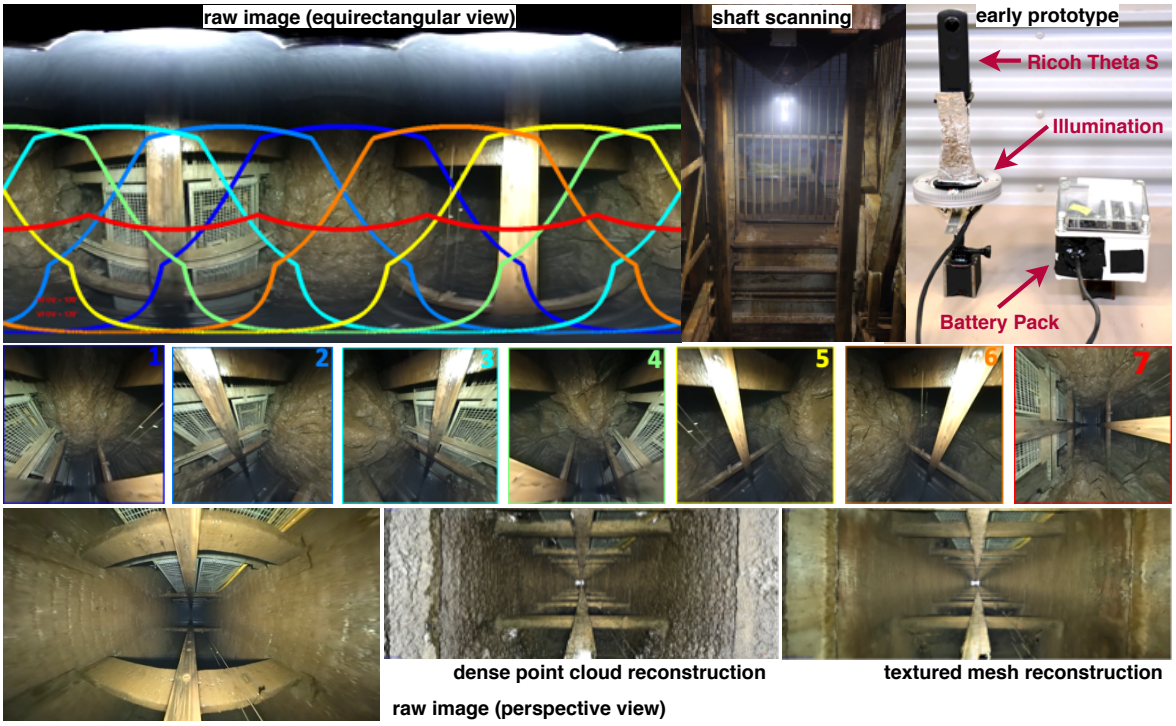

Fig. 5.9: Early *iDeepMon* prototype of a mine shaft scanning device based on image data (1st row). A *Ricoh Theta S* camera captures full omnidirectional images in equirectangular format, which are converted to seven overlapping directional pinhole camera images (2nd row). The 3rd row shows a raw footage and the corresponding dense point cloud reconstruction as well as the textured 3d model at the same location.

graphic library. The chapter explained a workflow that converts a source projection to a target projection based on their corresponding P2S-maps only. Finally, the proposed workflow was successfully applied to a real world example.

6 Relations between Two Camera Spheres

Brief Chapter Overview

This chapter comprehensively explaines the relation of point correspondences between two camera spheres. Section 6.1 describes the projection geometry on unit sphere and Section 6.2 explains how to recover 3d information from point correspondences using a DLT triangulation method in Section 6.2.1 and a novel closed-form midpoint triangulation method in Section 6.2.2. Section 6.3 derives the epipolar geometry for spherical cameras, followed by Section 6.4 that concentrates on the transformation recovery from essential matrix. It explaines cheirality in Section 6.4.1 and describes a standard procedure in Section 6.4.2 as well as two novel procedures in Sections 6.4.3 and 6.4.4 to resolve ambiguities from essential matrix decomposition. Essential matrix estimation algorithms are presented and compared in Section 6.5, where Section 6.5.1 describes the evaluation strategy, Section 6.5.2 refers to the error metric, Section 6.5.3 evaluates the obtained estimation results and Section 6.5.4 gives final remarks.

Section 6.6 is concerned with the optimization of derived two-view transformations. Section 6.6.1 presents known epipolar-based distance functions, whereas Section 6.6.2 introduces novel projection-based distance functions, which are all evaluated in Section 6.6.3. Going on with Section 6.7 that concentrates on two-view transformation scaling using available depth data. Section 6.7.1 proposes a method to obtain translation scale estimates and Section 6.7.2 explains translation scale optimization. Estimated and optimized translation scales are compared in Section 6.7.3. Finally Section 6.8 details the identification of degenerate motion and degenerate structure using homography. Section 6.8.1 describes homography applied to camera spheres. Section 6.8.2 explains the estimation of a homography matrix from point correspondences and Section 6.8.3 introduces a homography optimization approach. The relation between pure rotation and homography is examined in Section 6.8.4 and Section 6.8.5 combines homography and epipolar constraints to identify degenerate motion and degenerate structure.

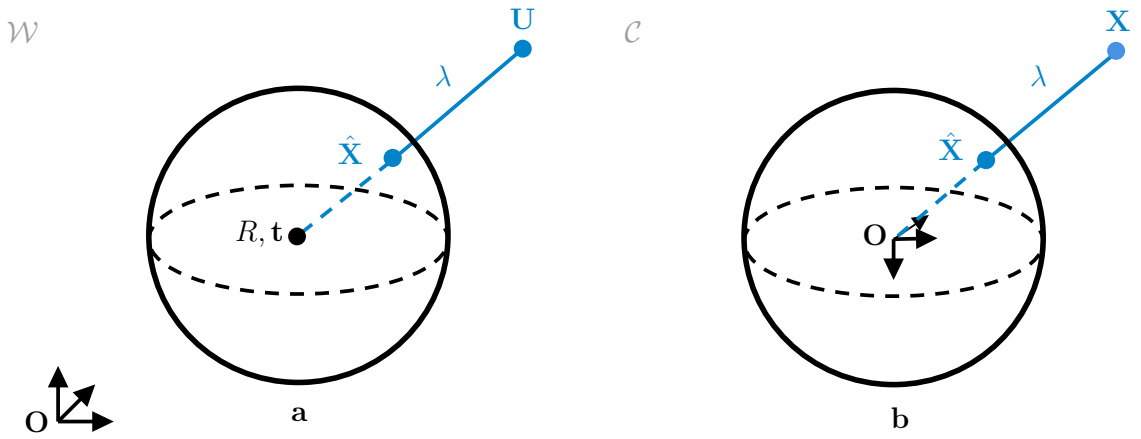

Fig. 6.1: Projection geometry on unit sphere: **a)** in world frame $\mathcal{W}$, **b)** in camera frame $\mathcal{C}$.

6.1 Forward and Backward Projection

The camera pose $[R, \mathbf{t}]$ describes the transformation from camera frame $\mathcal{C}$ to world frame $\mathcal{W}$. The inverse transformation from $\mathcal{W}$ to $\mathcal{C}$ is given by $\left[R^T, -R^T\mathbf{t}\right]$. The forward projection (or simply projection) maps $\mathbf{U} \in \mathcal{W}$ to $\hat{\mathbf{X}} \in \mathcal{S}$ and is described by

$$\boxed{\hat{\mathbf{X}} = \frac{\mathbf{X}}{\|\mathbf{X}\|} = \frac{R^T(\mathbf{U} - \mathbf{t})}{\|R^T(\mathbf{U} - \mathbf{t})\|}}. \tag{6.1}$$

The inverse mapping is called backward projection (or simply back-projection) given by

$$\boxed{\mathbf{U} = R\hat{\mathbf{X}}\lambda + \mathbf{t}}. \tag{6.2}$$

The unit sphere $\mathcal{S}$ is a subspace of the camera frame $\mathcal{C}$. Since $\|R\hat{\mathbf{X}}\| = 1$, λ represents the entire length from the optical camera center $[R, \mathbf{t}]$ to the 3d world point $\mathbf{U}$. Depth values can be re-obtained from a correspondence between image point and 3d world point. Taking Eq. (6.2) and subtracting $\mathbf{t}$ leads to

$$\underbrace{\mathbf{U} - \mathbf{t}}_{\mathbf{b}} = \underbrace{R\hat{\mathbf{X}}}_{\mathbf{a}}\lambda,$$

which can be solved for λ using the least squares solution $\lambda = \mathbf{a}^T\mathbf{b}/\|\mathbf{a}\|^2$ (as explained in Appendix A.2, page 174), which leads to the following equation

$$\boxed{\lambda = R\hat{\mathbf{X}}^T(\mathbf{U} - \mathbf{t})}. \tag{6.3}$$

6.2 Triangulation

Triangulation recovers the position of a 3d point from its given projection in two or more cameras. It represents a core element in stereo vision, SfM (Section 2.2, page 19), VO (Section 2.1.1, page 17) and accordingly Visual SLAM (Section 2.1, page 14). This section revisites a linear least squares triangulation method based on DLT [101, 96],

which is widely used in computer vision and robotics. It also describes a new approach, which is called *alternative* midpoint method [156, 157] obtaining depth information directly in a closed-form solution.

6.2.1 Linear Least Squares Method

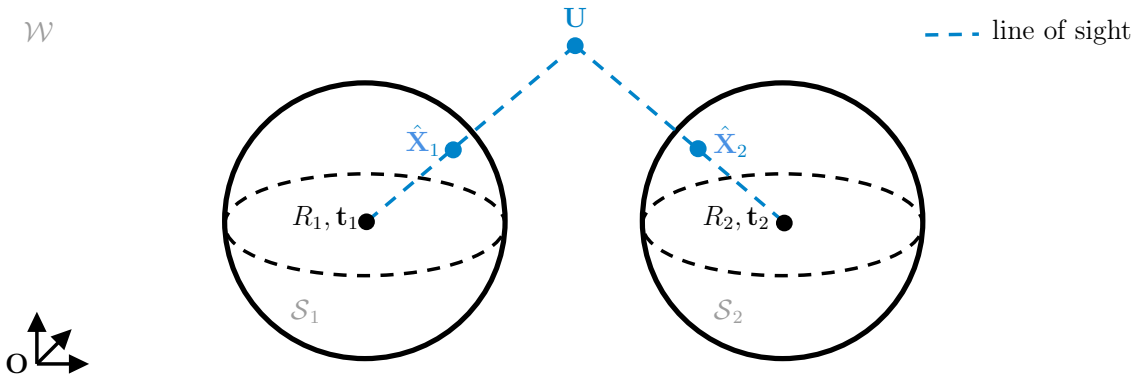

Fig. 6.2: The linear least squares triangulation method is based on collinearity between $\hat{\mathbf{X}}_1$ and $\mathbf{U}$ as well as between $\hat{\mathbf{X}}_2$ and $\mathbf{U}$.

The linear least squares triangulation method finds a 3d point that is closest to the intersection of two or more LoSs [93] from corresponding camera projections.
The following explains this method applied to image points on unit sphere. A point $\mathbf{U} \in \mathcal{W}$ is observed by two or more cameras $n \geq 2$ and their corresponding image projections $\hat{\mathbf{X}}_i \in \mathcal{S}_i$, $i = 1, \ldots, n$ on unit sphere as illustrated in Fig. 6.2. The linear least squares triangulation method is based on the collinearity between $\hat{\mathbf{X}}_i$ and $\mathbf{U}$. Taking the back-projection function Eq. (6.2) on page 88, subtracting $\mathbf{t}_i$ and cross-multiplying $R_i\hat{\mathbf{X}}_i$ in order to eleminate λ_i yields:

$$R_i\hat{\mathbf{X}}_i \times \left(\mathbf{U} - \mathbf{t}_i\right) = \underbrace{R_i\hat{\mathbf{X}}_i \times R_i\hat{\mathbf{X}}_i}_{\mathbf{a}\times\mathbf{a}=0}\lambda_i$$

$$\boxed{R_i\hat{\mathbf{X}}_i \times \left(\mathbf{U} - \mathbf{t}_i\right) = \mathbf{0}}\,. \tag{6.4}$$

Substituting $R_i\hat{\mathbf{X}}_i = \mathring{\mathbf{X}}_i$ simplifies the equation to

$$\mathring{\mathbf{X}}_i \times \left(\mathbf{U} - \mathbf{t}_i\right) = \mathbf{0}\,.$$

Excluding $\mathbf{U}$ and rewriting the resulting equation in a matrix form $B\mathbf{x} = 0$ yields:

$$\begin{aligned}
\mathring{\mathbf{X}}_i \times \mathbf{U} - \underbrace{\mathring{\mathbf{X}}_i \times \mathbf{t}_i}_{-\mathbf{a}\times\mathbf{b}=\mathbf{b}\times\mathbf{a}} &= \mathbf{0} \\
\underbrace{\mathring{\mathbf{X}}_i \times \mathbf{U}}_{\mathbf{a}\times\mathbf{b}=[\mathbf{a}]_\times\mathbf{b}} + \mathbf{t}_i \times \mathring{\mathbf{X}}_i &= \mathbf{0} \\
[\mathring{\mathbf{X}}_i]_\times \mathbf{U} + \mathbf{t}_i \times \mathring{\mathbf{X}}_i &= \mathbf{0} \\
\left[\, [\mathring{\mathbf{X}}_i]_\times \;\middle|\; \mathbf{t}_i \times \mathring{\mathbf{X}}_i \,\right] \begin{pmatrix} \mathbf{U} \\ \hline 1 \end{pmatrix} &= \mathbf{0} \\
B \begin{pmatrix} \mathbf{U} \\ \hline 1 \end{pmatrix} &= \mathbf{0} \\
B\mathbf{x} &= \mathbf{0}
\end{aligned}$$

with

$$B = \begin{bmatrix} 0 & -\mathring{\mathbf{X}}_{i(3)} & \mathring{\mathbf{X}}_{i(2)} & \mathbf{t}_{i(2)}\mathring{\mathbf{X}}_{i(3)} - \mathbf{t}_{i(3)}\mathring{\mathbf{X}}_{i(2)} \\ \mathring{\mathbf{X}}_{i(3)} & 0 & -\mathring{\mathbf{X}}_{i(1)} & \mathbf{t}_{i(3)}\mathring{\mathbf{X}}_{i(1)} - \mathbf{t}_{i(1)}\mathring{\mathbf{X}}_{i(3)} \\ -\mathring{\mathbf{X}}_{i(2)} & \mathring{\mathbf{X}}_{i(1)} & 0 & \mathbf{t}_{i(1)}\mathring{\mathbf{X}}_{i(2)} - \mathbf{t}_{i(2)}\mathring{\mathbf{X}}_{i(1)} \end{bmatrix}$$

in order to solve the problem via DLT by finding a non-zero solution for $\mathbf{x}$ that minimizes $\|B\mathbf{x}\|$ subjecting $\|\mathbf{x}\| = 1$ [101, 96]. The number of rows in B can be reduced since only the first two ones are linearly independent. A detailed proof of B's rank deficiency is described in Appendix A.3, page 175. Eliminating the linearly dependent third row reduces B to

$$B = \begin{bmatrix} 0 & -\mathring{\mathbf{X}}_{i(3)} & \mathring{\mathbf{X}}_{i(2)} & \mathbf{t}_{i(2)}\mathring{\mathbf{X}}_{i(3)} - \mathbf{t}_{i(3)}\mathring{\mathbf{X}}_{i(2)} \\ \mathring{\mathbf{X}}_{i(3)} & 0 & -\mathring{\mathbf{X}}_{i(1)} & \mathbf{t}_{i(3)}\mathring{\mathbf{X}}_{i(1)} - \mathbf{t}_{i(1)}\mathring{\mathbf{X}}_{i(3)} \end{bmatrix}.$$

All corresponding image projections n are stacked into B forming a linear system with

$$B = \begin{bmatrix} 0 & -\mathring{\mathbf{X}}_{1(3)} & \mathring{\mathbf{X}}_{1(2)} & \mathbf{t}_{1(2)}\mathring{\mathbf{X}}_{1(3)} - \mathbf{t}_{1(3)}\mathring{\mathbf{X}}_{1(2)} \\ \mathring{\mathbf{X}}_{1(3)} & 0 & -\mathring{\mathbf{X}}_{1(1)} & \mathbf{t}_{1(3)}\mathring{\mathbf{X}}_{1(1)} - \mathbf{t}_{1(1)}\mathring{\mathbf{X}}_{1(3)} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & -\mathring{\mathbf{X}}_{n(3)} & \mathring{\mathbf{X}}_{n(2)} & \mathbf{t}_{n(2)}\mathring{\mathbf{X}}_{n(3)} - \mathbf{t}_{n(3)}\mathring{\mathbf{X}}_{n(2)} \\ \mathring{\mathbf{X}}_{n(3)} & 0 & -\mathring{\mathbf{X}}_{n(1)} & \mathbf{t}_{n(3)}\mathring{\mathbf{X}}_{n(1)} - \mathbf{t}_{n(1)}\mathring{\mathbf{X}}_{n(3)} \end{bmatrix} \in \mathbb{R}^{2n\times 4}.$$

Applying an SVD to B with

$$B = U\Sigma V^T$$

obtains $U \in \mathbb{R}^{2n\times 4}$ and $V \in \mathbb{R}^{4\times 4}$, whose colums contain the left and right orthogonal vectors, as well as the diagonal matrix $\Sigma \in \mathbb{R}^{4\times 4}$ containing the four singular values $\sigma_1 \dots \sigma_4$. As result, $V_{(:,4)}$ is the column vector which corresponds to the smallest singular value σ_4 in Σ. Hence $V_{(:,4)}$ is the solution on null space of B in least squares sense and the first three elements devided by the last one of the solution vector represent

the triangulated point

$$\mathbf{U} = \frac{V_{(1:3,4)}}{V_{(4,4)}}.$$

In order to achieve a more stable numeric solution[74] [96] being less prone to noise, it is recommended to normalize each row of B in advance such that

$$B_{(j,:)} = \frac{B_{(j,:)}}{\|B_{(j,:)}\|}, \quad j = 1, \ldots, 2n$$

before applying SVD. The main advantages of the linear least squares triangulation method are the ease of implementation and the simplicity of triangulation if more projections are availabe by simply adding these observations to B. Since its introduction it has become the *standard* triangulation method most times in combination with pose recovery from essential matrix decomposition as described in Section 6.4.2, page 98.

6.2.2 Alternative Midpoint Method

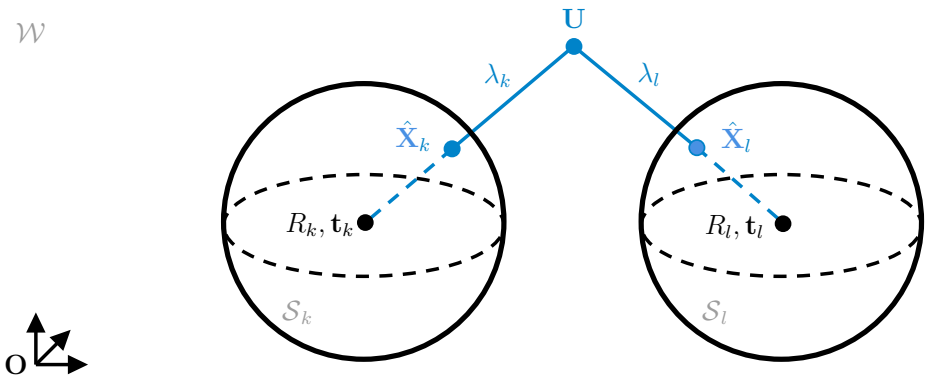

Fig. 6.3: The *alternative* midpoint method obtains the depth information λ_k and λ_l separately from each other. They are applied to the corresponding image points $\hat{\mathbf{X}}_k$ and $\hat{\mathbf{X}}_l$ in order to obtain $\mathbf{U}$, which is the midpoint between both back-projections.

This section describes a simplified triangulation approach, whose derivation - to the best of the author's knowledge - hasn't been published in the here presented form. In contrast to the linear least squares triangulation method, this one obtains the depth values λ_k, λ_l directly in order to determine the triangulated point by back-projecting the corresponding image points $\hat{\mathbf{X}}_k \in \mathcal{S}_k$, $\hat{\mathbf{X}}_l \in \mathcal{S}_l$ as shown in Fig. 6.3. Due to the presence of image noise as well as inaccuracies in camera alignment, the LoS are skewed and do not intersect. Consequently both back-projections do not coincide at $\mathbf{U} \in \mathcal{W}$, such that averaging obtains a midpoint solution, giving the method its name as it was originally introduced in [13, 14, 101]. It is nowadays also known as *classic* midpoint method, which obtains both depth values jointly by minimizing the squared distances between the LoS. The method has been further extended, allowing to obtain a midpoint solution from multiple views [241, 277, 37].

[74] https://stackoverflow.com/questions/2276445/triangulation-direct-linear-transform

The here presented version decouples the least squares optimization by obtaining each depth value separately. Suprisingly the same idea was discovered in [157] by developing an improved triangulation from two-view relations for directional pinhole cameras, which is named *alternative* midpoint method. The here presented *alternative* midpoint method presents a closed-form solution and neither minimize algebraic errors as the linear least squares method does nor minimize geometric errors based on image projection distances or angular projection distances [157].

Taking the back-projection function from Eq. (6.2) on page 88 for camera k and l such that:

$$\begin{aligned}\mathbf{U} &= R_k\hat{\mathbf{X}}_k\lambda_k + \mathbf{t}_k \\ \mathbf{U} &= R_l\hat{\mathbf{X}}_l\lambda_l \;\; + \mathbf{t}_l\,,\end{aligned}$$

and combining both yields:

$$\begin{aligned}R_k\hat{\mathbf{X}}_k\lambda_k + \mathbf{t}_k - R_l\hat{\mathbf{X}}_l\lambda_l - \mathbf{t}_l &= \mathbf{0} \\ \mathbf{t}_k - \mathbf{t}_l + R_k\hat{\mathbf{X}}_k\lambda_k - R_l\hat{\mathbf{X}}_l\lambda_l &= \mathbf{0}\,.\end{aligned} \tag{6.5}$$

Cross-multiplying $R_l\hat{\mathbf{X}}_l$ eliminates λ_l in Eq. (6.5):

$$\begin{aligned}R_l\hat{\mathbf{X}}_l \times \big(\mathbf{t}_k - \mathbf{t}_l\big) + \lambda_k\big(\underbrace{R_l\hat{\mathbf{X}}_l \times R_k\hat{\mathbf{X}}_k}_{\mathbf{a}\times\mathbf{b}=-(\mathbf{b}\times\mathbf{a})}\big) - \lambda_l\big(\underbrace{R_l\hat{\mathbf{X}}_l \times R_l\hat{\mathbf{X}}_l}_{\mathbf{a}\times\mathbf{a}=0}\big) &= \mathbf{0} \\ \underbrace{R_l\hat{\mathbf{X}}_l \times \big(\mathbf{t}_k - \mathbf{t}_l\big)}_{\mathbf{b}} - \lambda_k\big(\underbrace{R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l}_{\mathbf{a}}\big) &= \mathbf{0}\,.\end{aligned} \tag{6.6}$$

Analogous cross-multiplying $R_k\hat{\mathbf{X}}_k$ removes λ_k in Eq. (6.5) such that:

$$\begin{aligned}R_k\hat{\mathbf{X}}_k \times \big(\mathbf{t}_k - \mathbf{t}_l\big) + \lambda_k\big(\underbrace{R_k\hat{\mathbf{X}}_k \times R_k\hat{\mathbf{X}}_k}_{\mathbf{a}\times\mathbf{a}=0}\big) - \lambda_l\big(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\big) &= \mathbf{0} \\ \underbrace{R_k\hat{\mathbf{X}}_k \times \big(\mathbf{t}_k - \mathbf{t}_l\big)}_{\mathbf{b}} - \lambda_l\big(\underbrace{R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l}_{\mathbf{a}}\big) &= \mathbf{0}\,.\end{aligned} \tag{6.7}$$

An alternative derivation approach for Eqs. (6.6) and (6.7) is given in Appendix A.4, page 177. Both Eqs. (6.6) and (6.7) can be simplified to a linear least squares scaling problem $\mathbf{b} - \lambda\mathbf{a} = \mathbf{0}$ with the corresponding solution $\lambda = \mathbf{a}^T\mathbf{b}/\|\mathbf{a}\|^2$. This leads to the

following solutions:

$$\boxed{\lambda_k = \frac{\left(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right)^T \left(R_l\hat{\mathbf{X}}_l \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right)}{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\|^2}} \tag{6.8}$$

$$\boxed{\lambda_l = \frac{\left(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right)^T \left(R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right)}{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\|^2}}, \tag{6.9}$$

which are similiar to the ones as presented in [156, 157], where the authors describe the solution as relative pose between both cameras instead of absolute poses.

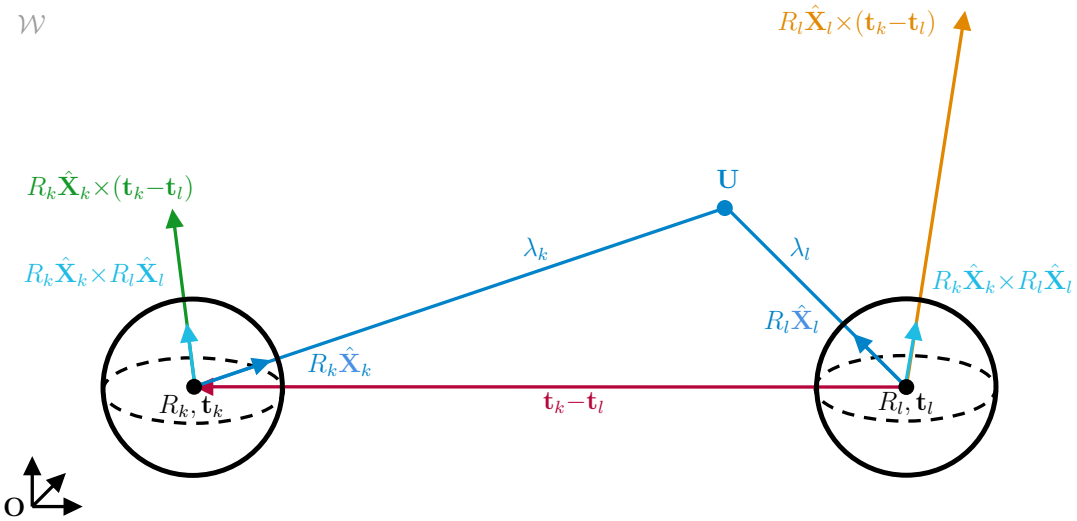

Fig. 6.4: Geometric explanation of the *alternative* midpoint method showing that λ_k is the relations between $R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l \leftrightarrow R_l\hat{\mathbf{X}}_l \times \left(\mathbf{t}_k - \mathbf{t}_l\right)$ and λ_l is the relations between $R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l \leftrightarrow R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)$.

Fig. 6.4 shows the geometric explanation of the described equations. Having a closer look clarifies that $R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l$ and $R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)$ as well as $R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l$ and $R_l\hat{\mathbf{X}}_l \times \left(\mathbf{t}_k - \mathbf{t}_l\right)$ are collinear, which Eqs. (6.6) and (6.7) also describe. These collinearity constraints further simplify Eq. (6.8) to

$$\boxed{\lambda_k = \frac{\left\|R_l\hat{\mathbf{X}}_l \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right\|}{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\|}}, \tag{6.10}$$

and analogous Eq. (6.9) to

$$\boxed{\lambda_l = \frac{\left\|R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right\|}{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\|}} \tag{6.11}$$

achieving similiar results as in [157]. Appendix A.5 page 178 gives detailed information about the simplification process. The derived depth values λ_k and λ_l are used in combination with Eq. (6.2), page 88 for back-projection to obtain $\mathbf{U}_k$ and $\mathbf{U}_l$. Using

real world data requires to average the individual solutions to derive the midpoint

$$\mathbf{U} = \frac{\mathbf{U}_k + \mathbf{U}_l}{2}.$$

In [157] the authors obtain the midpoint solution by applying an inverse depth weighting. Fortunately, this part of the triangualtion process is not considered here, sinces the here presented SCME pipeline does not need any triangulated 3d information and only relies on derived depth values.

Special cases

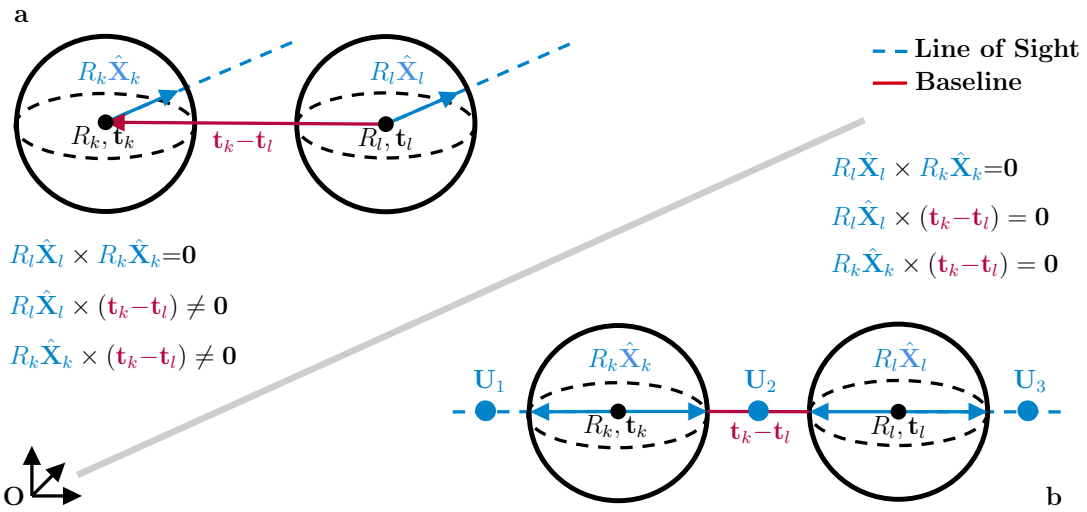

Fig. 6.5: Special cases of the *alternative* midpoint method where depth estimation fails. **a)** LoS are parallel such that $R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l = \mathbf{0}$. In that case $\mathbf{U}$ lies at infinity. **b)** LoS and baseline $\mathbf{t}_k - \mathbf{t}_l$ are collinear, if $\mathbf{U}$ lies on $\mathbf{t}_k - \mathbf{t}_l$. The drawing shows three examplary cases where depth estimation fails due to the location of $\mathbf{U}_1$, $\mathbf{U}_2$, $\mathbf{U}_3$.

This section shortly discusses the limitations of the *alternative* midpoint method, where depth estimation fails. By means of the reduced solutions, obvious restrictions are clearly visible, namely $R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l \neq \mathbf{0}$ (LoS mustn't be parallel, which is only the case if $\mathbf{U}$ lies at infinitiy), $\mathbf{t}_k - \mathbf{t}_l \neq \mathbf{0}$ (relative camera transformation must not stem from pure rotation) as well as $R_k\hat{\mathbf{X}}_k \times (\mathbf{t}_k - \mathbf{t}_l) \neq \mathbf{0}$ and $R_l\hat{\mathbf{X}}_l \times (\mathbf{t}_k - \mathbf{t}_l) \neq \mathbf{0}$ (LoS and baseline $\mathbf{t}_k - \mathbf{t}_l$ mustn't be parallel, which is the case if $\mathbf{U}$ and $\mathbf{t}_k - \mathbf{t}_l$ are collinear). The introduced cases are visualized in Fig. 6.5 and hold in theory. Due to the nature of real world data, these cases potentially will never occur.

6.3 Epipolar Geometry

The basic idea of the epipolar geometry goes back to von Sanden's work about photogrammetry [266], where he described a matrix similar to the essential matrix. Kruppa proved that a minimum number of five points solves the orientation problem. It is a landmark paper in computer vision and that's why this work recently was not

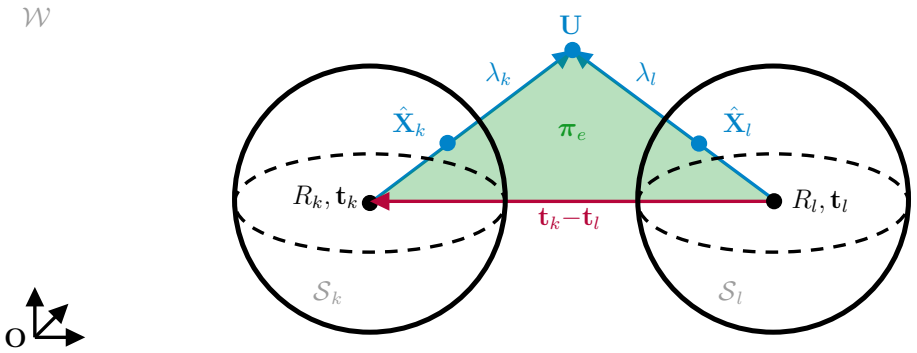

Fig. 6.6: Graphical representation of the epipolar geometry, which is based on the coplanarity between $\hat{\mathbf{X}}_k$, $\hat{\mathbf{X}}_l$ and the baseline $\mathbf{t}_k - \mathbf{t}_l$, since all three elements lie on the epipolar plane $\boldsymbol{\pi}_e$.

only translated into english but also the terminology was adaped to the computer vision jargon [72]. Longuet-Higgins introduced the essential matrix concept to the computer vision community [168], which was further investigated in [261, 110, 269, 108, 97]. In some literature the epipolar geometry is also mentioned as bilinear function or as bilinear constraint [176, 177]. The following explains the derivation of the essential matrix equation for image coordinates on a unit sphere.

Starting with Eq. (6.5), page 92

$$\mathbf{t}_l - \mathbf{t}_k = R_k \hat{\mathbf{X}}_k \lambda_k - R_l \hat{\mathbf{X}}_l \lambda_l$$

and cross-multiplying $R_k \hat{\mathbf{X}}_k$ from the right eliminates λ_k:

$$\begin{aligned} R_k \hat{\mathbf{X}}_k \times (\mathbf{t}_l - \mathbf{t}_k) &= \underbrace{(R_k \hat{\mathbf{X}}_k \times R_k \hat{\mathbf{X}}_k)}_{\mathbf{a}\times\mathbf{a}=0} \lambda_k - (R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l) \lambda_l \\ R_k \hat{\mathbf{X}}_k \times (\mathbf{t}_l - \mathbf{t}_k) &= -(R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l) \lambda_l \,. \end{aligned}$$

Scalar-multiplying $R_l \hat{\mathbf{X}}_l$ eliminates λ_l:

$$\begin{aligned} R_l \hat{\mathbf{X}}_l \bullet \underbrace{(R_k \hat{\mathbf{X}}_k \times (\mathbf{t}_l - \mathbf{t}_k))}_{\mathbf{a}\times\mathbf{b}=-\mathbf{b}\times\mathbf{a}} &= - \underbrace{R_l \hat{\mathbf{X}}_l \bullet (R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l)}_{\mathbf{a}\bullet(\mathbf{b}\times\mathbf{a})=0} \lambda_l \\ (R_l \hat{\mathbf{X}}_l)^T ((\mathbf{t}_k - \mathbf{t}_l) \times R_k \hat{\mathbf{X}}_k) &= 0 \,. \end{aligned} \tag{6.12}$$

Eq. (6.12) expresses the coplanarity between baseline $\mathbf{t}_k - \mathbf{t}_l$, $R_l \hat{\mathbf{X}}_l$ and $R_k \hat{\mathbf{X}}_k$, which form the epipolar plane $\boldsymbol{\pi}_e$ as illustrated in Fig. 6.6. Transferring Eq. (6.12) into camera frame $\mathcal{C}_l$, camera l becomes the center of origin $[I, \mathbf{0}]$ and camera k is expressed as relative transformation $[R, \mathbf{t}] = [R_l^T R_k, R_l^T (\mathbf{t}_k - \mathbf{t}_l)]$ such that:

$$\begin{aligned} \hat{\mathbf{X}}_l^T (\mathbf{t} \times R \hat{\mathbf{X}}_k) &= 0 \\ \hat{\mathbf{X}}_l^T ([\mathbf{t}]_\times R \hat{\mathbf{X}}_k) &= 0 \,. \end{aligned}$$

Substituting $[\mathbf{t}]_\times R = E$ yields the well-known epipolar geometry in essential matrix form for spherical cameras [256, 196, 195]

$$\boxed{\hat{\mathbf{X}}_l^T E \hat{\mathbf{X}}_k = 0}. \tag{6.13}$$

Due to its simple structure, the epipolar geometry in essential matrix form is a key element to recover a transformation from feature correspondences between two camera images. Hence it is often referred as a synonym for two-view geometry. In general the two-view geometry describes all available geometric relations between feature correspondences in two different camera views of the same 3d scene. Accordingly the two-view geometry is not limited to an epipolar constraint and can be also described by closed-form projection constraints as Section 6.6, page 107 shows.

6.4 Transformation Recovery from Essential Matrix

The relative transformation from camera k to camera l is covered by the essential matrix E. Since Eq. (6.13), page 96 uses coplanarity constraints without considering the orientation of the LoS, the decomposition of E into R and $\mathbf{t}$ generally has four solutions [96, 196] as shown in Fig. 6.7. Decomposing $E = U\Sigma V^T$ via SVD straightforwardly

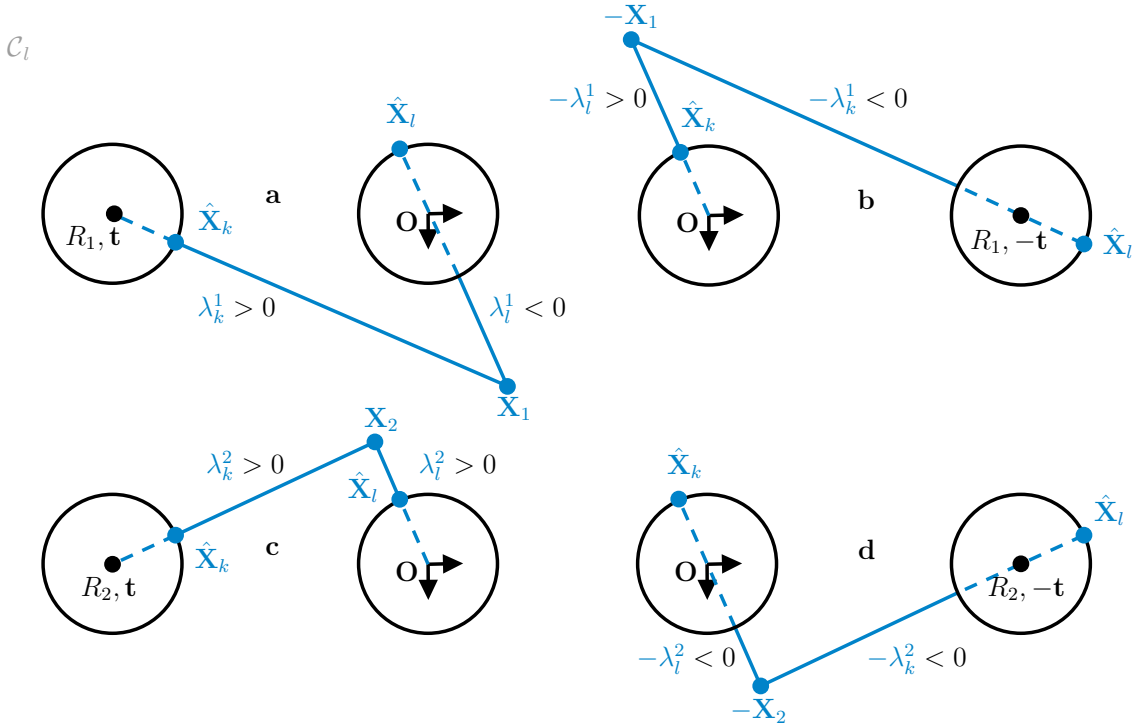

Fig. 6.7: Four solutions of an essential matrix decomposition showing that **c** is the geometrically correct solution with two positive depth values $\lambda_k > 0$ and $\lambda_l > 0$ satisfying cheirality.

obtains $R_1 = UWU^T$, $R_2 = UW^TU^T$ and $\mathbf{t} \pm U_{(:,3)}$ with

$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} .$$

Since the sign of $\mathbf{t}$ is arbitrary, the sign of E is also arbitrary:

$$\begin{aligned} E &= [\mathbf{t}]_\times R_1 & \qquad -E &= [\mathbf{t}]_\times R_2 \\ -E &= [-\mathbf{t}]_\times R_1 & \qquad E &= [-\mathbf{t}]_\times R_2 \end{aligned}$$

and does not violate the epipolar constraint:

$$\begin{aligned} \hat{\mathbf{X}}_l^T(-E)\hat{\mathbf{X}}_k &= 0 \\ -(\hat{\mathbf{X}}_l^T E \hat{\mathbf{X}}_k) &= 0 \\ \hat{\mathbf{X}}_l^T E \hat{\mathbf{X}}_k &= 0 \,. \end{aligned}$$

The forementioned decomposition does not always ensure the rotation to represent the correct geometric explanation. In [268] the authors state that the signs of R_1 as well as R_2 and accordingly the sign of W [24] are also arbitrary leading to eight possible solutions [165]. In order to halve the number of solutions the signs of R_1 and R_2 can be determined by verifying the determinant

$$R_i = \begin{cases} -R_i, & \text{if } det(R_i) = -1 \\ R_i, & \text{otherwise} \,. \end{cases} \qquad i = 1, 2$$

E has only five degrees of freedom, whereas R_i and $\mathbf{t}$ have three degrees of freedom each. This leads to a scale ambiguity of $\mathbf{t}$, subjecting $\|\mathbf{t}\| = 1$ [96].

The following sections describe the definition of cheirality and explain three procedures resolving ambiguity from essential matrix decomposition. The standard procedure (Section 6.4.2) is a review and compilation of known geometric properties and simplification possibilities. The simplified procedure (Section 6.4.3) and especially the improved procedure (Section 6.4.4) - as they were presented in this work - haven't been discovered in literature by the author so far. They describe novel approaches reducing calculation effort to resolve ambiguity.

6.4.1 Cheirality

Image points satifsying $\hat{\mathbf{X}}_l^T E \hat{\mathbf{X}}_k = 0$ do not necessarily correspond to any real geometry [270]. Cheirality defines a 3d point to be visible when it is located in front of the camera [96] and accordingly has a positive depth [98, 100] and positive viewing direction [155]. These constraints can be applied to omnidirectional cameras [270] as well, which additionally use projection as a cheirality constraint.

Projecting a 3d point onto the image plane of a directional camera does not allow to distinguished whether the 3d point originates from the front or the back, since the projection from both directions leads to the same position on image. Omnidirectional cameras have no real front or back, a projection from the back (opposite direction) yields an image point on the opposite site of the unit sphere. Thus an additional cheirality validation is not needed and thus reduces computational cost. Directional cameras in conjunction with P2S-maps have the advantage to use projection as cheirality constraint, too.

6.4.2 Standard Procedure

Going on with finding the correct solution for R and $\mathbf{t}$, the standard procedure uses the linear least squares triangulation method from Section 6.2.1, page 89 to obtain $\mathbf{X}_1$, $\mathbf{X}_2$ for the possible solutions $[R_1, \mathbf{t}], [R_2, \mathbf{t}]$. The corresponding depth values can be optained using Eq. (6.3), page 88 and applying $[R_l, \mathbf{t}_l] = [I, \mathbf{0}]$, $[R_k, \mathbf{t}_k] = [R_i, \mathbf{t}]$, $\mathbf{U} = \mathbf{X}_i$ yields:

$$\begin{aligned} \lambda_k^i &= \left(R_i \hat{\mathbf{X}}_k\right)^T \left(\mathbf{X}_i - \mathbf{t}\right) \\ \lambda_l^i &= \hat{\mathbf{X}}_l^T \mathbf{X}_i, \qquad\qquad i = 1, 2\,. \end{aligned}$$

The geometrically correct combination satisfies $\lambda_k > 0$ and $\lambda_l > 0$ as exemplary illustrated in Fig. 6.7. The calculation effort is reduced since $\mathbf{X}_1(R_1, \mathbf{t}) = -\mathbf{X}_1(R_1, -\mathbf{t})$ and $\mathbf{X}_2(R_2, \mathbf{t}) = -\mathbf{X}_2(R_2, -\mathbf{t})$ [165] as shown in Tab. 6.1. At least two points $(\mathbf{X}_1, \mathbf{X}_2)$

Tab. 6.1: Possible transformation combinations and their corresponding relations

Combination	Triangulated Point	Depth
$R_1, \mathbf{t}$	$\mathbf{X}_1$	λ_k^1, λ_l^1
$R_1, -\mathbf{t}$	$-\mathbf{X}_1$	$-\lambda_k^1, -\lambda_l^1$
$R_2, \mathbf{t}$	$\mathbf{X}_2$	λ_k^2, λ_l^2
$R_2, -\mathbf{t}$	$-\mathbf{X}_2$	$-\lambda_k^2, -\lambda_l^2$

need to be triangulated in order to obtain the four depth values $(\lambda_k^1, \lambda_l^1, \lambda_k^2, \lambda_l^2)$, which are used to validate cheirality such that:

$$[R, \mathbf{t}] = \begin{cases} [R_i, \mathbf{t}]\,, & \text{if } \lambda_k^i > 0 \text{ and } \lambda_l^i > 0 \\ [R_i, -\mathbf{t}]\,, & \text{if } \lambda_k^i < 0 \text{ and } \lambda_l^i < 0\,. \end{cases} \qquad i = 1, 2$$

The standard procedure cannot identify pure rotation, which is also referred as degenerate motion or degeneracy [269], since the decomposition of E always obtains $\mathbf{t} \neq \mathbf{0}$. The addtional criterion [269, 24, 285] to be validated is:

$$[R, \mathbf{t}] = \begin{cases} [R, \mathbf{0}]\,, & \text{if } \left\| R\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l \right\| = 0 \\ [R, \mathbf{t}]\,, & \text{elsewhere.} \end{cases}$$

This means, if the LoS from camera k is transferred to camera l and both LoS overlay such that $\left\|R\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l\right\| = 0$, then the transformation stems from a pure rotation. Finally, the standard procedure needs seven calculation steps $(\mathbf{X}_1, \mathbf{X}_2, \lambda_k^1, \lambda_l^1, \lambda_k^2, \lambda_l^2, \|R\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l\|)$ in order to resolve the ambiguity from essential matrix decomposition.

6.4.3 Simplified Procedure

The here presented simplified procedure is based on the derived equations of the *alternative* midpoint method from Section 6.2.2, page 91, which optains the depth values directly. Taking Eqs. (6.8) and (6.9), page 93 and applying $[R_l, \mathbf{t}_l] = [I, \mathbf{0}]$, $[R_k, \mathbf{t}_k] = [R_i, \mathbf{t}]$ leads to:

$$\lambda_k^i = \frac{\left(R_i\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l\right)^T\left(\hat{\mathbf{X}}_l \times \mathbf{t}\right)}{\left\|R_i\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l\right\|^2}$$
$$\lambda_l^i = \frac{\left(R_i\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l\right)^T\left(R_i\hat{\mathbf{X}}_k \times \mathbf{t}\right)}{\left\|R_i\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l\right\|^2}\,.$$

Since $\left\|R_i\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l\right\| > 0$ (except in case of pure rotation), it doesn't affect the sign of λ and removing the term leads to a calculation of approximate depth values as expressed by:

$$\tilde{\lambda}_k^i = \left(R_i\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l\right)^T\left(\hat{\mathbf{X}}_l \times \mathbf{t}\right) \tag{6.14}$$
$$\tilde{\lambda}_l^i = \left(R_i\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l\right)^T\left(R_i\hat{\mathbf{X}}_k \times \mathbf{t}\right) \qquad i, j = 1, 2\,. \tag{6.15}$$

This leads to a decreased computational effort and prevents the division by zero in case of pure rotation $R_i\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l = \mathbf{0}$ such that both approximate depth values become zero. Furthermore, the cheirality constraint can be still applied, since it validates the sign of the depth value rather than the length itself. On the basis of Eqs. (6.14) and (6.15) it is comprehensible that $\tilde{\lambda}_k^i(R_i, \mathbf{t}) = -\tilde{\lambda}_k^i(R_i, -\mathbf{t})$ and $\tilde{\lambda}_l^i(R_i, \mathbf{t}) = -\tilde{\lambda}_l^i(R_i, -\mathbf{t})$ reducing the depth calculation to four values $(\lambda_k^1, \lambda_l^1, \lambda_k^2, \lambda_l^2)$ in order to perform the cheirality test:

$$[R, \mathbf{t}] = \begin{cases} [R_i, \mathbf{t}]\,, & \text{if } \tilde{\lambda}_k^i > 0 \text{ and } \tilde{\lambda}_l^i > 0 \\ [R_i, -\mathbf{t}]\,, & \text{if } \tilde{\lambda}_k^i < 0 \text{ and } \tilde{\lambda}_l^i < 0 \\ [R_i, \mathbf{0}]\,, & \text{if } \tilde{\lambda}_k^i = 0 \text{ and } \tilde{\lambda}_l^i = 0\,. \end{cases} \qquad i = 1, 2$$

The simplified procedure resolves the ambiguity from essential matrix decomposition within four calculation steps $(\tilde{\lambda}_k^1, \tilde{\lambda}_l^1, \tilde{\lambda}_k^2, \tilde{\lambda}_l^2)$ including the detection of pure rotation.

6.4.4 Improved Procedure

This approach is different compared to the previous ones specifying a transformation which is validated by means of the obtained depth values. The improved procedure assumes the depth value in the first camera to be positive and seeks for the transforma-

tion that obtains a positive depth value in the second camera. Starting with Eq. (6.5), page 92 and applying $[R_l, \mathbf{t}_l] = [I, \mathbf{0}]$, $[R_k, \mathbf{t}_k] = [R, \mathbf{t}]$ yields

$$\mathbf{0} = \mathbf{t} + R\hat{\mathbf{X}}_k\lambda_k - \hat{\mathbf{X}}_l\lambda_l \,.$$

Cross-multiplying $\mathbf{t}$ such that

$$\mathbf{0} = \underbrace{\left(\mathbf{t} \times R\hat{\mathbf{X}}_k\right)}_{\mathbf{b}}\lambda_k - \underbrace{\left(\mathbf{t} \times \hat{\mathbf{X}}_l\right)}_{\mathbf{a}}\lambda_l$$

and solving for λ_l with in linear least squares sense $\lambda_l = \mathbf{a}^T\mathbf{b}/\|\mathbf{a}\|^2$ results to

$$\lambda_l = \frac{\left(\mathbf{t} \times \hat{\mathbf{X}}_l\right)^T\left(\mathbf{t} \times R\hat{\mathbf{X}}_k\right)}{\left\|\mathbf{t} \times \hat{\mathbf{X}}_l\right\|^2}\lambda_k \,, \tag{6.16}$$

which describes the relation between λ_l and λ_k. Assuming $\lambda_k > 0$ and $\left\|\mathbf{t} \times \hat{\mathbf{X}}_l\right\|^2 > 0$ as long as $\mathbf{t} \neq \mathbf{0}$ (which is prevented due to the nature of the essential matrix decomposition) and $\mathbf{t} \times \hat{\mathbf{X}}_l \neq \mathbf{0}$ (which belongs to a special case as described in Fig. 6.5**b** on page 94). Eq. (6.16) can be reduced to obtain approximate depth values

$$\tilde{\lambda}_l^i = \left(\mathbf{t} \times \hat{\mathbf{X}}_l\right)^T\left(\mathbf{t} \times R_i\hat{\mathbf{X}}_k\right) .$$

Since $\mathbf{t}$ is included in both terms, its sign has no influence on the sign of $\tilde{\lambda}_l^i$, which now depends on R_i only. Calculating $\tilde{\lambda}_l^1$ allows to find the correct rotation:

$$R = \begin{cases} R_1, & \text{if } \tilde{\lambda}_l^1 > 0 \\ R_2, & \text{otherwise.} \end{cases}$$

Using Eq. (6.14)

$$\tilde{\lambda}_k = \left(R\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l\right)^T\left(\hat{\mathbf{X}}_l \times \mathbf{t}\right)$$

resolves the correct translation even in case of degenerate motion:

$$\mathbf{t} = \begin{cases} \mathbf{t}, & \text{if } \tilde{\lambda}_k > 0 \\ -\mathbf{t}, & \text{if } \tilde{\lambda}_k < 0 \\ \mathbf{0}, & \text{if } \tilde{\lambda}_k = 0 \,. \end{cases}$$

The improved procedure needs two calculation steps only $(\tilde{\lambda}_l^1, \tilde{\lambda}_k)$ to find the correct transformation from essential matrix decomposition even for degenerate motion and outperforms both forementioned procedures concerning calculation effort. This comes particularly to bear if each point of a correspondence set is supposed to be validated

for outlier detection.

6.5 Two-View Estimation

This section presents and evaluates algorithms obtaining the relative transformation between two cameras based on corresponding image features. Longuet-Higgins developed an algorithm [168] that obtains the essential matrix from 8 point correspondences, which is sometimes referred as *classic* 8-point algorithm. Philip published the first 6-point non-iterative solver and showed that if 6 or more point correspondences are available the pose estimation problem can be solved from a linear formulation and hence yields an unique solution [200]. The relative pose problem for calibrated cameras needs at least 5 corresponding points to be solved resulting up to 10 distinct solutions [62, 61]. Nister developed the first non-linear 5-point algorithm that guaranteed to provide at most 10 solutions [191, 72] by formulating the pose estimation problem as a system of polynomal equations. Li and Hartley improved Nister's version in terms of numerical stability and processing time [163]. Further improvements were taken, e.g. Stewénius, Engels, and Nistér use Gröbner bases to solve the polynomal system [239] or Kukelova, Bujnak, and Pajdla make use of a polynomial eigenvalue method to recover the essential matrix solutions [150]. Li developed a non-linear 6-point algorithm that uses a hidden-variable technique [162]. On the contrary Fathian, Ramirez-Paredes, Doucette, Curtis, and Gans decouple translation and rotation estimation and do not solve the epipolar constraint embodied in an essential matrix [60, 59]. In their solution, a 5-point quaternion estimation technique obtains the rotation in a first step, translation as well as depth values are recovered simultaneously in an additional second step [58] via DLT. An overview of the evaluated pose estimation algorithms are given in Tab. 6.2. The mentioned Stewenius/Engels algorithm is an alternative to the Stewenius algorithm using a non-Gröbner base solver [239].

All considered algorithms have been developed for directional cameras and are now applied to spherical image coordinates as they are used for omnidirectional cameras.

6.5.1 Evaluation Strategy

Evaluation tests are performed under varying point configurations (structures) and motion conditions. The determined 6 cases are shown in Fig. 6.8 simulating directional and omnidirectional two-view camera configurations for full motion (**case a, b**) and degenerate (rotation-only) motion (**case d, e**). Furthermore, **case c** evaluates the

[1] https://www2.cs.duke.edu/courses/fall15/compsci527/notes/longuet-higgins.pdf
[2] https://github.com/SergioRAgostinho/five_point_algorithm
[3] https://mathworks.com/matlabcentral/fileexchange/67580-essential-matrix-estimation
[4] http://users.cecs.anu.edu.au/%7Ehartley/Software/5pt-6pt-Li-Hartley.zip
[5] https://sites.google.com/view/kavehfathian/code/quest-5-point

Tab. 6.2: Overview of selected solvers for two-view estimation

Method	Points	Solutions
8-point *classic*[1] [168]	8	1
5-point Nister[2] [191]	5	≤ 10
5-point Stewenius[3] [239]	5	≤ 10
5-point Stewenius/Engels[3] [239]	5	≤ 10
5-point Kukelova[3] [150]	5	≤ 10
5-point Li[4] [163]	5	≤ 10
6-point Li[4] [162]	6	≤ 10
QuEst[5] [60]	5	20

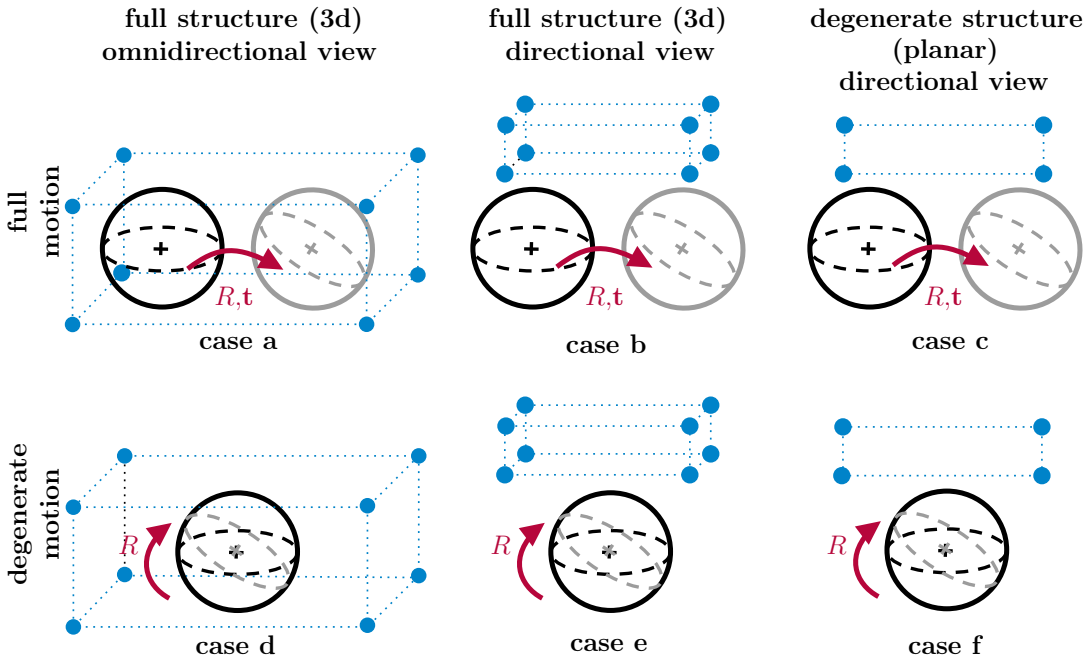

Fig. 6.8: Possible combination combinations of two-view cases conerning structure (3d or planar) and motion (full or degenerate)

algorithms' behaviour under a degenerate planar point configuration, since a plane can be only observed by a hemisphere, this case belongs to a directional view case. Usually homography is used in that case since it solves planar structure problems but does not work for arbitrary point configurations. The homography estimation spherical camera coordinates requires at least five planar point correspondences (Section 6.8.2,page 120), which cannot be ensured for all configurations under real world conditions such that homography is left out as reliable estimation technique in this evaluation. **Case f** focuses on degenerate motion combined with degenerate structure, which also belongs to a directional view case and suprisingly is seldomly mentioned in literature, where either degeneracy in struture or in motion is considered.

In each **case a-f** camera poses and structure points are generated randomly. The structure points are then projected onto each camera's sphere to obtain ground truth image coordinates, which are disturbed by Gaussian image noise $\mathcal{N}(0, \sigma^2_{\theta,\phi})$ with $\sigma_{\theta,\phi} = 0°, \ldots, 1°$, where σ_θ denotes be the noise standard deviation in polar direction and σ_ϕ denotes the noise standard deviation in azimuthal direction. The simulated image noise represents calibration and feature detection inaccuracies. Both, polar and azimuthal noise belong to the same noise level during simulation.
The relative rotation between both cameras is randomly sampled as a unit quaternion by normalizing a normal distributed four element vector. The random translation sampling is based on a normal distributed three element vector. 5000 iterations per noise level and per case are performed to achieve a statistical meaningful quantity to be evaluated. In each iteration, 8 structure points are sampled, where 5-point and 6-point algorithms use a subset. Even if an algorithm is able to work with more points, it is fed with the least point quantity in order to evaluate the algorithm's initial design performance.

Most of the solvers provide multiple essential matrix solutions to the relative orientation problem, except the linear *classic* 8-point algorithm as summarized in Tab. 6.2. This evaluation process uses the improved procedure from Section 6.4.4, page 99 for essential matrix decomposition. The number of positive depth values functions as coarse decision criterion for multiple essential matrix solutions. In terms of noisy data, essential matrix decomposition may also lead to ambiguous solutions since only a small quantity of 5 to 8 points are used. In that case all equal solutions per essential matrix are kept. The relative pose - from all preselected solutions - yielding the largest quantity of positive depth values from cheiralty validation will be selected. Even at this point it is possibile that multiple solutions share the same quantity of positive depth values. The relative pose yielding the least sum of squared geodesic distances (Eq. (6.20), page 108) will be selected, which is the second criterion for finer decision selection. At this point it should be also mentioned that the condition was rejected, which postulates that the correct essential matrix decomposition does not violate cheirality in any point correspondence. With increasing image noise, the pose estimation becomes more prone to error (independently of the chosen algorithm) such that some point correspondences may violate cheirality.

6.5.2 Error Metric

The error distance between ground truth R^{true} and estimated rotation R^{est} is defined as angle

$$\alpha_R = \cos^{-1}\left(\frac{\operatorname{trace}\left(\left(R^{\text{true}}\right)^T R^{\text{est}}\right) - 1}{2}\right), \tag{6.17}$$

where the trace function returns the sum of the entries on the main diagonal of the matrix. All pose estimation algorithms obtain translation up-to-scale making an Euclidean distance improper for evaluation purposes. Instead, the angle between the directions of the ground truth translation $\mathbf{t}^{\text{true}}$ and estimated one $\mathbf{t}^{\text{est}}$ is a convenient error definition [285] such that

$$\alpha_{\mathbf{t}} = \cos^{-1}\left(\frac{\left(\mathbf{t}^{\text{true}}\right)^T \mathbf{t}^{\text{est}}}{\|\mathbf{t}^{\text{true}}\| \, \|\mathbf{t}^{\text{est}}\|}\right). \tag{6.18}$$

$\alpha_{\mathbf{t}}$ is not obtained for degenerate motion **cases d-f**, since the ground truth is a null-vector. Sometimes a base vector is choosen as main translation direction. However, in this work there is no main direction, since random translation sampling covers all directions of a sphere equally and is not limited to a certain camera movement.

6.5.3 Evaluation of Estimation Algorithms

Case a, b

In **case a** as shown in Fig. 6.9, QuEst achieves the best estimation results concerning rotation and translation accuracy, followed by the *classic* 8-point algorithm and Li's non-linear 6-point algorithm, both beeing significantly less accurate. In **case b**, the 8-point algorithm performs best, followed by Li's non-linear 6-point algorithm, which obtains slightly similar results. Noticeably, QuEst isn't able to estimate the correct pose under noise-free condition. The same behaviour can be seen for all other 5-point algorithms (Nister, Stewenius, Stewenius/Engels, Kukelova, Li) in **case a, b**. Even if there is no perturbation, the algorithms are not able to obtain correct transformation estimates. Obviously, the 8-point algorithm achives overall good rotation and translation estimation results concerning directional and omnidirecitonal views. This is an opposite behaviour compared to other results in literature for directional cameras, where the *classic* 8-point algorithm performs worse [206, 60]. These circumstance led to the conclusion, that linear methods are highly sensitive to noise [268]. Pagani and Stricker [196] relate this to the fact that for directional cameras, it is recommended to normalize the points on image plane in a condition step to improve the stability of the results [96]. In this condition step, image points are transformed such that the mean is at origin and their average norm (their mean distance from origin) is $\sqrt{2}$. This step is not required using P2S-maps, where all image points are already spread around the center of a unit sphere. This might be a possible explanation for the improved *classic* 8-point algorithm's performance for directional and omnidirectional cameras in this work.

Case c

This special **case c** cannot be solved by any validated algorithm under noise-free condition. In contrast to the previous cases, there is no obvious algorithm, that performs best. Concerning rotation estimation, the *classic* 8-point algorithm and Li's non-linear

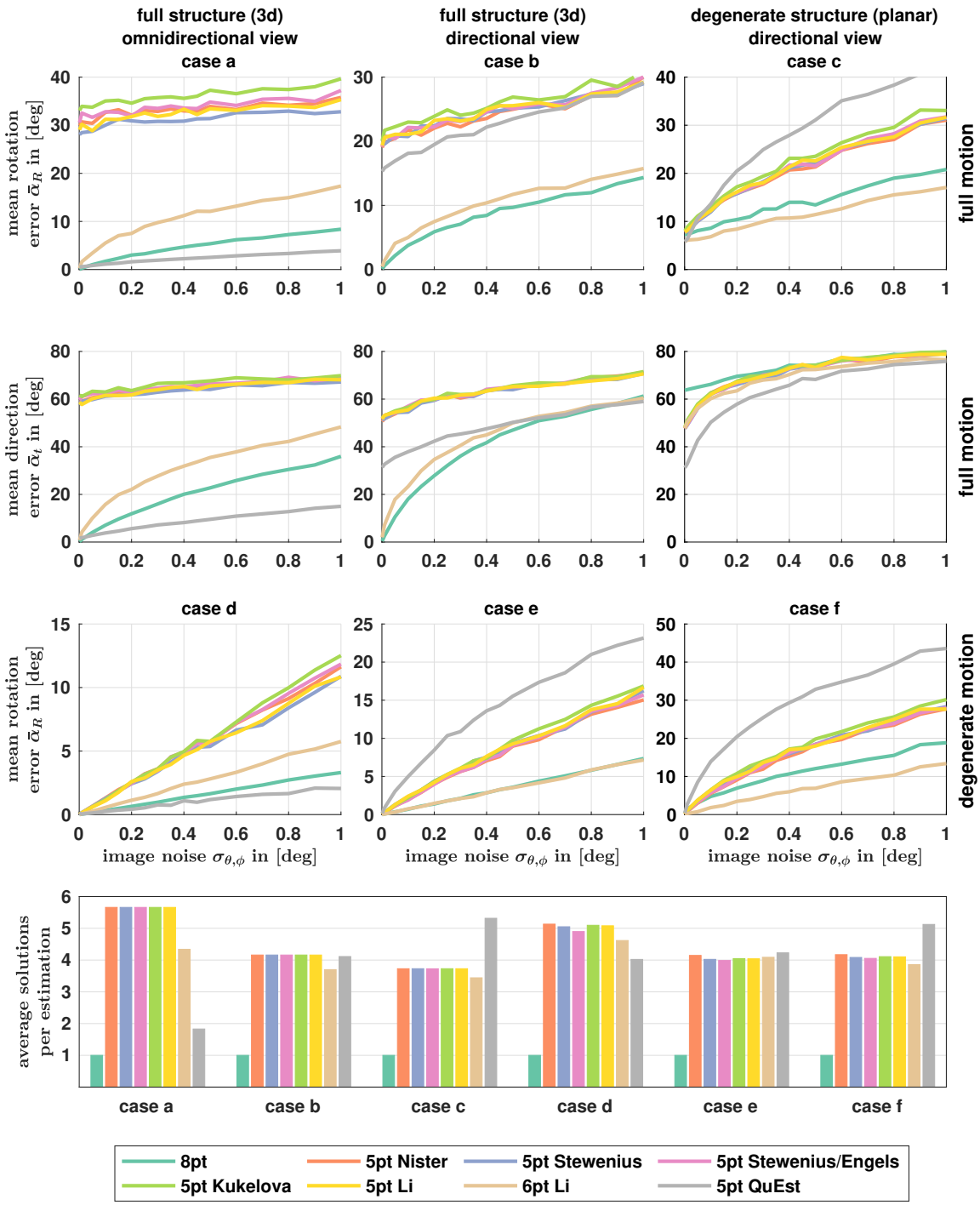

Fig. 6.9: Results from essential matrix estimation

6-point algorithm achieve less discrepancies, which are still unacceptable. Having a look at the translation estimation shows, that QuEst has less deviation, but has the worst performance for rotation estimation. A possible explanation for that behaviour might be the fact, that QuEst decouples rotation from translation estimation. Neither rotation estimation, nor translation estimation were satisfyingly obtained by any choosen algorithms. As mentioned in [59, 191], linear algorithms (6 and more points) do not work with point configurations on planes and in [201] the authors give an overview

of further degenerate structures such as cylinders. On the contrary, referring to [201] the 5-point Nister algorithm is unaffected by planar degeneracy for directional cameras. Fig. 6.9 shows a different behaviour, where none of the 5-point variants is able to estimate a correct transformation in a noiseless scenario. In general, neither linear, nor non-linear algorithms are able to handle planar structures.

Case d-f

All algorithms are able to estimate rotation from degenerate motion under noise-free condition. In **case d** QuEst achieves the best estimation results followed by the *classic* 8-point and Li's non-linear 6-point algorithm. The remaining 5-point algorithms (Nister, Stewenius, Stewenius/Engels, Kukelova, Li) obtain similar results, but significantly less accuarate compared to the other ones. Having a look at **case e** shows, that both algorithms - the *classic* 8-point and Li's non-linear 6-point algorithm - achieve nearly identical results. Again, the 5-point algorithms perform worse compared to the mentionen ones and QuEst has the poorest performance. **Case f**, which covers degeneracy in motion and in structure together, can be estimated by each of the validated algorithms. Li's non-linear 6-point algorithm achieves slightly better results compared to the *classic* 8-point algorithm, followed by the group of 5-point algorithms and completed by QuEst, which again has the worst performance. The *classic* 8-point algorithm achieves satisfying rotation estimates for spherical image coordinates, which is in contrast to directional cameras using image plane coordinates [269].

6.5.4 Concluding Remarks

The evaluation concerning essential matrix estimation under different predefined cases showed that the 5-point algorithms from Nister, Stewenius, Stewenius/Engels, Kukelova and Li achieve similar, but non-satisfying results. This leads to the conclusion that they are not able to work with spherical image coordinates under full motion scenarios as presented in **case a-c**. They are also less accurate under degenerate motion **case d-f**. None of the investigated eight algorithms is able to provide sufficient estimates for **case c**, which is solved by homography only (Section 6.8.1,page 119). Using image coordinates on unit sphere improves the pose estimation of the *classic* 8-point algorithm significantly compared to previous results in literature for directional cameras.
Li's non-linear 6-point algorithm is also suitable to work with spherical image point representation. QuEst shows anomalies. On the one hand this algorithm has the best performance in **case a, d**, but fails in **case b, e, f** (**case c** is not considered since all algorithms show poor performances). This is an interesting behaviour, because **case a, d** cover omnidirectional scenarios, for which this algorithm wasn't originally developed.

By means of the results presented in Fig. 6.9, the *classic* 8-point algorithm achieves overall satisfying results. Compared to Li's non-linear 6-point algorithm, which provides up to 5 essental matrix solutions, the *classic* 8-point algorithm

obtains one solution only, reducing the computational effort to resolve the correct transformation. A 6-point RANSAC needs less iteration steps compared to an 8-point RANSAC [218]. However for Li's non-linear 6-point algorithm the computational effort is 4 to 5 times higher per itertion in order to resolve the correct transformation. This consideration is a theoretical approach and does not include the actual runtime of each algorithm as it is assumed they are in equal magnitudes. [269] point out that linear algorithms are faster. This statement holds in theory, but also depends on the implementation itself and as shown in [60], where Li's 5 point algorithm is slightly faster than the *classic* 8-point algorithm.
Considering all mentioned aspects, the *classic* 8-point algorithm is preferred.

Noticed but not considered for evalutation

Kneip, Siegwart, and Pollefeys [142] describe an alternative 5-point approach that finds an exact global rotation independently of the translation and does not suffer under ambiguities due to essential matrix decomposition. Referring to the mentioned publication, it is more robust against noise and yields better rotation accuracy compared to Stewenius' 5-point algorithm using directional cameras. Unfortunately there is no global solution for the translation (in case of full motion) which is recovered from 2-point correspondences, and hence does not obtain an optimized solution in case of noisy data. Further developments of global orientation estimates without RANSAC are published in [20, 285]. These mentioned novel approaches are possible candidates to be evaluated for robust two-view geometry estimation of omnidirectional cameras in future work.

6.6 Two-View Optimization

This section addresses the optimization of transformation estimates from the previous Section 6.5. Suitable error distances are described and their accuracies are compared under increasing image noise, similiar to Section 6.5.3. This following sections shortly describe known epipolar-based distances for directional and omnidirectional cameras from literature and also introduce two new projection-based distance approaches.

6.6.1 Epipolar-Based Error Distances

Given a set of point correspondences $\hat{\mathbf{X}}_k^i \leftrightarrow \hat{\mathbf{X}}_l^i$, $i = 1, \ldots, p$ and an initial transformation $[R, \hat{\mathbf{t}}]$ from two-view estimates, a non-linear least squares optimization

$$\underset{R,\hat{\mathbf{t}}}{\operatorname{argmin}} \sum_{i=1}^{p} \left\| \epsilon^i \right\|^2,$$

using LM and subjecting $\|\hat{\mathbf{t}}\| = 1$. For optimization purpose the rotation is expressed as Rodrigues vector.

Epipolar Distance

The epipolar distance from Eq. (6.12), page 95 is given by

$$\boxed{\epsilon^i_{\text{epi}} = \left(\hat{\mathbf{X}}^i_l\right)^T \left(\hat{\mathbf{t}} \times R\hat{\mathbf{X}}^i_k\right)}. \tag{6.19}$$

Geodesic Distance

The geodesic distance is an angular error between the epipolar plane and the corresponding point on unit sphere

$$\epsilon^i_{\text{geo}} = \sin^{-1}\left(\left(\hat{\mathbf{X}}^i_l\right)^T \left(\hat{\mathbf{t}} \times R\hat{\mathbf{X}}^i_k\right)\right). \tag{6.20}$$

as referred to [196, 195], which is based on the findings of [67, 66]. As can be seen for small error values the geodesic distance equals the epipolar distance. This is also confirmed by simulation data, where both distances yield the same results concerning transformation optimization. Since the geodesic distance needs more computational effort compared to the epipolar distance, it is not considered in this evaluation.

Sampsons Distance

In case of directional cameras the Sampson distance is the recommended error distance for epipolar geometry expressed in fundamental matrix form [96] as well as in essential matrix form [206]. As mentioned in [96], the Sampson error provides a first order approximation of the distance from an image point to its corresponding epipolar line (great circle in case of omnidirectional cameras). Interestingly, this is in contrast to [91], where the authors state that the Sampson error is the exact geometric distance from an image point to the first order approximation of the epipolar line (great circle). Either way, as reported in [196] the Sampson distance is not suitable for spherical image points and leads to worse results. Unfortunately an implementation of the Sampson distance for spherical image coordinates isn't presented. The author of this work couldn't find any publication that applies the Samspson distance to spherical image coordinates. In case of directional cameras, the Sampson distance lies on an image plane, whereas in case of omnidirectional cameras, it is measured on unit sphere, such that the error formulation would become

$$\epsilon^i_{\text{sam}} = \frac{\left(\left(\hat{\mathbf{X}}^i_l\right)^T \left(\hat{\mathbf{t}} \times R\hat{\mathbf{X}}^i_k\right)\right)^2}{\left\|\hat{\mathbf{t}} \times R\hat{\mathbf{X}}^i_k\right\|^2 + \left\|R^T[\hat{\mathbf{t}}]^T_\times \hat{\mathbf{X}}^i_l\right\|^2}. \tag{6.21}$$

Perpendicular Distance

The perpendicular distance

$$\epsilon^i_{\text{per}} = \frac{\left(\hat{\mathbf{X}}^i_l\right)^T \left(\hat{\mathbf{t}} \times R\hat{\mathbf{X}}^i_k\right)}{\left\|\hat{\mathbf{t}} \times R\hat{\mathbf{X}}^i_k\right\|} \tag{6.22}$$

is used in [196] for non-linear optimization and achieves good convergence properties concerning spherical camera coordinates.

6.6.2 Projection-Based Error Distances

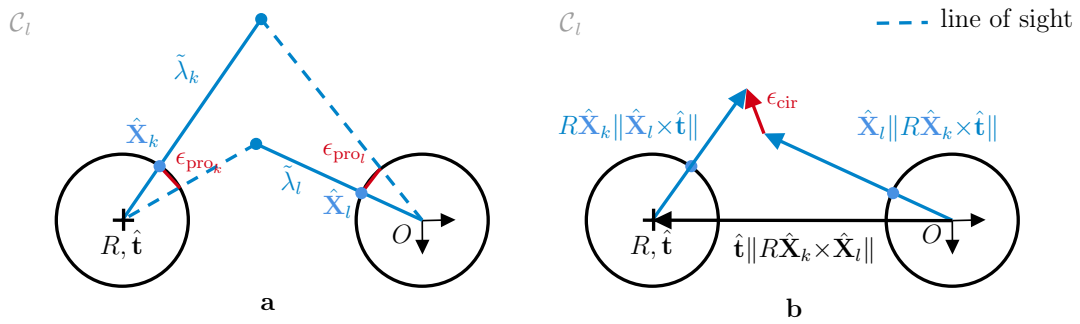

Fig. 6.10: Two-View optimization using **a)** projection error distance and **b)** circumferential error distance.

As can be seen, all forementioned distances are variations of the epipolar geometry and thus they are referred as epipolar-based distances. At this point two additional error distances are introduced as shown in Fig. 6.10, which rely on the derived closed-form projection constraints from Section 6.2.2, page 91, and haven't been presented in literature so far. They are referred as projection-based distances.

Projection Distance

Based on direct depth factor calculation from Eqs. (6.10) and (6.11), page 93 with:

$$\tilde{\lambda}_k^i = \frac{\|\hat{\mathbf{X}}_l^i \times \hat{\mathbf{t}}\|}{\|R\hat{\mathbf{X}}_k^i \times \hat{\mathbf{X}}_l^i\|} \tag{6.23}$$

$$\tilde{\lambda}_l^i = \frac{\|R\hat{\mathbf{X}}_k^i \times \hat{\mathbf{t}}\|}{\|R\hat{\mathbf{X}}_k^i \times \hat{\mathbf{X}}_l^i\|}, \tag{6.24}$$

the projection distance is given by

$$\epsilon_{\text{pro}}^i = \left\|\epsilon_{\text{pro}_k}^i\right\| + \left\|\epsilon_{\text{pro}_l}^i\right\| \tag{6.25}$$

consisting of two parts. The first one

$$\epsilon_{\text{pro}_k}^i = \hat{\mathbf{X}}_k^i - \frac{R^T\left(\hat{\mathbf{X}}_l^i \tilde{\lambda}_l^i - \hat{\mathbf{t}}\right)}{\left\|R^T\left(\hat{\mathbf{X}}_l^i \tilde{\lambda}_l^i - \hat{\mathbf{t}}\right)\right\|} \tag{6.26}$$

represents the distance between the i^{th} image point $\hat{\mathbf{X}}_k^i$ and its corresponding projection from $\hat{\mathbf{X}}_l^i$. The second one

$$\epsilon_{\text{pro}_l}^i = \hat{\mathbf{X}}_l^i - \frac{R\hat{\mathbf{X}}_k^i \tilde{\lambda}_k^i + \hat{\mathbf{t}}}{\left\|R\hat{\mathbf{X}}_k^i \tilde{\lambda}_k^i + \hat{\mathbf{t}}\right\|} \tag{6.27}$$

represents the distance between the i^{th} image point $\hat{\mathbf{X}}_l^i$ and its corresponding projection from $\hat{\mathbf{X}}_k^i$.

Circumferential Distance

The circumferential error is based on Eq. (6.5), page 92

$$0 = \hat{\mathbf{t}} + R\hat{\mathbf{X}}_k^i\tilde{\lambda}_k^i - \hat{\mathbf{X}}_l^i\tilde{\lambda}_l^i\,.$$

Applying Eqs. (6.23) and (6.24) and multiplying $\|R\hat{\mathbf{X}}_k^i \times \hat{\mathbf{X}}_l^i\|$ yields the circumferential distance

$$\boxed{\epsilon_{\text{cir}}^i = \hat{\mathbf{t}}\|R\hat{\mathbf{X}}_k^i \times \hat{\mathbf{X}}_l^i\| + R\hat{\mathbf{X}}_k^i\|\hat{\mathbf{X}}_l \times \hat{\mathbf{t}}\| - \hat{\mathbf{X}}_l^i\|R\hat{\mathbf{X}}_k^i \times \hat{\mathbf{t}}\|}\,. \tag{6.28}$$

6.6.3 Comparison between Error Distances

Based on the findings from Section 6.5, page 101 non-linear optimization uses transformation estimates provided by the *classic* 8-point algorithm. The following evaluation considers all motion and structure cases as presented in Fig. 6.8, page 102. The simulation also uses the parameter settings for image noise $\mathcal{N}(0, \sigma_{\theta,\phi}^2)$ with $\sigma_{\theta,\phi} = 0°, \ldots, 1°$ and performs 1000 iterations per noise level and per case, where at each single iteration 1000 point correspondences are sampled. The results of this simulation are shown in Fig. 6.11 for full motion and in Fig. 6.12 for degenerate motion.

As can be seen in **case a**, the circumferential distance achieves the best optimization results concerning rotation and translation. In contrast to other distances it yields a low translation direction error under increasing image noise. In **case b** the circumferential distance obtaines slightly worse (but sill satifiying) rotation results compared to epipolar-based distances, but provides significantly better translation results. In **case c** all optimization distances improve initial transformation estimates but still yield unsatisfying results, even under noiseless condition.

For all full motion cases depth values are recalculated (using Eqs. (6.8) and (6.9), page 93) from optimized transformations in order to validate cheirality (3^{rd} row). As can be clearly seen the circumferential distance maintains cheirality best. It obtains optimized transformations by enforcing positive depth values. The projection distance also consists of a cheirality constraint, however with increasing image noise it cannot enforce positive depth values for all point correspondences. This circumstance can be explained with the sampling distribution of points on unit sphere, where some points are located close to the epipoles. These points quickly tend to change the sign of the corresponding depth values due to noise influcence [157]. The proposed direct depth calculation uses a least squares approximation based on collinear constraints. Consequently small image deviations might lead to negative depth estimates even if the projection cheirality constraint isn't violated.

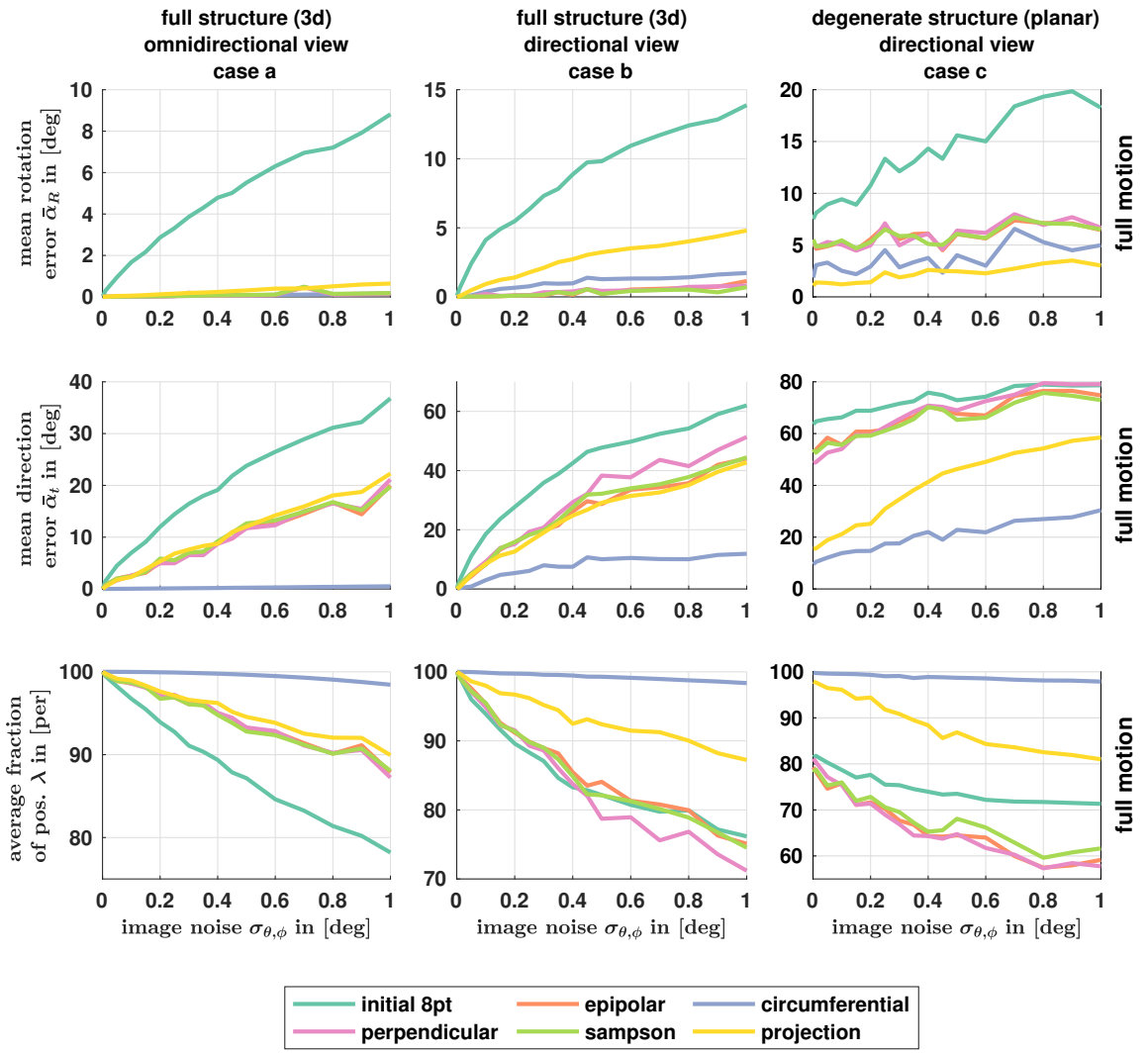

Fig. 6.11: Comparison of different cost functions for two-view transformation optimization (full motion).

Having a look at degenerate motion cases shows in **case d** that all distances yield similiar satisfying results concerning rotation. **Case e, f** are dominated by epipolar-based distances yielding continuous low rotation errors over increasing image noise in **case e** and slowly increasing rotation errors in **case f**.

All epipolar-based distances achieve similiar optimization results concerning full and degenerate motion cases. The use of the epipolar distance is preferred, due to its simplicity and decreased computational cost compared to prependicular distance and Sampson distance.

Based on the obtained results an optimization strategy is chosen that uses the circumferential distance to optimize transformation estimates in a first step, which are refined by a simple epipolar distance in a second step. The relation between both constraints is explained in Appendix A.6, page 179. The combination of projection

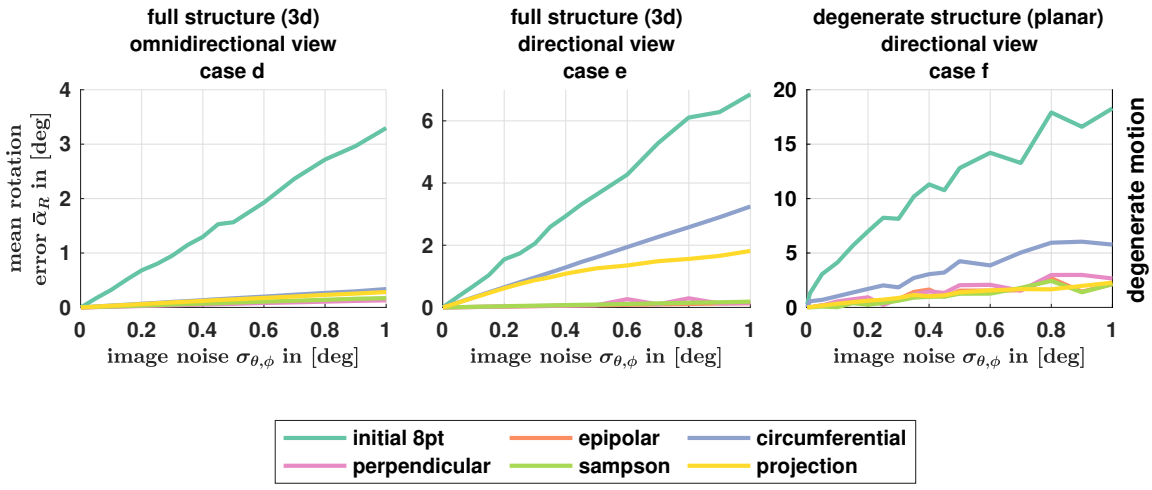

Fig. 6.12: Comparison of different cost functions for two-view transformation optimization (degenerate motion).

and epipolar constraints yields satisfying rotation and translation results in full motion cases and satisfying rotation results in degenerate motion cases as Fig. 6.13 shows. Using this proposed optimization scheme allows to further improve transformation results from circumferential distance except for **case c**, which however is not in scope of this work since degenerate structure relates to special field of interest.
The same can be said for **case d-f**, which are rotation-only transformations and do not represent a basis for 3d reconstruction. However it is still necessary to see how accurate transformations can be estimated under degeneracy in order to detect those cases.

Fig. 6.14 concentrates on **case a, b** and illustrates the rotation error $\Delta_{\gamma,\beta,\alpha}$ in Euler angles and the direction error as relative translation error $\Delta_{X,Y,Z}$. For simulation purposes camera transformations are equally sampled over all rotation angles (γ, β, α) and translation directions (X, Y, Z). The rotation standard derivation is denoted by $\sigma_{\gamma,\beta,\alpha}$ and the standard deviation concerning relative translation is denoted by $\sigma_{X,Y,Z}$.

In **case a** the mean rotation error increases to $\bar{\Delta}_{\gamma,\beta,\alpha} \approx 0.06°$ at $\sigma_{\theta,\phi} = 1°$, where the upper bound reaches $\bar{\Delta}_{\gamma,\beta,\alpha} + \sigma_{\gamma,\beta,\alpha} \approx 0.14°$. At the same time the mean relative translation error reaches $\bar{\Delta}_{X,Y,Z} \approx 0.24\%$ at $\sigma_{\theta,\phi} = 1°$ with an upper bound of $\bar{\Delta}_{X,Y,Z} + \sigma_{X,Y,Z} \approx 0.43\%$. This leads to the clear conclusion that transformation estimation based on omnidirectional distributed image features is less influenced by image noise and yields small relative transformation errors.
Optimization results in **case b** yield larger transformation errors such that mean rotation error is $\bar{\Delta}_{\gamma,\beta,\alpha} \approx 0.24°$ at $\sigma_{\theta,\phi} = 1°$ with a corresponding upper bound $\bar{\Delta}_{\gamma,\beta,\alpha} + \sigma_{\gamma,\beta,\alpha} \approx 0.78°$. The mean relative translation error reaches $\bar{\Delta}_{X,Y,Z} \approx 5\%$ at $\sigma_{\theta,\phi} = 1°$, where the upper bound is $\bar{\Delta}_{X,Y,Z} + \sigma_{X,Y,Z} \approx 21\%$. The relative translation error is more than a magnitude larger in the directional view case compared to the omnidirectional one. Hence it can be said that translation estimation in directional

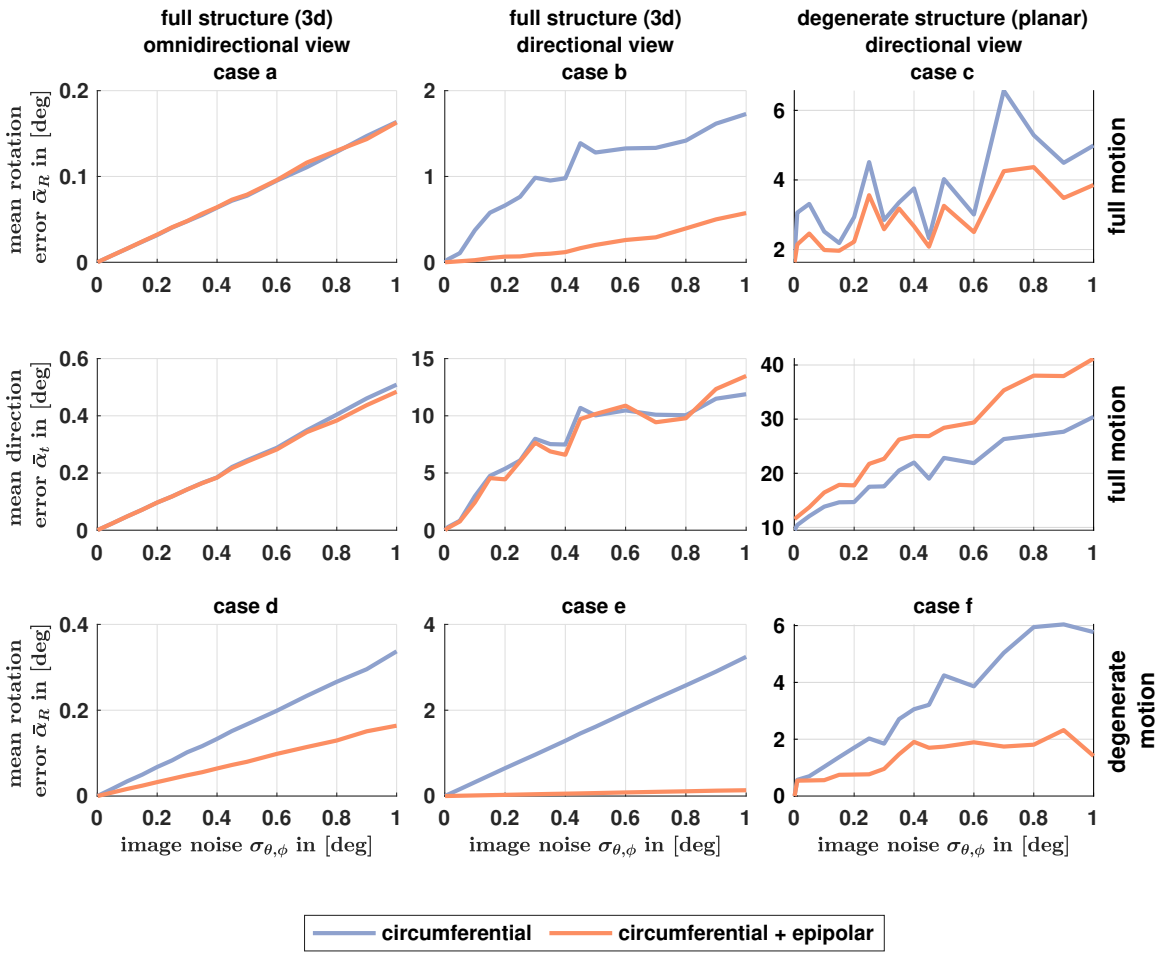

Fig. 6.13: Optimization results using a combination of circumferential and epipolar error distance in a two step refinement.

view cases is strongly influenced by image noise.

6.7 Two-View Translation Scaling

The relative pose between two cameras from optimized two-view geometry consists of an up-to-scale translation $\|\hat{\mathbf{t}}\| = 1$. The aim of this section is to use available depth data, e.g. from range sensors to correctly scale the translation's magnitude to real-world dimension without additionally influencing the translation's direction through depth inaccuracies. There are different methods to obtain a scaled real-world transformation between a pair of combined color and depth data such as ICP or PnP.
ICP is a 3d-3d alignment and requires depth data in both images. This restriction prevents the combination of RGB- and RGBD-cameras. PnP performs a 3d-2d alignment (Section 4.4.1,page 60) and requires depth information for all feature points in at least one of the two cameras. This is not always the case, since some image features are located outside the depth range or outside the FoV of the depth sensor. In most RGBD-cameras the depth sensor covers a smaller FoV than the color sensor does as

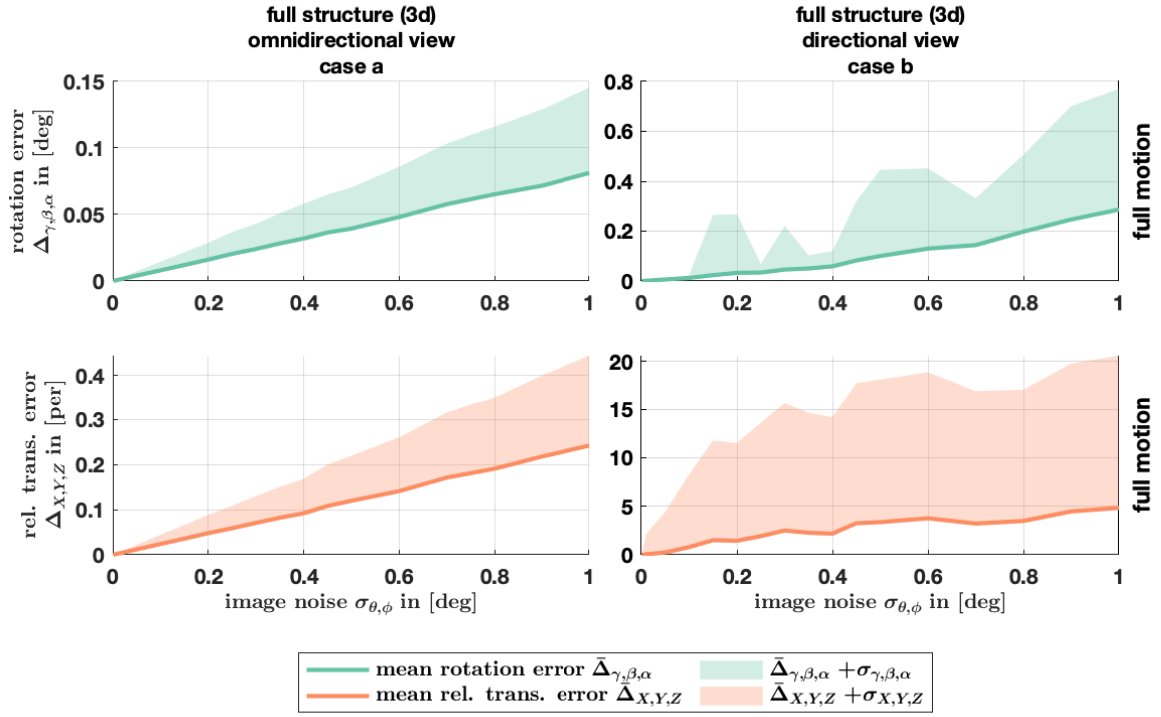

Fig. 6.14: Optimization results from proposed two step refinement showing mean rotation error $\bar{\Delta}_{\gamma,\beta,\alpha}$ and mean relative translation error $\bar{\Delta}_{X,Y,Z}$ over increasing image noise $\sigma_{\theta,\phi}$.

can be seen at hand of the calibration example from Appendix A.1, page 170. Hence, only a subset of all available feature correspondences can be used for PnP alignment. This circumstance may cause drift, if feature correspondences with depth values are not equally distributed over the entire image.
Some methods also classify features according their depth values to split rotation from translation estimation. Features at short distance are better for translation estimation, wheras features that are farther away achieve a better and more robust rotation estimation [126]. In order to overcome these mentioned problems, 2d-2d image alignment via two-view geometry is recommended [218]. Available depth data are only used to scale the translation magnitude.
The following describes a least squares estimation approach of the real-world translation scale factor and its non-linear optimization.

6.7.1 Linear Least Squares Estimation

The here presented approach is based on findings from Section 6.2.2, page 91 that directly obtain depth values from feature correspondences of a two-view transformation. The real-world translation scale factor is estimated by minimizing the sum of Euclidean distances between given depth values and recalculated ones as comprehensively shown in the following.
Assuming two cameras k and l with known exerior relative transformation $[R_l, \mathbf{t}_l] = [I, \mathbf{0}]$, $[R_k, \mathbf{t}_k] = [R, \mathbf{t}]$ and a set of point correspondences $\hat{\mathbf{X}}_k \leftrightarrow \hat{\mathbf{X}}_l$. Considering the circumstance that a point pair has only one corresponding depth value in either camera

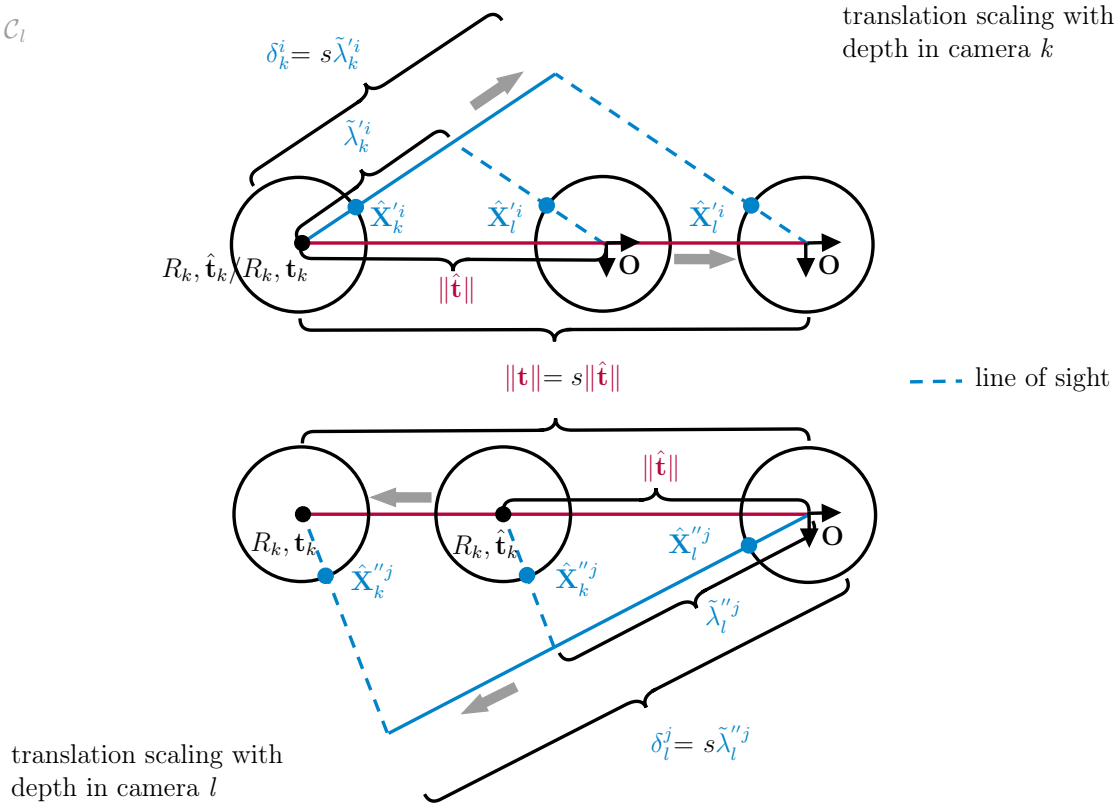

Fig. 6.15: Basic principle of the two-view real-world translation scale estimation

k or camera l. $\hat{\mathbf{X}}_k^{\prime i} \leftrightarrow \hat{\mathbf{X}}_l^{\prime i}$ denotes a point pair subset corresponding to m given depth values δ_k^i in camera k and analogous $\hat{\mathbf{X}}_k^j \leftrightarrow \hat{\mathbf{X}}_l^{\prime\prime j}$ is a point pair subset corresponding to n given depth values δ_l^j in camera l. Supposing noise free data and a real-world scale translation $\mathbf{t}$, such that $\lambda_k^{\prime i} = \delta_k^i$ and $\lambda_l^{\prime\prime j} = \delta_l^j$. Applying mentioned constraints to Eqs. (6.6) and (6.7), page 92 yields:

$$\hat{\mathbf{X}}_l^{\prime i} \times \mathbf{t} = \delta_k^i \left(R\hat{\mathbf{X}}_k^{\prime i} \times \hat{\mathbf{X}}_l^{\prime i} \right), \quad i = 1, \ldots, m$$
$$R\hat{\mathbf{X}}_k^{\prime\prime j} \times \mathbf{t} = \delta_l^j \left(R\hat{\mathbf{X}}_k^{\prime\prime j} \times \hat{\mathbf{X}}_l^{\prime\prime j} \right), \quad j = 1, \ldots, n\,.$$

Substituting $\mathbf{t} = s\hat{\mathbf{t}}$, with s being an unknown real-world scale factor and $\hat{\mathbf{t}}$ being the normalized translation from two-view geometry leads to:

$$\underbrace{s\left(\hat{\mathbf{X}}_l^{\prime i} \times \hat{\mathbf{t}}\right)}_{\mathbf{a}_k^i} = \underbrace{\delta_k^i \left(R\hat{\mathbf{X}}_k^{\prime i} \times \hat{\mathbf{X}}_l^{\prime i} \right)}_{\mathbf{b}_k^i}$$
$$\underbrace{s\left(R\hat{\mathbf{X}}_k^{\prime\prime j} \times \hat{\mathbf{t}}\right)}_{\mathbf{a}_l^j} = \underbrace{\delta_l^j \left(R\hat{\mathbf{X}}_k^{\prime\prime j} \times \hat{\mathbf{X}}_l^{\prime\prime j} \right)}_{\mathbf{b}_l^j}\,.$$

Stacking all elements of $\mathbf{a}_k^i, \mathbf{a}_l^j$ and $\mathbf{b}_k^i, \mathbf{b}_l^j$ such that:

$$\mathbf{a} = \left((\mathbf{a}_k^1)^T, \ldots, (\mathbf{a}_k^m)^T, (\mathbf{a}_l^1)^T, \ldots, (\mathbf{a}_l^n)^T \right)^T \in \mathbb{R}^{3(m+n)\times 1}$$
$$\mathbf{b} = \left((\mathbf{b}_k^1)^T, \ldots, (\mathbf{b}_k^m)^T, (\mathbf{b}_l^1)^T, \ldots, (\mathbf{b}_l^n)^T \right)^T \in \mathbb{R}^{3(m+n)\times 1}\,,$$

allows straightforwardly to obtain s using least-squares approximation

$$\boxed{s_{\text{init}} = (\mathbf{a}^T\mathbf{b})/\|\mathbf{a}\|^2}. \tag{6.29}$$

Fig. 7.4 highlights the basic idea behind the proposed scaling estimation approach and shows that scaling $\delta_k^i = \tilde{\lambda}_k'^i s$ and / or $\delta_l^j = \tilde{\lambda}_l''^j s$ also scales $\mathbf{t} = s\hat{\mathbf{t}}$.

Appendix A.9, page 182 describes an approach to obtain the real-world scale factor's uncertainty from given linear estimation.

6.7.2 Non-Linear Least Squares Optimization

Real-world translation scale factor optimization is based on projection constraints. It is a common method that uses depth values in order to back-project corresponding image points from camera sphere $\mathcal{S}_l$ onto camera sphere $\mathcal{S}_k$ and vice versa. In this scenario $\hat{\mathbf{X}}_k'^i$ denotes a measured image point on camera sphere $\mathcal{S}_k$ and $\hat{\mathbf{Y}}_k^i$ represents its corresponding back-projection from camera sphere $\mathcal{S}_l$ such that

$$\hat{\mathbf{Y}}_k^i = \frac{\mathbf{Y}_k^i}{\|\mathbf{Y}_k^i\|} = \frac{\delta_l^i R^T \hat{\mathbf{X}}_l'^i - sR^T\hat{\mathbf{t}}}{\|\delta_l^i R^T \hat{\mathbf{X}}_l'^i - sR^T\hat{\mathbf{t}}\|}, \quad i = 1, \ldots, m\,. \tag{6.30}$$

Analogous $\hat{\mathbf{X}}_l'^j$ denotes a measured image point on camera sphere $\mathcal{S}_l$ and $\hat{\mathbf{Y}}_l^j$ represents its corresponding back-projection from camera sphere $\mathcal{S}_k$

$$\hat{\mathbf{Y}}_l^j = \frac{\mathbf{Y}_l^j}{\|\mathbf{Y}_l^j\|} = \frac{\delta_k^j R \hat{\mathbf{X}}_k''^j + s\hat{\mathbf{t}}}{\|\delta_k^j R \hat{\mathbf{X}}_k''^j + s\hat{\mathbf{t}}\|}, \qquad j = 1, \ldots, n\,. \tag{6.31}$$

Stacking all measurements into $\hat{\mathbf{X}}$ and their corresponding back-projections into $\hat{\mathbf{Y}}$ such that:

$$\begin{aligned}\hat{\mathbf{X}} &= \left(\hat{\mathbf{X}}_k'^1, \ldots, \hat{\mathbf{X}}_k'^m, \hat{\mathbf{X}}_l''^1, \ldots, \hat{\mathbf{X}}_l''^n\right)^T \in \mathbb{R}^{3(m+n)\times 1} \\ \hat{\mathbf{Y}} &= \left(\hat{\mathbf{Y}}_k^1, \ldots, \hat{\mathbf{Y}}_k^m, \hat{\mathbf{Y}}_l^1, \ldots, \hat{\mathbf{Y}}_l^n\right)^T \in \mathbb{R}^{3(m+n)\times 1}\,.\end{aligned}$$

The real-world translation scale factor is determined by minimizing the sum of squared Euclidean distances between measured image points $\hat{\mathbf{X}}$ and their corresponding projections $\hat{\mathbf{Y}}$

$$\boxed{s_{\text{opt}} = \min_s \|\hat{\mathbf{X}} - \hat{\mathbf{Y}}\|^2}, \tag{6.32}$$

which is solved by LM.
Appendix A.10, page 183 describes an approach to obtain the scaling factor's uncertainty from given non-linear optimization.

6.7.3 Comparison between Initial and Optimized Scaling Factor

Assuming s_{true} to be the ground truth scaling factor as well as s_{init} being the initial scaling factor from Eq. (6.29) and s_{opt} beeing the optimized scaling factor obtained via Eq. (6.32). The relative scaling error

$$\Delta_s = \left| \frac{s_{\text{init/opt}} - s_{\text{true}}}{s_{\text{true}}} \right| \tag{6.33}$$

is calculated for the omnidirectional view **case a** as well as for the directional view **case b**. This section analyzes the influence of different noise sources on translation scaling such as image noise $\mathcal{N}(0, \sigma^2_{\theta,\phi})$ with $\sigma_{\theta,\phi} = 0°, \ldots, 1°$, rotation noise $\mathcal{N}(0, \sigma^2_{\gamma,\beta,\alpha})$ with $\sigma_{\gamma,\beta,\alpha} = 0°, \ldots, 1°$, relative translation noise $\mathcal{N}(0, \sigma^2_{X,Y,Z})$ with $\sigma_{X,Y,Z} = 0\%, \ldots, 10\%$ and relative depth noise $\mathcal{N}(0, \sigma^2_{\delta})$ with $\sigma_{\delta} = 0\%, \ldots, 1\%$.

At every noise level, 15000 samples with 100 point correspondences each are randomly generated. The results are shown in Fig. 6.16. Image noise $\sigma_{\theta,\phi}$ simulates calibration and feature detection inaccuracies, rotation noise $\sigma_{\gamma,\beta,\alpha}$ as well as translation noise $\sigma_{X,Y,Z}$ cover erroneous two-view estimates and depth noise σ_{δ} represents measuring uncertainties of range sensors. Each noise source is evaluated separately, the remaining ones are set to zero. An evaluation of all noise sources at once would lead to extensive computational effort and an overall evaluation is not straightforward. Hence the conclusion is made from the separate evaluations.

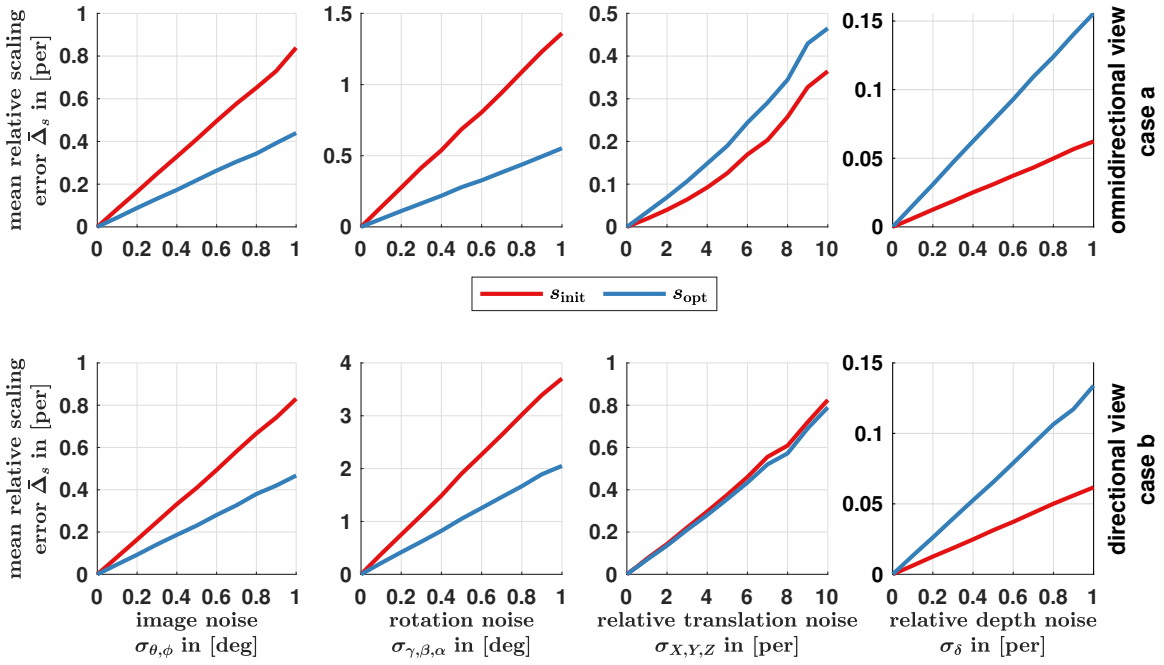

Fig. 6.16: Overview of the mean relative translation scaling error $\bar{\Delta}_s$ caused by image noise $\sigma_{\theta,\phi}$, rotation noise $\sigma_{\gamma,\beta,\alpha}$, translation noise $\sigma_{X,Y,Z}$ and depth noise σ_{δ} for omnidirectional view (1st row, **case a**) and directional view (2nd row, **case b**).

Non-linear optimization via projection constraints descreases the relative mean scaling error $\bar{\Delta}_s$ in case of image noise at $\sigma_{\theta,\phi} = 1°$ from $\bar{\Delta}_s \approx 0.8\%$ to $\bar{\Delta}_s \approx 0.4\%$ for both, omnidirectional and directional view cases. $\bar{\Delta}_s$ also decreases under rotation noise from $\bar{\Delta}_s \approx 1.4\%$ to $\bar{\Delta}_s \approx 0.6\%$ at $\sigma_{\gamma,\beta,\alpha} = 1°$ for omnidirectional view case and from $\bar{\Delta}_s \approx 3.8\%$ to $\bar{\Delta}_s \approx 2.1\%$ for directional view case. Interestingly the proposed optimization slighly increases $\bar{\Delta}_s$ in case of translation noise from $\bar{\Delta}_s \approx 0.36\%$ to $\bar{\Delta}_s \approx 0.46\%$ at $\sigma_{X,Y,Z} = 10\%$ for omnidirectional view case. In constrast to that, $\bar{\Delta}_s$ negligibly decreases from $\bar{\Delta}_s \approx 0.81\%$ to $\bar{\Delta}_s \approx 0.80\%$ at $\sigma_{X,Y,Z} = 10\%$ for directional view case. Considering $\bar{\Delta}_s$ in relation to depth noise clearly shows a decrease from $\bar{\Delta}_s \approx 0.16\%$ to $\bar{\Delta}_s \approx 0.06\%$ at $\sigma_\delta = 1\%$ for both, omnidirectional and directional view. As a result for both view cases, rotation noise turns out to have the strongest influence on the real-world scale factor calculation, approximately an order of magnitude larger than the remaining noises. Non-linear optimization is recommended, since $\bar{\Delta}_s$ decreases in case of rotation noise, which compensates the $\bar{\Delta}_s$ increase in case of depth noise. Translation noise influence is small and doesn't show significant differences between initial and optimized real-world scale factors. Image noise leads to small intial mean relative scale errors, that can be further halfed by non-linear optimization.

Real World Example

The proposed translation scaling is a flexible method to adjust relative translation to real world scale and needs at least one depth value in one of the two images. Fig. 6.17 shows feature matches and their corresponding projection in the 3d scene.

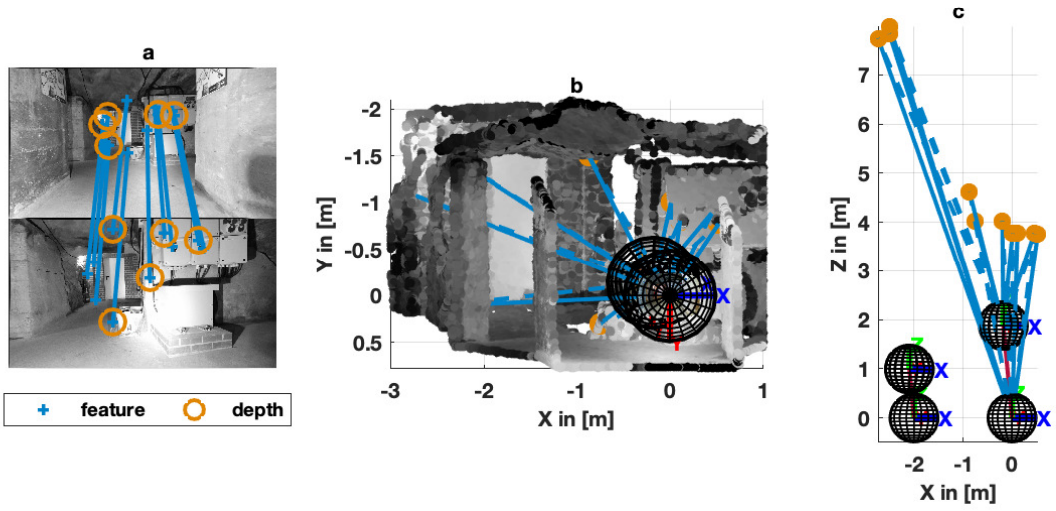

Fig. 6.17: Real world example using the proposed two-view scaling. Solid lines (—) denote back-projections, dashed lines (- - -) denote their corresponding projections onto the other sphere. **a)** Feature matches with highlighted depth values ◯. **b)** Front view of the reconstructed 3d scene with both camera spheres. **c)** Top view of the 3d scene showing that scaling influences the distance between both spheres but not the direction.

6.8 Homography to Identify Degeneracies

Identifying degenerate motion is an important part in every reconstruction pipeline since this kind of motion doesn't provide any 3d information. Furthermore, degenerate structure can only be described by homography and leads to erroneous transformations when solved via epipolar or projection constraints. Some Visual SLAM implementations [189, 190, 27] are able to handle degenerate structures and recover the transformation from a homography matrix. However the proposed SCME pipeline does not incorporate degenerate structure.

Degenerate motion can be identified when resolving the ambiguity from essential matrix decomposition as explained in Sections 6.4.2 to 6.4.4. The improved procedure (Section 6.4.4) needs two calculation steps only, however pure rotation in case of noisy data leads to the circumstance that $\tilde{\lambda}_k \approx 0$ and requires a threshold. Since $\tilde{\lambda}_k$ is an approximate depth value, it is up-to-scale and makes thresholding cumbersome. Obtaining the scaled depth value $\lambda_k = \tilde{\lambda}_k / \left\| R\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l \right\|$ leads to the problem that $\left\| R\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l \right\| \to 0$ in case of pure rotation and thus generates a large λ_k. Consequently large depth values would indicate a pure rotation. However a full motion using distant features also generates large depth values. Hence, depth thresholding would limit the usage of distant feature points.

In [24, 285] the authors propose $\left\| R\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l \right\| = 0$ as decision criterion to identify pure rotation, which is also used for the standard procedure (Section 6.4.2). However this pure rotation constraint is only valid under the assumption that the recovered rotation from essential matrix decomposition yields sufficent accuracy. As illustrated in Fig. 6.9, page 105, under increasing image noise the extracted rotation for degnerate motion becomes less accurate (**case d-f**) and leads to similiar rotation errors as for full motion (**case a, b**). Thus a decision based on the pure rotation constraint cannot be made under increased image noise. The essential matrix is able to describe the relation of point correspondences under pure rotation, however the extracted rotation matrix does not necessarly yield satifying accuracy to describe a valid transformation between the point correspondences.

Degenerate motion doesn't deliver 3d motion information and degenerate structure is not fully described by introduced two-view relations from Section 6.6. Hence both types of degeneracies must be identified and rejected from the set of valid image pairs.

6.8.1 Homography for Spherical Cameras

The following describes the homography geometry applied to camera spheres in camera frame $\mathcal{C}_l$. Assuming a 3d point $\mathbf{U}$ on a plane $\boldsymbol{\pi}_\mathrm{h}$. The homography matrix H describes

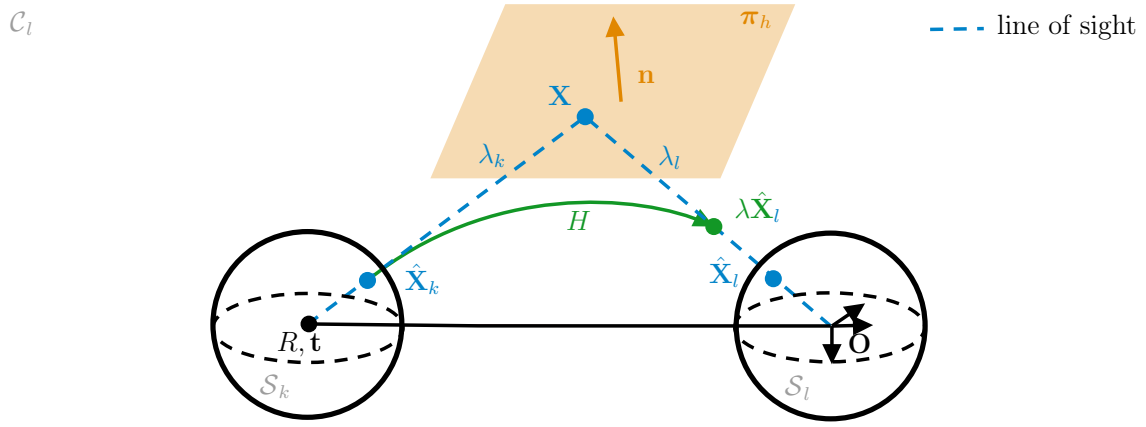

Fig. 6.18: Graphical representation of the homography geometry

the relation between the corresponding projections such that

$$\lambda_l \hat{\mathbf{X}}_l = \lambda_k H \hat{\mathbf{X}}_k \tag{6.34}$$

with $\lambda_l/\lambda_k = \tilde{\lambda}$ leading to

$$\tilde{\lambda}\hat{\mathbf{X}}_l = H\hat{\mathbf{X}}_k\,. \tag{6.35}$$

Eliminating $\tilde{\lambda}$ by cross-multiplying $\hat{\mathbf{X}}_l$ yields

$$\mathbf{0} = \hat{\mathbf{X}}_l \times H\hat{\mathbf{X}}_k \tag{6.36}$$

As Fig. 6.18 illustrates, homography applied to spherical coordinates maps $\hat{\mathbf{X}}_k$ to a point $\tilde{\lambda}\hat{\mathbf{X}}_l$ on the LoS between $\mathbf{X}$ and $\hat{\mathbf{X}}_l$. Since the sign of λ is arbitrary, normalizing $\lambda\hat{\mathbf{X}}_l/\|\lambda\hat{\mathbf{X}}_l\|$ could lead to a projection $-\hat{\mathbf{X}}_l$ onto the opposite hemisphere.
According to [217] the essential matrix can be decomposed

$$H = R + \tilde{\mathbf{t}}\mathbf{n}^T\,, \tag{6.37}$$

where $\mathbf{n}$ denotes the plane's normal vector and $\tilde{\mathbf{t}}$ denotes the up-to-scale translation vector. However in this work a decomposition of the homography matrix is not required as shown in the following.

6.8.2 Homography Estimation

The homography matrix is estimated using a DLT and a set of at least five ($p \geq 5$) point correspondences $\mathbf{X}_k^i \leftrightarrow \mathbf{X}_l^i$, $i = 1, \ldots, p$. Based on Eq. (6.36) the homography constraint can be rearranged into the form $B\mathbf{x} = \mathbf{0}$ with

$$\mathbf{x} = \begin{pmatrix} h_{11} & h_{12} & h_{13} & h_{21} & h_{22} & h_{23} & h_{31} & h_{32} & h_{33} \end{pmatrix}^T \in \mathbb{R}^{1\times 9} \tag{6.38}$$

containing all matrix elements

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \in \mathbb{R}^{3\times 3} \tag{6.39}$$

and

$$\begin{aligned} \mathbf{b}_1^i &= \begin{pmatrix} \mathbf{X}_{k(1)}^i \mathbf{X}_{l(1)}^i & \mathbf{X}_{k(2)}^i \mathbf{X}_{l(1)}^i & \mathbf{X}_{k(3)}^i \mathbf{X}_{l(1)}^i \end{pmatrix} \in \mathbb{R}^{1\times 3} \\ \mathbf{b}_2^i &= \begin{pmatrix} \mathbf{X}_{k(1)}^i \mathbf{X}_{l(2)}^i & \mathbf{X}_{k(2)}^i \mathbf{X}_{l(2)}^i & \mathbf{X}_{k(3)}^i \mathbf{X}_{l(2)}^i \end{pmatrix} \in \mathbb{R}^{1\times 3} \\ \mathbf{b}_3^i &= \begin{pmatrix} \mathbf{X}_{k(1)}^i \mathbf{X}_{l(3)}^i & \mathbf{X}_{k(2)}^i \mathbf{X}_{l(3)}^i & \mathbf{X}_{k(3)}^i \mathbf{X}_{l(3)}^i \end{pmatrix} \in \mathbb{R}^{1\times 3} \end{aligned}$$

forming

$$B^i = \begin{bmatrix} \mathbf{0}_{1\times 3} & -\mathbf{b}_3^i & \mathbf{b}_2^i \\ \mathbf{b}_3^i & \mathbf{0}_{1\times 3} & -\mathbf{b}_1^i \\ -\mathbf{b}_2^i & \mathbf{b}_1^i & \mathbf{0}_{1\times 3} \end{bmatrix} .$$

Since $\operatorname{rank}(B) = 2$ [164, 50] it can be reduced such that

$$B^i = \begin{bmatrix} \mathbf{0}_{1\times 3} & -\mathbf{b}_3^i & \mathbf{b}_2^i \\ \mathbf{b}_3^i & \mathbf{0}_{1\times 3} & -\mathbf{b}_1^i \end{bmatrix} .$$

Stacking all available point correspondences into B leads to

$$B = \begin{bmatrix} \mathbf{0}_{1\times 3} & -\mathbf{b}_3^1 & \mathbf{b}_2^1 \\ \mathbf{b}_3^1 & \mathbf{0}_{1\times 3} & -\mathbf{b}_1^1 \\ \vdots & \vdots & \vdots \\ \mathbf{0}_{1\times 3} & -\mathbf{b}_3^p & \mathbf{b}_2^p \\ \mathbf{b}_3^p & \mathbf{0}_{1\times 3} & -\mathbf{b}_1^p \end{bmatrix} \in \mathbb{R}^{2p\times 9} .$$

Applying an SVD to B with

$$B = U\Sigma V^T$$

obtains the right unitary matrix $V \in \mathbb{R}^{9\times 9}$. The last column $V_{(:,9)}$ corresponds to the smallest singular value in $\Sigma \in \mathbb{R}^{9\times 9}$, which represents the solution $\mathbf{x} = V_{(:,9)}$. Before applying the SVD, normalizing each row of B leads to a more stable numerical solution. Rearranging the elements of $\mathbf{x}$ provides H.

6.8.3 Homography Optimization

Homography optimization is based on Eq. (6.35)

$$\tilde{\lambda}\hat{\mathbf{X}}_l = H\hat{\mathbf{X}}_k ,$$

which is solved for λ in least squares sense (Appendix A.2,page 174) such that

$$\tilde{\lambda} = \hat{\mathbf{X}}_l^T \left(H \hat{\mathbf{X}}_k \right) . \tag{6.40}$$

Applying Eq. (6.40) to Eq. (6.35) yields the cost function

$$\underset{H}{\operatorname{argmin}} \sum_{i=1}^{p} \left\| \hat{\mathbf{X}}_l - \frac{H \hat{\mathbf{X}}_k}{\hat{\mathbf{X}}_l^T \left(H \hat{\mathbf{X}}_k \right)} \right\|^2 .$$

subjecting $\|H\|_{\mathrm{F}} = 1$ for non-linear optimization using LM.

6.8.4 Homography and Pure Rotation

Assuming a point on a plane $\boldsymbol{\pi}_{\mathrm{h}}$ at infinity such that $\lambda_k, \lambda_l \to \infty$ and thus $\tilde{\lambda} = 1$. As a result for spherical camera coordinates Eq. (6.35) becomes

$$\hat{\mathbf{X}}_l = H \hat{\mathbf{X}}_k ,$$

which is similar to $\hat{\mathbf{X}}_l = R \hat{\mathbf{X}}_k$. Hence the homography for points on a plane $\boldsymbol{\pi}_{\mathrm{h}}$ at infinity describes a pure rotation and consequently point correspondences from pure camera rotation relate to homography. The latter circumstance is especially used for image stichting.
Generally speaking, H describes the transformation caused by pure rotation or a moving camera capturing a planar structure [96, 224]. Estimating and optimizing H solves for the transformation from degenerate structure (point on plane $\boldsymbol{\pi}_{\mathrm{h}}$) and from degenerate motion (pure rotation). As a consequence, in [224] the authors destinguish between homography from pure rotation and from planar structure in order to reject pure rotation only.

6.8.5 Homography in Epipolar Geometry

The *classic* 8-point algorithm is able to obtain an essential matrix E that mathematically solves the epipolar geometry under degeneracies. Referring to Section 6.5, page 101 the decomposition of E delivers unsatisfying two-view transformation results for degenerate structure (**case c**) and degenerate motion (**case d-f**).

The *classic* 8-point algorithm uses a set of eight point correspondences to obtain the essential matrix E. Using the same point set and following Section 6.8.2 delivers a homography matrix H. In case of degenerate motion/structure $(\hat{\mathbf{X}}_l)^T E \hat{\mathbf{X}}_k = 0$ and $\|\hat{\mathbf{X}}_l \times H \hat{\mathbf{X}}_k\| \approx 0$, whereas in case of full motion/structure $(\hat{\mathbf{X}}_l)^T E \hat{\mathbf{X}}_k = 0$ and $\|\hat{\mathbf{X}}_l \times H \hat{\mathbf{X}}_k\| > 0$. As can be seen the homography error might be a useful indicator. Hower it is not straightforward to find a threshold that distinguishes between $\|\cdot\| \approx 0$ and $\|\cdot\| > 0$.

The following shows a new strategy by comparing two epipolar error distances in order identify degenerate cases. This startegy can be used in conjunction with essential matrix estimation in a RANSAC framework.
Essential matrix E and homography matrix H are able to transform point correspondences under structure and motion degeneracies such that:

$$0 = \hat{\mathbf{X}}_l^T E \hat{\mathbf{X}}_k \tag{6.41}$$

$$\tilde{\lambda}\hat{\mathbf{X}}_l = H\hat{\mathbf{X}}_k\,. \tag{6.42}$$

Rearranging Eq. (6.42) leads to

$$\hat{\mathbf{X}}_l = \frac{1}{\tilde{\lambda}} H \hat{\mathbf{X}}_k \tag{6.43}$$

and entering Eq. (6.43) into Eq. (6.41) yields:

$$\begin{aligned} 0 &= \frac{1}{\tilde{\lambda}} \left(H\hat{\mathbf{X}}_k\right)^T E \hat{\mathbf{X}}_k \\ 0 &= \frac{1}{\tilde{\lambda}} \left(H\hat{\mathbf{X}}_k\right)^T E \hat{\mathbf{X}}_k \\ \boxed{0 &= \frac{1}{\tilde{\lambda}} \hat{\mathbf{X}}_k^T H^T E \hat{\mathbf{X}}_k}\,. \end{aligned} \tag{6.44}$$

Eq. (6.44) combines essential matrix E with homography matrix H and is only valid in degenerate cases. Applying $E = [\mathbf{t}]_\times R$ to Eq. (6.41) and to Eq. (6.44) leads to:

$$\epsilon_E = \left(\hat{\mathbf{X}}_l\right)^T \left([\mathbf{t}]_\times R \hat{\mathbf{X}}_k\right) \tag{6.45}$$

$$\epsilon_H = \frac{1}{\tilde{\lambda}} \left(\hat{\mathbf{X}}_k\right)^T H^T \left([\mathbf{t}]_\times R \hat{\mathbf{X}}_k\right) \tag{6.46}$$

with $|\tilde{\lambda}| = 1/\|H\hat{\mathbf{X}}_k\|$, where ϵ_E denotes the epipolar error based on E and ϵ_H denotes the epipolar error based on E and H.

In case of degeneracy the essential matrix decomposition into R and $\mathbf{t}$ becomes erroneous, which leads to larger epipolar errors $|(\hat{\mathbf{X}}_l)^T\left([\mathbf{t}]_\times R\hat{\mathbf{X}}_k\right)| \geq 0$. Furthermore, H is obtained such that $\|\hat{\mathbf{X}}_l \times H\hat{\mathbf{X}}_k\| = 0$ leading to the fact that $\epsilon_H = \epsilon_E$. Considering $\tilde{\lambda}\hat{\mathbf{X}}_l = H\hat{\mathbf{X}}_k$ from Eq. (6.42) shows that since $\|H\hat{\mathbf{X}}_k\| \geq 1$ and $\|\hat{\mathbf{X}}_l\| = 1$ consequently $\tilde{\lambda} \leq 1$. If $\tilde{\lambda}$ is brought to the left side of Eq. (6.46), the error becomes $\epsilon_H = \tilde{\lambda}\epsilon_H$ and thus $\epsilon_H \leq \epsilon_E$.
The epipolar errors are calculated for the entire point set p:

$$\epsilon_E = \sum_{i=1}^{p} \left\| \left(\hat{\mathbf{X}}_l^i\right)^T \left([\mathbf{t}]_\times R \hat{\mathbf{X}}_k^i\right) \right\| \tag{6.47}$$

$$\epsilon_H = \sum_{i=1}^{p} \left\| \left(\hat{\mathbf{X}}_k^i\right)^T H^T \left([\mathbf{t}]_\times R \hat{\mathbf{X}}_k^i\right) \right\|. \tag{6.48}$$

Comparing both errors indicates a degeneracy if $\epsilon_H \leq \epsilon_E$.

The described procedure can be also integrated into the optimization process of two-view transformations in order to verify the optimized results. This step is required if filtering rejects feature correspondences from the point set during optimization in order to ensure that the remaining correspondences still belong to full motion/structure. In that case it is recommended to additionally optimize H as Section 6.8.3 describes before calculating the epipolar errors.

Brief Chapter Summary

This chapter extensively described the geometric relations of point correspondences between two camera spheres. A closed-from triangulation approach was introduced, which is called *alternative* midpoint method. Different two-view estimation algorithms were presented and compared under varying camera motion and scene structure, where the *classic* 8-point algorithm obtained best results. Furthermore, a novel procedure was developed to resolve transformation ambiguities from essential matrix decomposition withing two calculation steps only. For two-view optimization popular epipolar-based distanced function were presented and novel projection-based ones were introduced. A combination of projection and epipolar constraints obtained sufficient optimization results. Additionally, up-to-scale two-view transformations were scaled using available depth data and a procedure was explained to detect degenerate motion and degenerate structure using homography.

7 Relations between Three Camera Spheres

Brief Chapter Overview

This chapter comprehensively examines the relation of point correspondences between three camera spheres. Section 7.1 describes in detail the derivation of the three-view constraint applied to camera spheres. Section 7.2 introduces a novel *crossing epipolar planes* geometry and Section 7.3 reformulates the trifocal geometry for spherical cameras. The relations between three-view, *crossing epipolar planes* and trifocal geometry are clarified in Section 7.4. Section 7.5 concentrates on the recovery of the translation ratio from up-to-scale two-view transformations. Section 7.5.1 describes a structureless approach to determine the translation ratio from three-view, *crossing epipolar planes* and trifocal constraints, whereas Section 7.5.2 presents a structure-based approach using triangulation. Finally Section 7.5.3 compares derived translation ratios from discussed approaches.

7.1 Three View Geometry

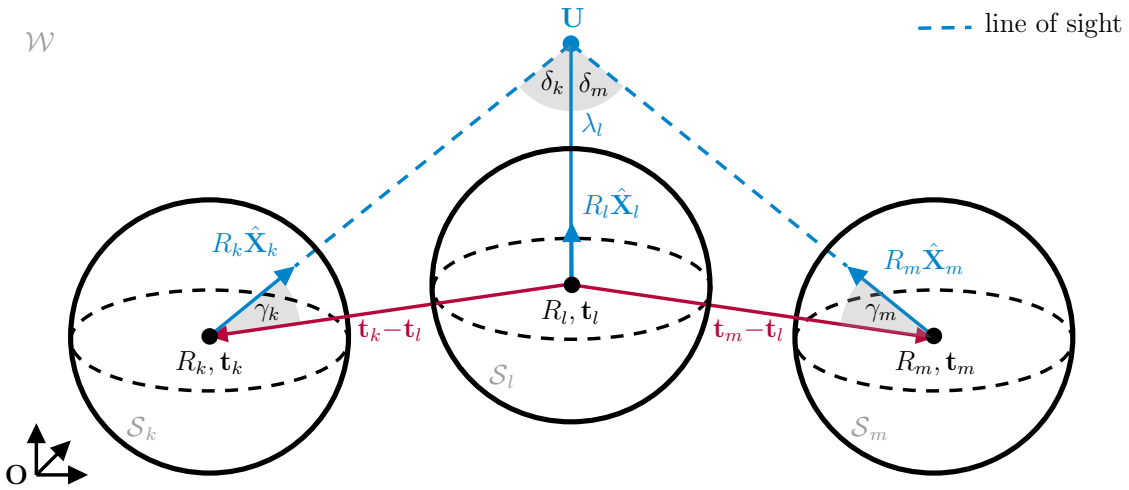

Fig. 7.1: Three-view geometry derivation

Indelman first mentioned the three-view geometry in [115] and used this constraint as a basic element in his structureless BA approach [116], which is referred as *incremental light bundle adjustment* [118, 117, 119]. The original three-view constraint derivation

is explained in [115]. This work explains a different derivation approach based on simple geometric relations as shown in the following. Furthermore, this new approach also explains the constraint's geometric meaning.

The three-view relation can be best described by two adjoining triangles defined by vertices $\triangle(\mathbf{t}_l, \mathbf{t}_k, \mathbf{U})$ and $\triangle(\mathbf{t}_m, \mathbf{t}_l, \mathbf{U})$ with a touching edge λ_l as shown in Fig. 7.1. This derivation is based on the law of sines, which reveals the relation for $\triangle(\mathbf{t}_l, \mathbf{t}_k, \mathbf{U})$

$$\frac{\lambda_l}{\sin\gamma_k} = \frac{\left\|\mathbf{t}_k - \mathbf{t}_l\right\|}{\sin\delta_k} \tag{7.1}$$

and for $\triangle(\mathbf{t}_m, \mathbf{t}_l, \mathbf{U})$

$$\frac{\lambda_l}{\sin\gamma_m} = \frac{\left\|\mathbf{t}_m - \mathbf{t}_l\right\|}{\sin\delta_m}\,. \tag{7.2}$$

The sine-terms are substituted with:

$$|\sin\delta_k| = \frac{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\|}{\left\|R_k\hat{\mathbf{X}}_k\right\|\left\|R_l\hat{\mathbf{X}}_l\right\|} = \left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\| \tag{7.3}$$

$$|\sin\delta_m| = \frac{\left\|R_l\hat{\mathbf{X}}_l \times R_m\hat{\mathbf{X}}_m\right\|}{\left\|R_l\hat{\mathbf{X}}_l\right\|\left\|R_m\hat{\mathbf{X}}_m\right\|} = \left\|R_l\hat{\mathbf{X}}_l \times R_m\hat{\mathbf{X}}_m\right\| \tag{7.4}$$

$$|\sin\gamma_k| = \frac{\left\|R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right\|}{\left\|R_k\hat{\mathbf{X}}_k\right\|\left\|\mathbf{t}_k - \mathbf{t}_l\right\|} = \frac{\left\|R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right\|}{\left\|\mathbf{t}_k - \mathbf{t}_l\right\|} \tag{7.5}$$

$$|\sin\gamma_m| = \frac{\left\|R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right\|}{\left\|R_m\hat{\mathbf{X}}_m\right\|\left\|\mathbf{t}_k - \mathbf{t}_l\right\|} = \frac{\left\|R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right\|}{\left\|\mathbf{t}_m - \mathbf{t}_l\right\|}\,. \tag{7.6}$$

Applying Eqs. (7.3) and (7.5) to Eq. (7.1) yields

$$\lambda_l = \frac{\left\|\mathbf{t}_k - \mathbf{t}_l\right\|}{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\|} \frac{\left\|R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right\|}{\left\|\mathbf{t}_k - \mathbf{t}_l\right\|} = \frac{\left\|R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right\|}{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\|} \tag{7.7}$$

and analogous applying Eqs. (7.4) and (7.6) to Eq. (7.2) leads to

$$\lambda_l = \frac{\left\|\mathbf{t}_m - \mathbf{t}_l\right\|}{\left\|R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l\right\|} \frac{\left\|R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right\|}{\left\|\mathbf{t}_m - \mathbf{t}_l\right\|} = \frac{\left\|R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right\|}{\left\|R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l\right\|}\,, \tag{7.8}$$

which are similiar to Eqs. (6.10) and (6.11), page 93. Substituting λ_l in Eq. (7.7) with Eq. (7.8) yields

$$\underbrace{\left\|R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l\right\|\left\|R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right\|}_{\|\mathbf{a}\|\,\|\mathbf{b}\| = \mathbf{a}\bullet\mathbf{b}\frac{1}{\cos\varphi}} = \underbrace{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\|\left\|R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right\|}_{\|\mathbf{c}\|\,\|\mathbf{d}\| = \mathbf{c}\bullet\mathbf{d}\frac{1}{\cos\varphi}}\,, \tag{7.9}$$

which can be further transformed such that

$$\begin{aligned}\left(R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l\right)\bullet\left(R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right)\frac{1}{\cos\varphi} = \\ \left(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right)\bullet\left(R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right)\frac{1}{\cos\varphi}\,.\end{aligned} \tag{7.10}$$

Both sides in Eq. (7.10) contain the angle φ. Due to the fact that $R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)$ and $R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l$ as well as $R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)$ and $R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l$ are parallel (as already discussed in Section 6.2.2, page 91, where collinearity was proved, the following vector combinations confine the angle φ:

$$\begin{aligned}
&I. && \varphi = \angle\left(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l \;, R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l\right)\\
&II. && \varphi = \angle\left(R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right), R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l\right)\\
&III. && \varphi = \angle\left(R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right), R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right)\\
&IV. && \varphi = \angle\left(R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right), R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right).
\end{aligned}$$

The relations $I. - IV.$ are also visualized in Fig. 7.2. Multiplying $\cos\varphi$ to both sides

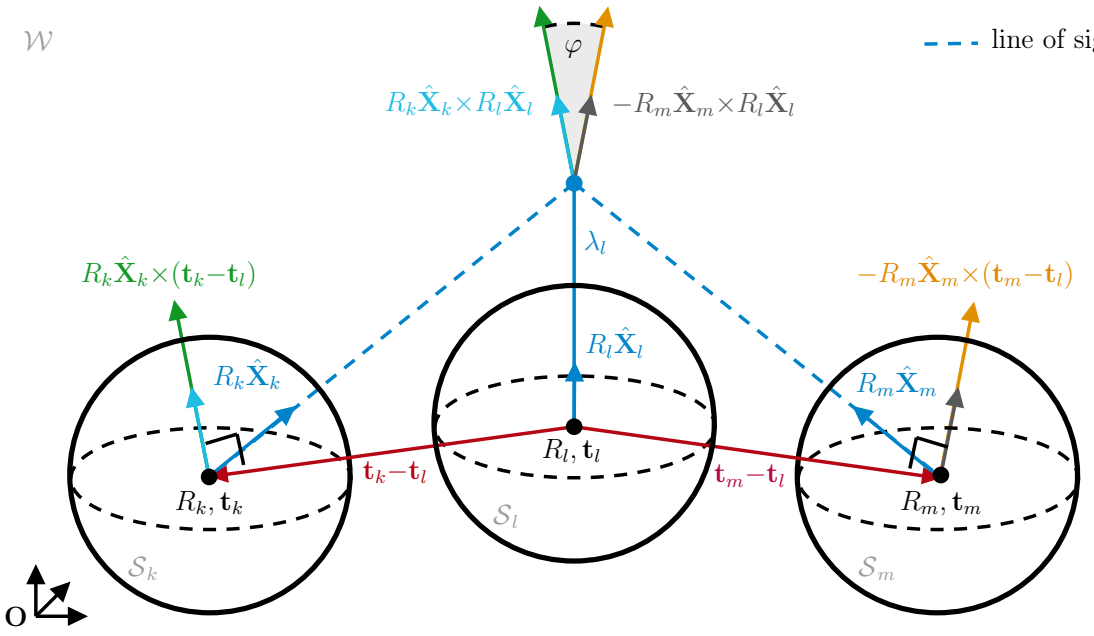

Fig. 7.2: Geometric relations of the three-view geometry

in Eq. (7.10) eliminates φ and subtracting the righthand site leads to the three-view relation

$$\boxed{0 = \left(R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right)\bullet\left(R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l\right) - \left(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right)\bullet\left(R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right)}\,. \tag{7.11}$$

As shown by the derived equation Eq. (7.11) the three-view geometry is able to scale two adjoining triangles $\triangle(\mathbf{t}_l, \mathbf{t}_k, \mathbf{U})$ and $\triangle(\mathbf{t}_m, \mathbf{t}_l, \mathbf{U})$, such that the touching edge be-

comes the same lenght in both triangles. Generally speaking, the three-view constraint connects two epipolar constraints by bringing the translations $(\mathbf{t}_k - \mathbf{t}_l) \leftrightarrow (\mathbf{t}_m - \mathbf{t}_l)$ into a correct scale as stated in [117], where however no explanation is given in a geometrical context.

In some older literature [256, 176, 177, 96] the term three-view is used to describe the trifocal geometry due to the fact, that Indelman's discovery wasn't published at that time. And even later publications may mix up these terms, since Indelman's three-view geometry is well-known in the field of robotics but not in computer vision.

Appendix A.12, page 184 describes two alternative derivation approaches to obtain the three-view constraint.

7.2 Crossing Epipolar Planes Geometry

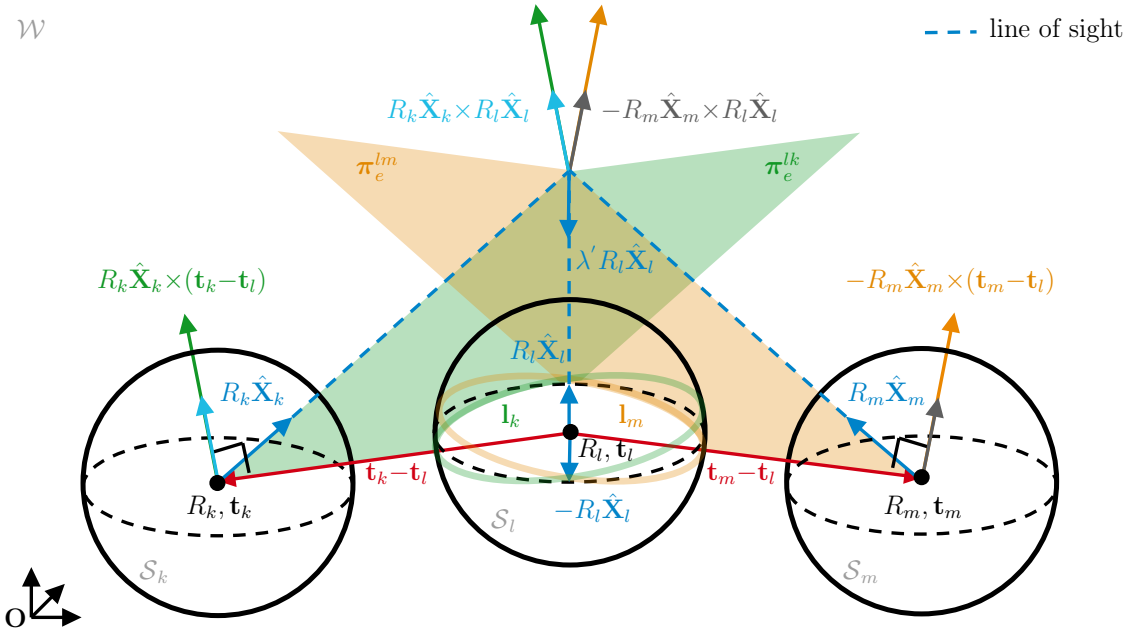

Fig. 7.3: Illustration of the *crossing epipolar planes* geometry

The idea of *crossing epipolar planes* is inspired by the *epipolar point transfer* function [96] for directional cameras, which transfers point correspondences between two known cameras into a third camera. This method represents an alternative to the naïve method consisting of triangulation and projection, where point pairs are triangulated first and then projected into the third camera. As a results, this method depends on triangulation uncertainties, which are projected into the third camera.

The following describes the derivation of the *epipolar point transfer* function for omnidirectional cameras using spherical coordinates in a first step. Based on these

findings the *crossing epipolar planes* constraint is developed in a second step.

Considering a given point correspondence $\hat{\mathbf{X}}_k \leftrightarrow \hat{\mathbf{X}}_m$ as well as the cameras' exterior orientations $[R_k, \mathbf{t}_k], [R_l, \mathbf{t}_l]$ and $[R_m, \mathbf{t}_m]$ in order to establish two epipolar geometries between cameras $k \leftrightarrow l$ and $l \leftrightarrow m$ as illustrated in Fig. 7.3. $\hat{\mathbf{X}}_l$ denotes the unknown corresponding point on camera sphere $\mathcal{S}_l$. ${\boldsymbol{\pi}_\mathrm{e}}^{lk}$ and ${\boldsymbol{\pi}_\mathrm{e}}^{lm}$ represent the epipolar planes with their corresponding normal vectors $R_k\hat{\mathbf{X}}_k \times (\mathbf{t}_k - \mathbf{t}_l)$ and $R_m\hat{\mathbf{X}}_m \times (\mathbf{t}_m - \mathbf{t}_l)$, respectively. The unkown LoS formed by $R_l\hat{\mathbf{X}}_l$ is the intersection line of both epipolar planes and the line's intersection point with the camera sphere $\mathcal{S}_l$ obtains the location of $\hat{\mathbf{X}}_l$. This calculation leads to two solutions, which are validated by cheirality using depth criterion.
This relationship can be also explained in a different way. The LoSs $R_k\hat{\mathbf{X}}_k$ and $R_m\hat{\mathbf{X}}_m$ have corresponding projections $\mathbf{l}_k$ and $\mathbf{l}_m$ on camera sphere $\mathcal{S}_l$, which are called great circles [257, 256] (similar to epipolar lines for directional cameras). Since the unknown point must lie on both great circles, the intersections between them are the solutions for the location of $\mathcal{S}_l$, comparable to the intersection of epipolar lines for directional cameras [229, 40, 96].

For the sake of completeness the derivation of the *epipolar point transfer* is described by means of the intersection of epipolar planes as this has so far not been explained in detail in the literature. Starting with the epipolar geometry from Eq. (6.12), page 95 for the relation between cameras $l \leftrightarrow k$ and $l \leftrightarrow m$:

$$0 = R_l\hat{\mathbf{X}}_l \bullet \left(R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right) \tag{7.12}$$
$$0 = R_l\hat{\mathbf{X}}_l \bullet \left(R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right). \tag{7.13}$$

Each epipolar equation yields a scalar value allowing to multiply $R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)$ to Eq. (7.12) and $R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)$ to Eq. (7.13) from the right:

$$\mathbf{0} = \left(R_l\hat{\mathbf{X}}_l \bullet \left(R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right)\right)\left(R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right) \tag{7.14}$$
$$\mathbf{0} = \left(R_l\hat{\mathbf{X}}_l \bullet \left(R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right)\right)\left(R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right). \tag{7.15}$$

Equalizing Eqs. (7.14) and (7.15) as well as bringing both terms to the right site yields

$$\begin{aligned}\mathbf{0} = &\left(\underbrace{R_l\hat{\mathbf{X}}_l}_{\mathbf{a}} \bullet \left(\underbrace{R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)}_{\mathbf{b}}\right)\right)\left(\underbrace{R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)}_{\mathbf{c}}\right) \\ &- \left(\underbrace{R_l\hat{\mathbf{X}}_l}_{\mathbf{a}} \bullet \left(\underbrace{R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)}_{\mathbf{c}}\right)\right)\left(\underbrace{R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)}_{\mathbf{b}}\right),\end{aligned} \tag{7.16}$$

which is simplified considering $(\mathbf{a} \bullet \mathbf{b})\mathbf{c} - (\mathbf{a} \bullet \mathbf{c})\mathbf{b} = \mathbf{a} \times (\mathbf{b} \times \mathbf{c})$ such that

$$\mathbf{0} = R_l\hat{\mathbf{X}}_l \times \left(\left(R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right) \times \left(R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right)\right). \tag{7.17}$$

Solving Eq. (7.17) for $\hat{\mathbf{X}}_l$ obtains the up-to-scale (λ') solution

$$\boxed{\lambda' R_l \hat{\mathbf{X}}_l = \left(R_k \hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right) \times \left(R_m \hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right)}, \tag{7.18}$$

as shown in Fig. 7.3. The solution of Eq. (7.18) equals the one for directional cameras presented in [96]. Normalizing $\lambda' R_l \hat{\mathbf{X}}_l$ projects the solution onto camera sphere $\mathcal{S}_l$ and multiplying R_l^T determines $\hat{\mathbf{X}}_l$. Since the sign of λ' is arbitrary, the sign of $\hat{\mathbf{X}}_l$ is also arbitrary, yielding two solutions $\hat{\mathbf{X}}_l$ and $-\hat{\mathbf{X}}_l$. Obtaining the sign of the depth value corresponding to $\hat{\mathbf{X}}_l$ via (Appendix A.11, page 184) resolves this ambiguity.
On the one hand the epipolar transfer equation obtains $\lambda' R_l \hat{\mathbf{X}}_l$ directly. On the other hand it is independent of $\|\mathbf{t}_k - \mathbf{t}_l\|$ and $\|\mathbf{t}_m - \mathbf{t}_l\|$, due to the circumstance that Eq. (7.17) finds a solution for an intersection line, which only depends on the directions $\mathbf{t}_k - \mathbf{t}_l$ and $\mathbf{t}_m - \mathbf{t}_l$. Thus the *epipolar point transfer* function is unable to determine the relative translation scale between two epipolar geometries.

Having a closer look at Fig. 7.3 reveals additonal solutions, which form a vector that is collinear with $R_l \hat{\mathbf{X}}_l$:

$$\begin{aligned}
&I. && \lambda' R_l \hat{\mathbf{X}}_l = \left(R_k \hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right) \times \left(R_m \hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right) \\
&II. && \lambda'' R_l \hat{\mathbf{X}}_l = \left(R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l\right) \times \left(R_m \hat{\mathbf{X}}_m \times R_l \hat{\mathbf{X}}_l\right) \\
&III. && \lambda''' R_l \hat{\mathbf{X}}_l = \left(R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l\right) \times \left(R_m \hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right) \\
&IV. && \lambda''' R_l \hat{\mathbf{X}}_l = \left(R_k \hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right) \times \left(R_m \hat{\mathbf{X}}_m \times R_l \hat{\mathbf{X}}_l\right).
\end{aligned}$$

As can be seen *III.* and *IV.* yield the same solution $\lambda''' R_l \hat{\mathbf{X}}_l$. Both relations are equalized and rearranged such that

$$\boxed{\mathbf{0} = \left(R_k \hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right) \times \left(R_m \hat{\mathbf{X}}_m \times R_l \hat{\mathbf{X}}_l\right) - \left(R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l\right) \times \left(R_m \hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right)}. \tag{7.19}$$

In contrast to the derived *epipolar point transfer* function this new constraint is able to determine the relative translation scale between two epipolar geometries. To the best of the author's knowledge, the determined constraint hasn't been published in literature before. In this work it is referred to as *crossing epipolar planes*.

Limitations

As shown in Fig. 7.3, Eq. (7.19) is only valid if $\left(R_k \hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right) \times \left(R_m \hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right) \neq \mathbf{0}$, (the normal vectors of the epipolar planes must not be parallel), such that both epipolar planes intersect in a line, which is given as long as $R_l \hat{\mathbf{X}}_l \bullet \left(R_k \hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right) \neq 0$ and consequently $R_k \hat{\mathbf{X}}_k \bullet \left(R_m \hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right)\right) \neq 0$. This means a point $\hat{\mathbf{X}}_k$ on camera sphere $\mathcal{S}_k$ must not be coplanar with the epipolar plane π_m formed by the point's correspondence on camera sphere $\mathcal{S}_l$ and $\mathcal{S}_m$ and vice versa. A further limitation is straight camera motion such that $\left(\mathbf{t}_k - \mathbf{t}_l\right) \times \left(\mathbf{t}_m - \mathbf{t}_l\right) = \mathbf{0}$.

7.3 Trifocal Geometry

The trifocal geometry describes the relation of corresponding line and point features between three camera views [205, 124]. It was originally introduced as trilinearity or trilinear equations (or functions, norms, relations) by [231, 229, 230] and [94, 99] showed that these constraints stem from a common tensor, the so called trifocal tensor. In [96] the authors give a detailed overview concerning the derivation of the trifocal geometry, its properties and application fields including camera calibration, point transfer and motion estimation for directional cameras. In [176, 177] the authors show another derivation approach by developing the trifocal constraints from a multi-view matrix.

Referring to [257, 256] for omnidirectional cameras, the trifocal constraint for point correspondences is given by

$$0_{3\times3} = [\mathbf{X}_k]_\times \Big(\sum_{i=1}^{3} \hat{\mathbf{X}}_{l(i)} \mathrm{T}_{(:,:,i)}\Big) [\mathbf{X}_m]_\times$$

where the trifocal tensor $\mathrm{T}_{(:,:,i)} = R_{ml(:,i)}\mathbf{t}_{kl}^T - \mathbf{t}_{ml}R_{kl(:,i)}^T$. $[R_{kl}, \mathbf{t}_{kl}]$ denotes the camera pose $[R_l, \mathbf{t}_l]$ transferred into camera frame $\mathcal{C}_k$ and similarly $[R_{ml}, \mathbf{t}_{ml}]$ denotes the camera pose $[R_l, \mathbf{t}_l]$ transferred into camera frame $\mathcal{C}_m$ such that:

$$R_{kl} = R_k^T R_l \qquad (7.20) \qquad \mathbf{t}_{kl} = R_k^T\big(\mathbf{t}_l - \mathbf{t}_k\big) \qquad (7.22)$$

$$R_{ml} = R_m^T R_l \qquad (7.21) \qquad \mathbf{t}_{ml} = R_m^T\big(\mathbf{t}_l - \mathbf{t}_m\big)\,. \qquad (7.23)$$

$\mathrm{T}_{(:,:,i)} \in \mathbb{R}^{3\times3\times3}$ indicates the i^{th} tensor slice. Referring to [176, 177] the trifcal tensor notation can be brought to matrix notation

$$0_{3\times3} = [\hat{\mathbf{X}}_k]_\times \big[\mathbf{t}_{kl}\hat{\mathbf{X}}_l^T R_{ml}^T - R_{kl}\hat{\mathbf{X}}_l\mathbf{t}_{ml}^T\big][\hat{\mathbf{X}}_m]_\times\,. \qquad (7.24)$$

Entering Eqs. (7.20) to (7.23) into Eq. (7.24) and rearranging yields:

$$\begin{aligned}
0_{3\times3} &= [\hat{\mathbf{X}}_k]_\times \big[R_k^T\big(\mathbf{t}_l - \mathbf{t}_k\big)\hat{\mathbf{X}}_l^T\big(R_m^T R_l\big)^T - R_k^T R_l\hat{\mathbf{X}}_l\big(R_m^T\big(\mathbf{t}_l - \mathbf{t}_m\big)\big)^T\big][\hat{\mathbf{X}}_m]_\times \\
0_{3\times3} &= [\hat{\mathbf{X}}_k]_\times \big[R_k^T\big(\mathbf{t}_l - \mathbf{t}_k\big)\hat{\mathbf{X}}_l^T R_l^T R_m - R_k^T R_l\hat{\mathbf{X}}_l\big(\mathbf{t}_l - \mathbf{t}_m\big)^T R_m\big][\hat{\mathbf{X}}_m]_\times \\
0_{3\times3} &= [\hat{\mathbf{X}}_k]_\times R_k^T\big[\big(\mathbf{t}_l - \mathbf{t}_k\big)\hat{\mathbf{X}}_l^T R_l^T - R_l\hat{\mathbf{X}}_l\big(\mathbf{t}_l - \mathbf{t}_m\big)^T\big]R_m[\hat{\mathbf{X}}_m]_\times \\
0_{3\times3} &= [R_k\hat{\mathbf{X}}_k]_\times \big[\big(\mathbf{t}_l - \mathbf{t}_k\big)\big(R_l\hat{\mathbf{X}}_l\big)^T - R_l\hat{\mathbf{X}}_l\big(\mathbf{t}_l - \mathbf{t}_m\big)^T\big][R_m\hat{\mathbf{X}}_m]_\times \\
0_{3\times3} &= \underbrace{[R_k\hat{\mathbf{X}}_k]_\times\big[\big(\mathbf{t}_l - \mathbf{t}_k\big)}_{[\mathbf{a}]_\times\mathbf{b}=\mathbf{a}\times\mathbf{b}} \underbrace{\big(R_l\hat{\mathbf{X}}_l\big)^T\big][R_m\hat{\mathbf{X}}_m]_\times}_{\mathbf{a}^T[\mathbf{b}]_\times=([\mathbf{b}]_\times^T\mathbf{a})^T=(\mathbf{a}\times\mathbf{b})^T} - \underbrace{[R_k\hat{\mathbf{X}}_k]_\times\big[R_l\hat{\mathbf{X}}_l}_{[\mathbf{a}]_\times\mathbf{b}=\mathbf{a}\times\mathbf{b}} \underbrace{\big(\mathbf{t}_l - \mathbf{t}_m\big)^T\big][R_m\hat{\mathbf{X}}_m]_\times}_{\mathbf{a}^T[\mathbf{b}]_\times=([\mathbf{b}]_\times^T\mathbf{a})^T=(\mathbf{a}\times\mathbf{b})^T} \\
0_{3\times3} &= \big(R_k\hat{\mathbf{X}}_k \times \big(\mathbf{t}_l - \mathbf{t}_k\big)\big)\big(R_l\hat{\mathbf{X}}_l \times R_m\hat{\mathbf{X}}_m\big)^T - \big(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\big)\big(\big(\mathbf{t}_l - \mathbf{t}_m\big) \times R_m\hat{\mathbf{X}}_m\big)^T \\
\boxed{0_{3\times3} &= \big(R_k\hat{\mathbf{X}}_k \times \big(\mathbf{t}_k - \mathbf{t}_l\big)\big)\big(R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l\big)^T - \big(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\big)\big(R_m\hat{\mathbf{X}}_m \times \big(\mathbf{t}_m - \mathbf{t}_l\big)\big)^T} \qquad (7.25)
\end{aligned}$$

Eq. (7.25) represents the trifocal constraint for omnidirectional cameras in matrix notation.

7.4 Relation between Trifocal, Three-View and Crossing Epipolar Planes

This section explains the relations between trifocal constraint, three-view constraint and the *crossing epipolar planes* constraint. For the sake of convenience the following terms are substituted:

$$\begin{aligned} \mathbf{d}_k &= R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l & \mathbf{g}_k &= R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right) \\ \mathbf{d}_m &= R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l & \mathbf{g}_m &= R_m\hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l\right) \end{aligned}$$

and applied to Eqs. (7.11), (7.19) and (7.25), pages 127, 130 and 131 yield the following relations:

$$\begin{aligned} \textbf{three-view:} \quad & 0 && = \mathbf{g}_k^T\mathbf{d}_m && -\mathbf{d}_k^T\mathbf{g}_m \\ \textbf{crossing epipolar planes:} \quad & \mathbf{0} && = [\mathbf{g}_k]_\times\mathbf{d}_m && -[\mathbf{d}_k]_\times\mathbf{g}_m \\ \textbf{trifocal:} \quad & 0_{3\times3} && = \mathbf{g}_k\mathbf{d}_m^T && -\mathbf{d}_k\mathbf{g}_m^T\,. \end{aligned}$$

Defining the trifocal constraint $\mathbf{g}_k\mathbf{d}_m^T - \mathbf{d}_k\mathbf{g}_m^T = A$ and solving it leads to:

$$\begin{aligned} \mathbf{g}_k\mathbf{d}_m^T - \mathbf{d}_k\mathbf{g}_m^T &= \begin{bmatrix} \mathbf{d}_{m(1)}\mathbf{g}_{k(1)} - \mathbf{d}_{k(1)}\mathbf{g}_{m(1)} & \mathbf{d}_{m(2)}\mathbf{g}_{k(1)} - \mathbf{d}_{k(1)}\mathbf{g}_{m(2)} & \mathbf{d}_{m(3)}\mathbf{g}_{k(1)} - \mathbf{d}_{k(1)}\mathbf{g}_{m(3)} \\ \mathbf{d}_{m(1)}\mathbf{g}_{k(2)} - \mathbf{d}_{k(2)}\mathbf{g}_{m(1)} & \mathbf{d}_{m(2)}\mathbf{g}_{k(2)} - \mathbf{d}_{k(2)}\mathbf{g}_{m(2)} & \mathbf{d}_{m(3)}\mathbf{g}_{k(2)} - \mathbf{d}_{k(2)}\mathbf{g}_{m(3)} \\ \mathbf{d}_{m(1)}\mathbf{g}_{k(3)} - \mathbf{d}_{k(3)}\mathbf{g}_{m(1)} & \mathbf{d}_{m(2)}\mathbf{g}_{k(3)} - \mathbf{d}_{k(3)}\mathbf{g}_{m(2)} & \mathbf{d}_{m(3)}\mathbf{g}_{k(3)} - \mathbf{d}_{k(3)}\mathbf{g}_{m(3)} \end{bmatrix} \\ &\triangleq \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \in \mathbb{R}^{3\times3}\,. \end{aligned} \tag{7.26}$$

Solving the three-view constraint $\mathbf{g}_k^T\mathbf{d}_m - \mathbf{d}_k^T\mathbf{g}_m$ and substituting the result with elements from A yields:

$$\begin{aligned} \mathbf{g}_k^T\mathbf{d}_m - \mathbf{d}_k^T\mathbf{g}_m =& \mathbf{d}_{m(1)}\mathbf{g}_{k(1)} - \mathbf{d}_{k(1)}\mathbf{g}_{m(1)} + \\ & \mathbf{d}_{m(2)}\mathbf{g}_{k(2)} - \mathbf{d}_{k(2)}\mathbf{g}_{m(2)} + \\ & \mathbf{d}_{m(3)}\mathbf{g}_{k(3)} - \mathbf{d}_{k(3)}\mathbf{g}_{m(3)} \\ \triangleq& a_{11} + a_{22} + a_{33}\,. \end{aligned} \tag{7.27}$$

Doing the same with the *crossing epipolar planes* leads to:

$$[\mathbf{g}_k]_\times \mathbf{d}_m - [\mathbf{d}_k]_\times \mathbf{g}_m = \begin{pmatrix} \mathbf{d}_{m(3)}\mathbf{g}_{k(2)} - \mathbf{d}_{k(2)}\mathbf{g}_{m(3)} - (\mathbf{d}_{m(2)}\mathbf{g}_{k(3)} - \mathbf{d}_{k(3)}\mathbf{g}_{m(2)}) \\ \mathbf{d}_{m(1)}\mathbf{g}_{k(3)} - \mathbf{d}_{k(3)}\mathbf{g}_{m(1)} - (\mathbf{d}_{m(3)}\mathbf{g}_{k(1)} - \mathbf{d}_{k(1)}\mathbf{g}_{m(3)}) \\ \mathbf{d}_{m(2)}\mathbf{g}_{k(1)} - \mathbf{d}_{k(1)}\mathbf{g}_{m(2)} - (\mathbf{d}_{m(1)}\mathbf{g}_{k(2)} - \mathbf{d}_{k(2)}\mathbf{g}_{m(1)}) \end{pmatrix}$$
$$\hat{=} \begin{pmatrix} a_{23} - a_{32} \\ a_{31} - a_{13} \\ a_{12} - a_{21} \end{pmatrix} . \tag{7.28}$$

It is shown that the solution elements from three-view and *crossing epipolar planes* are combinations of the trifocal solution elements. Since $A \hat{=} 0_{3\times3}$, the trifocal constraint from Eq. (7.26) also satisfies both, the three-view geometry from Eq. (7.27) and the *crossing epipolar planes* from Eq. (7.28). If the trifocal constraint is used as a cost function for optimization purposes, it also satisfies the forementioned constraints. The relation between three-view and trifocal constraint was published in [56], however the remaining elements in A aren't analyzed and [118, 119] showed the relation between both constraints from the perspective of a common derivation approach. Appendix A.13, page 185 shows the relation between trifocal constraint and *alternative* midpoint method, which however is not discussed at this point.

7.5 Translation Ratio between Up-To-Scale Two-View Transformations

The aforementioned constraints from Sections 7.1 to 7.3 describe the relation of point correspondences between three camera spheres k, l, m and maintain the relation between both translations $\mathbf{t}_k - \mathbf{t}_l$ and $\mathbf{t}_m - \mathbf{t}_l$. Hence they can be used to determine the translation ratio between a pair of up-to-scale two-view transformations as shown in the following.

7.5.1 Structureless Determination Approaches

Crossing epipolar planes, three-view and trifocal constraints describe a direct relation between the translation terms and do not require any 3d structure for scaling purposes. Supposing a point correspondence $\hat{\mathbf{X}}_k \leftrightarrow \hat{\mathbf{X}}_l \leftrightarrow \hat{\mathbf{X}}_m$ between three cameras with given exterior orientations $[R_k, \mathbf{t}_k], [R_l, \mathbf{t}_l], [R_m, \mathbf{t}_m]$, which are transferred into camera frame $\mathcal{C}_l$ such that camera l becomes the new center of origin $[I, \mathbf{0}]$. Camera k and camera m are expressed as relative transformations:

$$R_{lk} = R_l^T R_k \qquad \mathbf{t}_{lk} = R_l^T(\mathbf{t}_k - \mathbf{t}_l)$$
$$R_{lm} = R_l^T R_m \qquad \mathbf{t}_{lm} = R_l^T(\mathbf{t}_m - \mathbf{t}_l)\,.$$

Assuming $\mathbf{t}_{lk} = s_{lk}\hat{\mathbf{t}}_{lk}$ and $\mathbf{t}_{lm} = s_{lm}\hat{\mathbf{t}}_{lm}$ with s_{lk}, s_{lm} being the corresponding translation scale factors as introduced in Section 6.7, page 113. $[R_{lk}, \hat{\mathbf{t}}_{lk}]$ represents the relative up-to-scale transformation between cameras $l \leftrightarrow k$ and $[R_{lm}, \hat{\mathbf{t}}_{lm}]$ represents the relative up-to-scale transformation between cameras $l \leftrightarrow m$, both are obtained via two-view estimation/optimization. Similiar to the previous section the following terms are substituted:

$$\mathbf{d}_k = R_{lk}\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l \qquad \tilde{\mathbf{g}}_k = R_{lk}\hat{\mathbf{X}}_k \times \hat{\mathbf{t}}_{lk}$$
$$\mathbf{d}_m = R_{lm}\hat{\mathbf{X}}_m \times \hat{\mathbf{X}}_l \qquad \tilde{\mathbf{g}}_m = R_{lm}\hat{\mathbf{X}}_m \times \hat{\mathbf{t}}_{lm}$$

in order to derive a simplified expression for the proposed constraints:

$$\textbf{three-view:} \quad 0 = s_{lk}\tilde{\mathbf{g}}_k^T\mathbf{d}_m - s_{lm}\mathbf{d}_k^T\tilde{\mathbf{g}}_m \tag{7.29}$$
$$\textbf{crossing epipolar planes:} \quad \mathbf{0} = s_{lk}[\tilde{\mathbf{g}}_k]_\times\mathbf{d}_m - s_{lm}[\mathbf{d}_k]_\times\tilde{\mathbf{g}}_m \tag{7.30}$$
$$\textbf{trifocal:} \quad 0_{3\times 3} = s_{lk}\tilde{\mathbf{g}}_k\mathbf{d}_m^T - s_{lm}\mathbf{d}_k\tilde{\mathbf{g}}_m^T\,. \tag{7.31}$$

The translation ratio between a pair of two-view transformations is defined by $r_{klm} = s_{lk}/s_{lm}$ yielding:

$$\textbf{three-view:} \quad \underbrace{\mathbf{d}_k^T \tilde{\mathbf{g}}_m}_{\mathbf{b}} = r_{klm} \underbrace{\tilde{\mathbf{g}}_k^T \mathbf{d}_m}_{\mathbf{a}} \tag{7.32}$$

$$\textbf{crossing epipolar planes:} \quad \underbrace{[\mathbf{d}_k]_\times \tilde{\mathbf{g}}_m}_{\mathbf{b}} = r_{klm} \underbrace{[\tilde{\mathbf{g}}_k]_\times \mathbf{d}_m}_{\mathbf{a}} \tag{7.33}$$

$$\textbf{trifocal:} \quad \underbrace{\mathbf{d}_k \tilde{\mathbf{g}}_m^T}_{\mathbf{b}} = r_{klm} \underbrace{\tilde{\mathbf{g}}_k \mathbf{d}_m^T}_{\mathbf{a}} . \tag{7.34}$$

The column vectors $\mathbf{a}$ and $\mathbf{b}$ collect the elements of the two terms formed by $\tilde{\mathbf{g}}_k$, $\mathbf{d}_m$ and $\mathbf{d}_k$, $\tilde{\mathbf{g}}_m$. Following Appendix A.2, page 174 the translation ratio is optained via

$$\boxed{r_{klm} = (\mathbf{a}^T \mathbf{b})/\|\mathbf{a}\|^2} \tag{7.35}$$

in least squares sense. Supposing a set of given point correspondences $\hat{\mathbf{X}}_k^i \leftrightarrow \hat{\mathbf{X}}_l^i \leftrightarrow \hat{\mathbf{X}}_m^i$, $i = 1, \ldots, p$, the resulting $\mathbf{a}_i$, $\mathbf{b}_i$ can be simply stacked together:

$$\mathbf{a} = \left(\mathbf{a}_1^T, \ldots, \mathbf{a}_p^T\right)^T \tag{7.36}$$

$$\mathbf{b} = \left(\mathbf{b}_1^T, \ldots, \mathbf{b}_p^T\right)^T . \tag{7.37}$$

to obtain an optimized solution for r_{klm} over all point correspondences.

Similiar to Appendix A.9, page 182 the derived ratio factor's uncertainty as standard deviation is approximated such that

$$\boxed{\sigma_{r_{klm}} = \frac{\|\mathbf{b} - r_{klm}\mathbf{a}\|}{\sqrt{p-1}\|\mathbf{a}\|}} . \tag{7.38}$$

7.5.2 Structure-Based Determination Approaches

The proposed naïve approach is inspired by [218] and obtains the translation ratio factor between triangulated points from two different camera pairs as shown in Fig. 7.4. Thus this approach generates 3d structures for scaling purpose. $\mathbf{X}_{lk}$ denotes the triangulated point from $\hat{\mathbf{X}}_k \leftrightarrow \hat{\mathbf{X}}_l$ and $\mathbf{X}_{lm}$ denotes the triangulated point from $\hat{\mathbf{X}}_l \leftrightarrow \hat{\mathbf{X}}_m$. Both points represent the same observation, however they do not coincide since they stem from up-to-scale translations and need to be brought into correct relation such that

$$r_{klm} = \frac{s_{lk}}{s_{lm}} = \frac{\|\mathbf{X}_{lk}\|}{\|\mathbf{X}_{lm}\|} . \tag{7.39}$$

As can be seen the relation between the triangulated points $\|\mathbf{X}_{lk}\|/\|\mathbf{X}_{lm}\|$ equals the one between the translation scales s_{lk}/s_{lm}. Since $\mathbf{X}_{lk}$ and $\mathbf{X}_{lm}$ are in the same camera

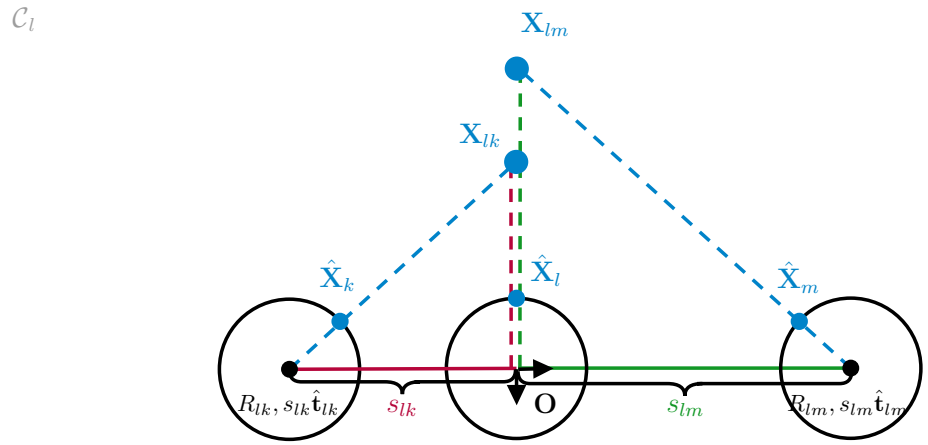

Fig. 7.4: Naïve triangulation approach to determine the translation ratio. The triangulated points $\mathbf{X}_{lk}$ and $\mathbf{X}_{lm}$ represent the same observed 3d point. Since both triangulation are based on up-to-scale transformations, $\mathbf{X}_{lk}$ and $\mathbf{X}_{lm}$ do not coincide. As can be seen, the triangulated points are in the same relation as the baselines scales are $\|\mathbf{X}_{lk}\|/\|\mathbf{X}_{lm}\| = s_{lk}/s_{lm}$. As a consequence, scaling the triangulated points such that they coincide also scales the baselines to a correct relation.

frame $\mathcal{C}_l$, Eq. (7.35) can be used to obtain the least squares solution for r_{klm} with:

$$\mathbf{a} = \left((\mathbf{X}_{lk}^1)^T, \ldots, (\mathbf{X}_{lk}^p)^T\right)^T \tag{7.40}$$

$$\mathbf{b} = \left((\mathbf{X}_{lm}^1)^T, \ldots, (\mathbf{X}_{lm}^p)^T\right)^T, \tag{7.41}$$

in case of multiple triangulated point correspondences $\mathbf{X}_{lk}^i \leftrightarrow \mathbf{X}_{lm}^i$, $i = 1, \ldots, p$. The approach depends on the accuray of the underlying triangulation method. In this work it incorporates the linear least squares method (Section 6.2.1, page 89) and the *alternative* midpoint method (Section 6.2.2, page 91).

7.5.3 Comparison between Proposed Approaches

The approaches from Sections 7.5.1 and 7.5.2 are used to obtain an estimated translation ratio r_{est} from a pair of up-to-scale two-view transformations $[R_{lk}, \hat{\mathbf{t}}_{lk}]$, $[R_{lk}, \hat{\mathbf{t}}_{lm}]$ and a set of given point correspondences $\hat{\mathbf{X}}_k^i \leftrightarrow \hat{\mathbf{X}}_l^i \leftrightarrow \hat{\mathbf{X}}_m^i$, $i = 1, \ldots, p$ under varying perturbations for the omnidirectional view **case a** as well as for the directional view **case b**.

Similiar to Section 6.7.3, page 117 this section analyzes the influence of different noise sources on translation ratio determination such as image noise $\mathcal{N}(0, \sigma_{\theta,\phi}^2)$ with $\sigma_{\theta,\phi} = 0°, \ldots, 1°$, rotation noise $\mathcal{N}(0, \sigma_{\gamma,\beta,\alpha}^2)$ with $\sigma_{\gamma,\beta,\alpha} = 0°, \ldots, 1°$ and relative translation noise $\mathcal{N}(0, \sigma_{X,Y,Z}^2)$ with $\sigma_{X,Y,Z} = 0\%, \ldots, 10\%$.

The relative ratio error Δ_r is obtained via

$$\Delta_r = \left| \frac{r_{\text{est}} - r_{\text{true}}}{r_{\text{true}}} \right|, \tag{7.42}$$

where $r_{\text{true}} = \|\mathbf{t}_{lk}\|/\|\mathbf{t}_{lm}\|$ denotes the ground truth.

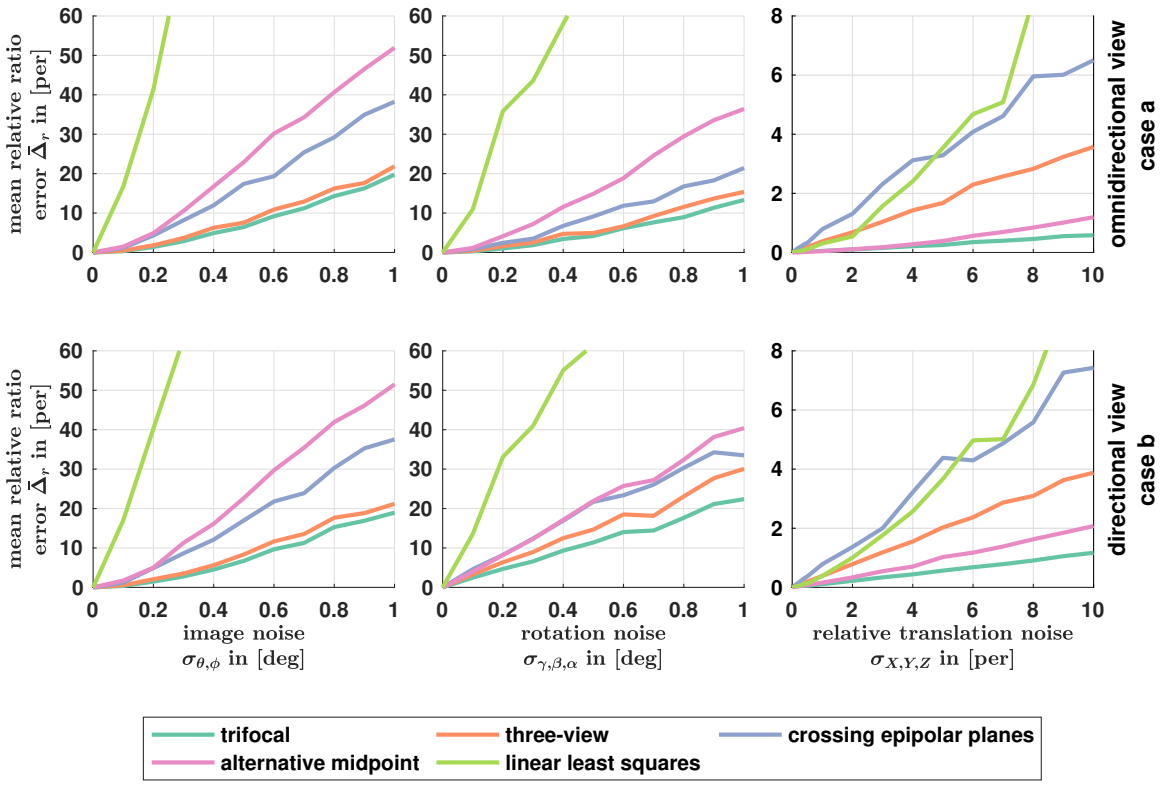

Fig. 7.5: Overview of the mean relative translation scaling error $\bar{\Delta}_r$ caused by image noise $\sigma_{\theta,\phi}$, rotation noise $\sigma_{\gamma,\beta,\alpha}$ and translation noise $\sigma_{X,Y,Z}$ for omnidirectional view (1st row, **case a**) and directional view (2nd row, **case b**).

As Fig. 7.5 shows, structured-based approaches (linear least squares methods, *alternative* midpoint method) are more prone to noise. Especially the linear least squares triangulation method yields large estimation errors even in case of low image and rotation noise. Suprisingly the *alternative* midpoint method is less influenced by relative translation noise and achieves satisfying results. The trifocal constraint obtains the best results over all noise sources and view cases. The three-view constraint yields similiar results for image and rotation noise, but is less accurate under translation noise. *Crossing epipolar planes* has the worst performance of all structureless approaches. Image and rotation noise stronger influence the ratio factor calculation - up to a magnitude larger errors - compared to the relative translation noise.

Considering all mentioned aspects the trifocal constraint is preferred to obtain the translation ratio between up-to-scale two-view transformations.

Brief Chapter Summary

This chapter explained three constraints describing the relation of point correpsondences between three camera spheres. The three-view geometry was derived for spherical camera and a novel *crossing epipolar planes* geometry was developed. Additionally the trifocal geometry was reformulated Furthermore, the relation between them was explained. A method was presented to recover the translation ratio between up-to-scale two-view transformations based on mentioned geometry constraints and compared with a triangulaiton approach. The different geometrical constraints were evaluated, where the trifocal constraint obtained best results. Based on the findings of this chapter the translation ratio between up-to-scale two-view transformations can be recovered to be used for global scaling.

8 Pose Graphs

Brief Chapter Overview

This chapters gives a general explanation to PGO in Section 8.1 representing the back-end of the proposed SCME pipeline. Section 8.2 introduces state-of-the-art standalone solvers for PGO, which are used in this work. An overview of additional solvers is given in Section 8.2.1 and Section 8.2.2 addresses the topic of false loop closure detection. Section 8.3 explains a workflow to extract a trajectory from an unordered set of relative transformations e.g. from scaled two-view transformations. Section 8.3.1 applies the proposed extraction workflow to polygon models in order to test the pipeline intergration and to generate synthetic pose graph data. Finally, in Section 8.3.2 standalone solvers are applied to synthetic pose graph data for evaluation purpose.

8.1 Optimization Principle

This section describes the basic optimization principle of PGO and follows the description from [28, 25, 33, 32]. Pose graphs represent a special form of a factor graph (Section 2.1, page 14) dedicated to pose estimation, such that features are not explicitly modelled and thus are not part of the optimization problem. PGO estimates a set of n poses $\{[R_1, \mathbf{t}_1], \ldots, [R_n, \mathbf{t}_n]\}$ from pairwise relative transformation measurements. This problem can be visualized as a directed graph, in which nodes correspond to camera poses $[R_k, \mathbf{t}_k]$ (and $[R_l, \mathbf{t}_l]$, respectively), while edges correspond to relative transformation measurements $[R_{k,l}, \mathbf{t}_{k,l}]$ between the k^{th} and l^{th} node, and belong to a set of available node pairs $(k, l) \in \mathcal{E}$.
The pose graph solver estimates $[R_k, \mathbf{t}_k]$ and $[R_l, \mathbf{t}_l]$ in order to minimize the cost functions $\mathbf{d}_{\mathrm{SO}(3)}(\cdot,\cdot)$ and $\mathbf{d}_{\mathbb{R}^3}(\cdot,\cdot)$ with respect to the available measurements $[R_{kl}, \mathbf{t}_{kl}]$ by solving the least squares optimization problem

$$\underset{R_{k,l},\mathbf{t}_{k,l}}{\operatorname{argmin}} \sum_{(k,l)\in\mathcal{E}} \mathbf{d}_{\mathrm{SO}(3)} \left(R_{kl}, R_k^T R_l\right)^2 + \mathbf{d}_{\mathbb{R}^3} \left(\mathbf{t}_{kl}, R_k^T \left(\mathbf{t}_l - \mathbf{t}_k\right)\right)^2 . \tag{8.1}$$

Here, $\mathbf{d}_{\mathrm{SO}(3)}(\cdot,\cdot)$ denotes a distance metric between two rotations. This example uses the angular distance yielding

$$\mathbf{d}_{\mathrm{SO}(3)} \left(R_{kl}, R_k^T R_l\right) \quad = \left\| \mathrm{Log} \left(R_{kl}^T R_k^T R_l\right) \right\|_{\Omega_{kl}^{\mathrm{rot}}} , \tag{8.2}$$

where $\mathrm{Log}(\cdot)$ stands for the logarithm map, which - generally speaking - converts the rotation matrix to a vector (e.g. Euler angles, axis angles, Rodrigues vector or quaternion). Furthermore, $\mathbf{d}_{\mathbb{R}^3}(\cdot,\cdot)$ denotes the simple Euclidean distance between two vectors:

$$\begin{aligned}\mathbf{d}_{\mathbb{R}^3}\left(\mathbf{t}_{kl}, R_k^T\left(\mathbf{t}_l - \mathbf{t}_k\right)\right) &= \left\|\mathbf{t}_{kl} - R_k^T\left(\mathbf{t}_l - \mathbf{t}_k\right)\right\|_{\Omega_{kl}^{\text{trans}}} \\ &= \left\|\mathbf{t}_l - \mathbf{t}_k - R_k\mathbf{t}_{kl}\right\|_{\Omega_{kl}^{\text{trans}}} . \end{aligned} \tag{8.3}$$

Each relative transformation is affected by noise causing measuring uncertainties, which are described by the rotation information matrix Ω_{kl}^{rot} and by the translation information matrix $\Omega_{kl}^{\text{trans}}$. Applying Eqs. (8.2) and (8.3) to Eq. (8.1) yields

$$\underset{R_{k,l},\mathbf{t}_{k,l}}{\operatorname{argmin}} \sum_{(k,l)\in\mathcal{E}} \left\|\mathrm{Log}\left(R_{kl}^T R_k^T R_l\right)\right\|_{\Omega_{kl}^{\text{rot}}}^2 + \left\|\mathbf{t}_l - \mathbf{t}_k - R_k\mathbf{t}_{kl}\right\|_{\Omega_{kl}^{\text{trans}}}^2 \tag{8.4}$$

and developing the Mahalanobis norms leads to

$$\begin{aligned}\underset{R_{k,l},\mathbf{t}_{k,l}}{\operatorname{argmin}} \sum_{(k,l)\in\mathcal{E}} &\mathrm{Log}\left(R_{kl}^T R_k^T R_l\right)\Omega_{kl}^{\text{rot}}\mathrm{Log}\left(R_{kl}^T R_k^T R_l\right) \\ &+ \left(\mathbf{t}_l - \mathbf{t}_k - R_k\mathbf{t}_{kl}\right)\Omega_{kl}^{\text{trans}}\left(\mathbf{t}_l - \mathbf{t}_k - R_k\mathbf{t}_{kl}\right) . \end{aligned} \tag{8.5}$$

Depending on the chosen solver implementation, the cost functions may vary. In [31] the authors employ the *chordal distance* [95] $\mathbf{d}_{\mathrm{SO}(3)} = \left\|R_l - R_k R_{k,l}\right\|_{\mathrm{F}}$ as rotation cost function. A very good overview of existing cost function strategies gives [7].

8.2 Solvers

The following described solvers are used for PGO in this work. The *g2o* file format acts as interface between proposed front-end and standalone graph solvers acting as back-end. These solvers concentrate on computation time reduction, memory consumption [242] and improved robustness to solve large scale problems efficiently. In [86] the authors give an additional overview of current solvers and corresponding cost function implementations.

GTSAM

GTSAM[75] (Georgia Tech Smoothing and Mapping) is an open-source *C++* library [47] and offers SLAM, VO, SfM and BA functionality [121]. The library uses factor graphs and Bayesian Belief Networks to model non-linear least square problems, rather than using sparse linear algebra. It provides LM, GN, Dl and iSAM/iSAM2 optimizers [125]. GTSAM's fundamental principles are best described in [49].

[75] https://gtsam.org

iSAM2

iSAM (incremental Scanning and Mapping) is an optimizer for GTSAM introduced in [127]. It is based on sparse matrix factorization [48, 132] and in contrast to batch processing iSAM performs fast incremental updates each time when new measurements become available. iSAM only executes calculations for entries that are actually affected by the new measurements. The successor iSAM2 [130, 131] operates incrementally as well but uses an efficient Bayes tree structure [129], which leads to improved efficiency and accuracy. This algorithm is part of the GTSAM library.

g2o

g2o[76] is an open-source *C++* framework and is introduced in [151]. It uses a graph structure to optimize non-linear graph-based error functions in a batch process and is able to provide solutions to serveral SLAM and BA problems. The framework uses popular solvers like LM, GN or Dl to solve PGO [125]. The framework can be easily extended by embedding new error functions and solvers.

Ceres

Ceres[77] [3] is an open-source *C++* library provided by *Google*. It is used to solve large non-linear problems in least-squares sense using e.g. LM or Dl implementation [125]. It is popular for solving large-squale BA problems (e.g. it is implemented in *Colmap*, *Theia* SfM and *Open MVG*) but also performs PGO. Since *Ceres* doesn't provide pose graph initialization capability, it is used in conjunction with MASAT.

SLAM++

SLAM++[78][79] is a *C++* package providing an efficient implementation of incremental non-linear least squares solvers [114]. The algorithm maintains a sparse and scalable state representation for large scale problems and performs matrix block operations, which lead to fast matrix manipulation and arithmetic operations. Combined with the use of parallel computing units, SLAM++ achieves higher speed performances compared to other SLAM implementations. The algorithm provides fast mean and covariance updates for the estimates. Instead of re-linearizing the entire graph at each optimization step, the algorithm selectively updates the factor contributions and recomputes the factorization only for some changed regions. SLAM++ also performs BA by using an incremental Dl solver instead of LM [113].

Note: In order to prevent confusion, there is an object-based SLAM algorithm, which

[76] https://github.com/RainerKuemmerle/g2o
[77] http://ceres-solver.org
[78] https://sourceforge.net/p/slam-plus-plus/wiki/Home/
[79] http://lukas-polok.cz/proj_slam++.htm

is also called SLAM++ [214]. Both algorithms were developed at the same time, named independently and accidently presented at the same conference (ICRA2013).

MatLab

A first pose graph solver for *MatLab* is presented in [265]. Since version R2019b, *MatLab* provides an own 3d pose graph solver, which is used for the extrinsic calibration pipeline in this work (see Section 4.4.5, page 67). However, the here presented PGO problems are magnitudes larger (in both sizes, number of nodes and edges). The *MatLab* implementation is less effective in terms of computational time compared to standalone solvers.

Masat

MASAT[80] (Multi-Ancestor Spatial Approximation Tree) is a pose graph initialization algorithm written in *C++* [92]. It deals as a pre-processing step before running the actual pose graph solver, since non-linear optimization may diverge or converge into a local minimum if the initial estimates are too far from the ground truth [32]. A more detailed insight into pose graph initialization is given in [31, 33]. In contrast to complex initialization techniques (also known as bootstrappers) such as *LAGO*[81] (Linear Approximation for Graph Optimization, restricted to 2d scenarios only) [29, 30], *TORO*[82] (Tree-based netwORk Optimizer) [87, 85, 84] or *Cauchy M-estimator* [109], the algorithm is built up on low complexity to be computationally efficient. MASAT traverses the pose graph and approximates each node's location based on its already positioned neighbors. This simply approach obtaines better initial estimates compared to a standard pose graph solver as Fig. 8.1 illustrates at hand of synthetic data from Section 8.3.1, page 147. In this work MASAT is used as pre-processor for PGO since initial estimates are much closer to the ground truth and hence reduce the number of optimization steps in the pose graph solver.

8.2.1 Additional Graph Solvers

This section briefly lists additional software providing graph solvers, which were discovered during research. They are not used as backend in this work, but for the sake of a more complete overview they are mentioned here.

- **AprilSAM**[83] decides between incremental and batch updates and uses a new dynamic variable reordering algorithm [267] to achieve improved performance.
- **Kimera-RPGO**[84] (Robust Pose Graph Optimization) is part of the open-source *C++* library **Kimera** (Open-Source Library for Real-Time Metric-Semantic Lo-

[80] https://github.com/karoly-hars/MASAT_IG_for_SLAM
[81] https://github.com/rrg-polito/lago
[82] https://github.com/OpenSLAM-org/openslam_toro
[83] https://github.com/xipengwang/AprilSAM
[84] https://github.com/MIT-SPARK/Kimera-RPGO

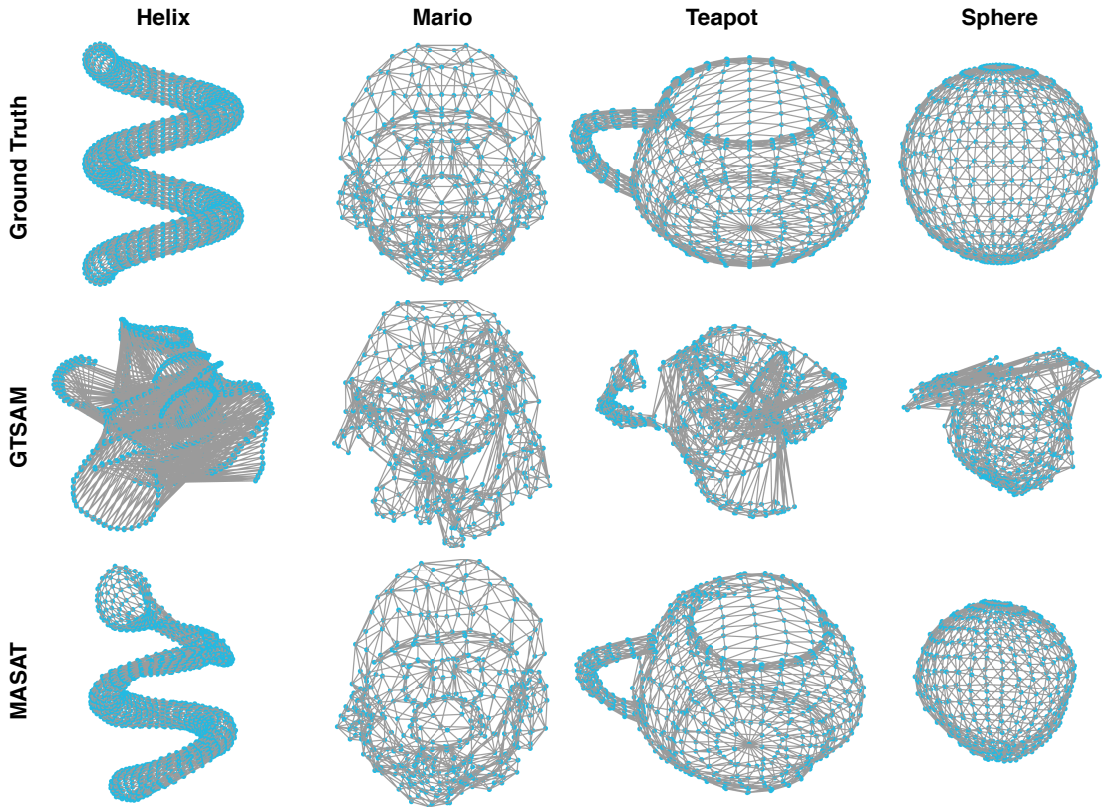

Fig. 8.1: Comparison of pose graph initialization between GTSAM, MASAT and ground truth using different synthetic data sets (*Helix*, *Mario*, *Teapot*, *Sphere*). Initialization results from iSAM2 and SLAM++ look similiar to GTSAM and hence are not illustrated here.

calization and Mapping). It uses GTSAM as back-end [211] and modern techniques for outlier rejection.

- **COP-SLAM**[85] (Closed-form Online Pose-chain) is a highly efficient closed-form solver approach, which optimizes pose-chains (specific type of extremely sparse pose-graphs) in real time [52].

- **miniSAM** is a flexible, general and lightweight factor graph optimization framework, that aims at providing a full *Python* API (Application Programming Interface) and *CUDA* (Compute Unified Device Architecture) supported sparse linear solvers [51].

- **MRPT**[86] (Mobile Robot Programming Toolkit) is a collection of *C++* libraries providing a wide range of functionality to the computer vision and robotics community as it also provides PGO.

- **SE-Sync**[87] is a certifiably correct algorithm obtaining unknown poses given noisy measurements of relative transformations [209, 210]. It uses a combination of low-rank factorization and fast local search method (referred as *Riemannian*

[85] https://github.com/OpenSLAM-org/openslam_copslam
[86] https://www.mrpt.org
[87] https://github.com/david-m-rosen/SE-Sync

truncated-Newton trust-region method [125, 254]) to solve for poses efficiently and is capable to certify the correctness of the solutions that it recovers. *SE-Sync* belongs to the class of algorithms that give up the ability to solve every instance of a problem, meaning that it is capable of efficiently solve a generally intractable problem within a restricted operational regime. The algorithm reduces the pose estimation and optimization problem to a rotational one by analytically eliminating the translational variables [21, 254].

- **Cartan-Sync**[88][89] is a pose optimization framework [21] similar to *SE-Sync*. It uses an SDP (Semidefinite Program) relaxation, which jointly obtains rotations and translations.

8.2.2 False Loop Closure Detection

PGO is based on least-squares optimization (see Section 8.1, page 139), which means the solver keeps the topology of the graph fixed. Hence optimization is not robust against outliers like false loop closures [244], which leads to defective solutions or optimization failures. The front-end constructs the pose graph's topology from available sensor data and the back-end has to rely on the correctness of the graph structure. Due to perceptual aliasing in the underlying data association [1], there is no guarantee to provide a pose graph, that is free of outliers. Thus false loop closure constraints might be inserted into the pose graph.
Robust graph optimizer change parts of the graph's topological structure during the optimization process in order to reject false loop closures [245]. The following popular algorithms implement false loop closure detection techniques into exsting graph solver frameworks.

- **Switchable Constraints**[90] is a technique that adds switch variables (being modelled as normally distributed Gaussian variables) in conjunction with a switch function to each loop closure constraint [244, 245]. This changes the pose graph topology, which is now subject to the optimization, by enabling or disabling edge constraints. *Switchable Constraints* is implemented in *VERTIGO*[91] (Versatile Extensions for RobusT Inference using Graph Optimization), which is an extension library for *g2o* and GTSAM.

- **RRR**[92] (Realizing, Reversing, and Recovering) is a loop closure verfication algorithm presented in [152, 153]. It robustly obtains wrong data associations by checking the consistency between loop closures and trajectory estimates [246]. Loop closure constraints are clustered according their appearance. The algorithm then finds the maximum set of clustered edges, which is mutually consistent with

[88] https://bitbucket.org/jesusbriales/cartan-sync/src/master/
[89] https://gitlab.com/jbriales/cartan-sync
[90] https://nikosuenderhauf.github.io/projects/switchableConstraints/
[91] https://github.com/haidai/vertigo
[92] https://github.com/ylatif/rrr

each other and rejects false loop closure edges from the graph [1]. The technique is based on repeatedly solving a graph that contains subsets of the loop closure links. An implementation of *RRR* is published for the *g2o* framework.

- **Dynamic Covariance Scaling** generalizes the *Switchable Constraints* method by deriving an analytical solution for the weighting factor calculations [1]. This technique leads to a significantly faster convergence and reduces the computational overhead. *Dynamic Covariance Scaling* is implemented in the *g2o* framework.

The presented techniques allow pose graph solvers to robustly optimize a given pose graph structure, which contains false data associations within a certain margin. In [154] the authors give a detailed overview concerning the mentioned implementations and compare them under varying false loop closure conditions. Implementations of *Switchable Constraints* and *RRR* are only availabe for older versions of GTSAM and *g2o*, respectively. *Dynamic Covariance Scaling* has become a part of *g2o* and can be activated using the flag *'-robustKernel','DCS'*.

8.3 Pose Graph Generation

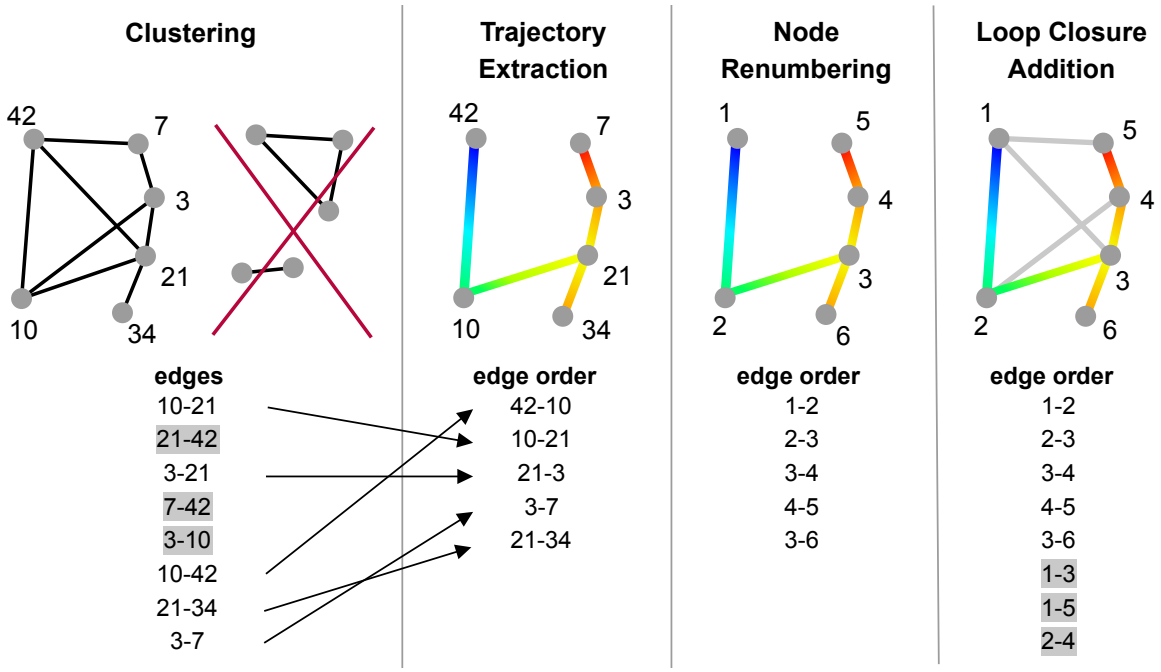

Fig. 8.2: Pose graph generation from non-sequential edge order. Edges are clustered to obtain connected nodes. The cluster with the largest quantity of nodes retains. A spanning tree technique extracts a subgraph, connecting all nodes by a subset of edges. These subset nodes are renumbered to form a squential edge order which represents a trajectory. Finally, all unused edges (grey) from subgraph extraction are also renumbered and added to the end of the trajectory to represent loop closures.

The advantage of SfM over VO and visual SLAM is to work with a non-sequential collection of images. In contrast to that, a graph solver optimizes a trajectory, which

is built up from successive sensor data. Consequently, most of the available solvers require sequential input data, even if they are used in post-processing. They are aimed at processing data in real-time in order to update the pose graph each time a new egde constraint is added. This edge constraint contains either the relative transformation between a new node and an existing one or between two existing nodes.
In the first case, the solver initializes a new node to the graph by using the relative transformation that links to an existing node, which does not need to be the direct predecessor, but must exist in the current graph. This circumstance also allows to split the trajectory into branches as Fig. 8.2**b** depicts. Trajectory branching isn't common in visual SLAM since data usually stems from one sensor setup. However in the proposed SCME pipeline, trajectory branching is a central role as presented in the following.
In the second case, the solver adds the relative transformation as link between two existing nodes, that haven't been directly connected before. This link closes a loop between both nodes and hence it is called loop closure. With the help of loop closures the solver optimizes all nodes within the loop. The larger the loop, the more nodes are optimized.

In order to use a pose graph solver with unordered edge constraints, they must be converted into a pseudo sequence in order to extract a trajectory. A pose graph is a directed graph, since each edge constraint contains the transformation from one node to another node, which however can be easily switched by inverting the relative transformation. The idea is similiar to the extrinsic calibration problem from Section 4.4.5, page 67, which however belongs to a special case since only relative transformations between targets and cameras are given. That circumstance allows to apply a certain scheme to extract the pose graph, which cannot be applied here.

The following four step workflow describes the conversion from unordered edges to a sequential series as Fig. 8.2 illustrates.

Step 1: Clustering

This step finds connected graph nodes and groups them into clusters. The largest cluster of connected edges is retained and remainig edges are removed.

Step 2: Trajectory Extraction

The retained edges are used to extract a subgraph using a spanning tree technique. This subgraph consists of a minimum quantity of edges, that starts from an arbitrary chosen node and connects all other nodes, but contains no cycle/loop closure. It represents a virtual sensor trajectory, which may split into serverals branches.

Step 3: Node Renumbering

Node numbers in the trajectory edge order are mixed up and as a consequence they need to be renumbered into an ascending edge order to form an ongoing sequence. The

node renumbering is saved in an additional list to recover the original node indication.

Step 4: Loop Closure Addition

The remaining edges act as loop closures and their corresponding nodes are also renumbered according the list. They are added to the end of the trajectory edges, in order to guarantee that loop closures won't be listed before corresponding nodes are initialized by the graph solver.

The described workflow allows to convert an unordered collection of relative transformations into a sequential structure that can be processed by pose graph solvers.

8.3.1 Generation of Synthetic Pose Graph Data

The forementioned workflow is proved by creating synthetic pose graph data. The idea is inspired by [33, 32] providing synthetic sample data[93] without ground truth data for testing purposes. In this work the proposed test pipline uses 3d meshes from availabe polygon models as basic data, which are similiar to pose graphs. Mesh data also consist of nodes (vertices) containing translatory information and edges simply linking between nodes. Random generated rotations are added to the nodes since vertex information from meshes do not contain rotation. Each edge is augmented with the relative transformation between the corresponding nodes. Furthermore, Gaussian noise ($\sigma_{\gamma,\beta,\alpha} = 5\%$, $\sigma_{X,Y,Z} = 10\%$, given as relative values) is added to these transformations in order to simulate measuring uncertainties. The corresponding covariance matrices - describing the transformation uncertainties - are obtained from given perturbations of the state variables.
As a result of this, edges represent the desired collection of unordered transformations based on the underlying mesh topology. Their corresponding nodes deal as ground truth pose data for evaluation. Following the four step workflow description from the previous section, a pose graph is generated from a 3d mesh as shown in Fig. 8.3. Further pose graph generation examples can be found in Appendix A.14, page 187. An overview of each pose graph's topological properties summarizes Tab. 8.1.

8.3.2 Optimization of Synthetic Pose Graph Data

The generated pose graphs are optimized using *g2o*, GTSAM, iSAM2, *Ceres* and SLAM++ in order to verify the graph generation pipeline as well as to proof the graph solver integration. The solver settings can be found in Appendix A.15, page 189. After optimization, the pose graphs are compared with known ground truth data. In [262, 284] the authors present distance metrics regarding trajectory comparison between an

[93] https://lucacarlone.mit.edu/datasets/
[94] https://www.models-resource.com/nintendo_64/superMario/model/685/

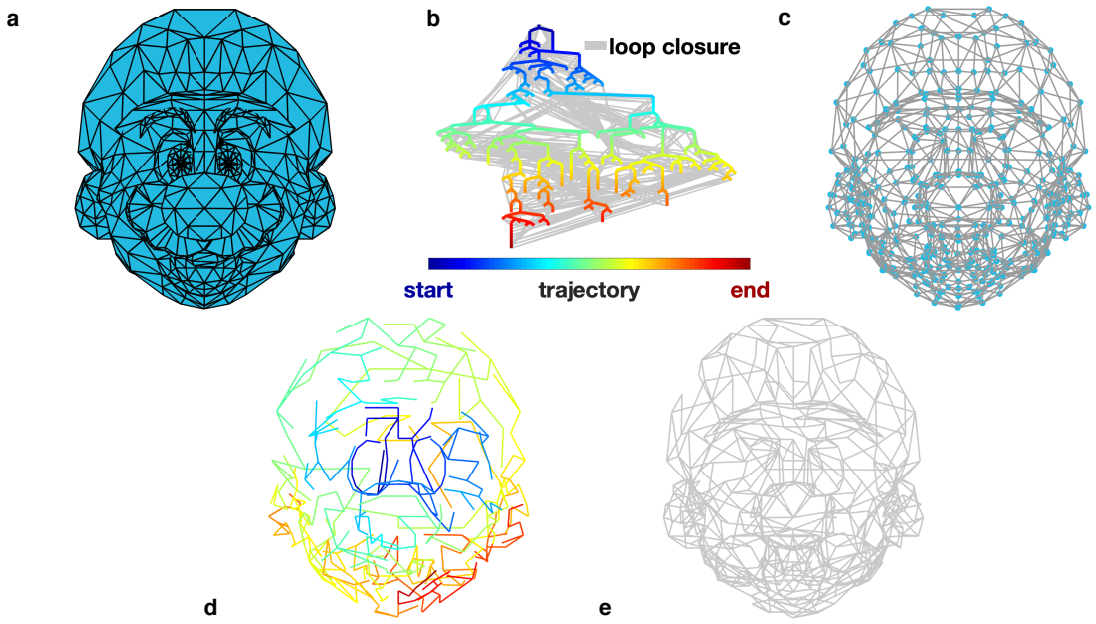

Fig. 8.3: Pose graph extraction from the mesh of the polygon model *Mario*[94]. The following is shown in: **a)** original mesh (textures omitted), **b)** subgraph indicating the trajectory's pathway from **start** to **end** with branching, ▬ illustrate loop closures. **c)** final pose graph, **d** trajectory's pathway in 3d model, **e)** loop closure edges in 3d model.

Tab. 8.1: Properties of generated pose graphs

Model	Vertices	Trajectory Edges	Loop Closure Edges
Mario	440	419	875
Shark	411	410	707
Teapot	732	731	1417
Helix	1620	1619	3201
Sphere	482	481	959

estimated odometry from VO/VIO and ground truth data. Following their recommendation, ICP[95] is used to obtain an initial transformation $[R, \mathbf{t}]$ between estimated graph nodes $[R_i^{\text{est}}, \mathbf{t}_i^{\text{est}}]$ and corresponding ground truth nodes $[R_i^{\text{true}}, \mathbf{t}_i^{\text{true}}]$, which is further non-linear refined in least squares sense

$$\underset{R,\mathbf{t}}{\operatorname{argmin}} \sum_{i=1}^{n} \left\| \mathbf{t}_i^{\text{true}} - R\mathbf{t}_i^{\text{est}} - \mathbf{t} \right\|^2 . \tag{8.6}$$

[95] https://github.com/poisonfox/ICP

The derived transformation is used to obtain the rotation error Δ_R^i (as already used in Eq. (6.17), page 103) and the translation error $\Delta_\mathbf{t}^i$ such that:

$$\Delta_R^i = \cos^{-1}\left(\frac{\text{trace}\left((R_i^{\text{true}})^T R R_i^{\text{est}}\right) - 1}{2}\right) \tag{8.7}$$

$$\Delta_\mathbf{t}^i = \left\|\mathbf{t}_i^{\text{true}} - R\mathbf{t}_i^{\text{est}} - \mathbf{t}\right\|. \tag{8.8}$$

An additional error Δ_l is introduced, describing the difference between ground truth trajetory length l^{true} and estimated one l^{est} with

$$\Delta_l = \left|l^{\text{true}} - l^{\text{est}}\right|. \tag{8.9}$$

The ATE (Absolute Trajectory Error) [284]), is the RMSE (Root Mean Square Error), quantifying the entire estimated trajectory concerning rotation and translation accuracy:

$$\Delta_R^* = \left(\frac{1}{n}\sum_{i=1}^{n}\left\|\Delta_R^i\right\|^2\right)^{\frac{1}{2}} \tag{8.10}$$

$$\Delta_\mathbf{t}^* = \left(\frac{1}{n}\sum_{i=1}^{n}\left\|\Delta_\mathbf{t}^i\right\|^2\right)^{\frac{1}{2}}. \tag{8.11}$$

An overview of obtained Δ_R^*, $\Delta_\mathbf{t}^*$ and Δ_l results for each solver and each model is given in Tab. 8.2. Fig. 8.4 graphically illustrates each solver's solution compared to ground truth data for the selected model *Shark*. Appendix A.16, page 190 contains further visualized PGO results of the remaining models. As can be seen in Fig. 8.4

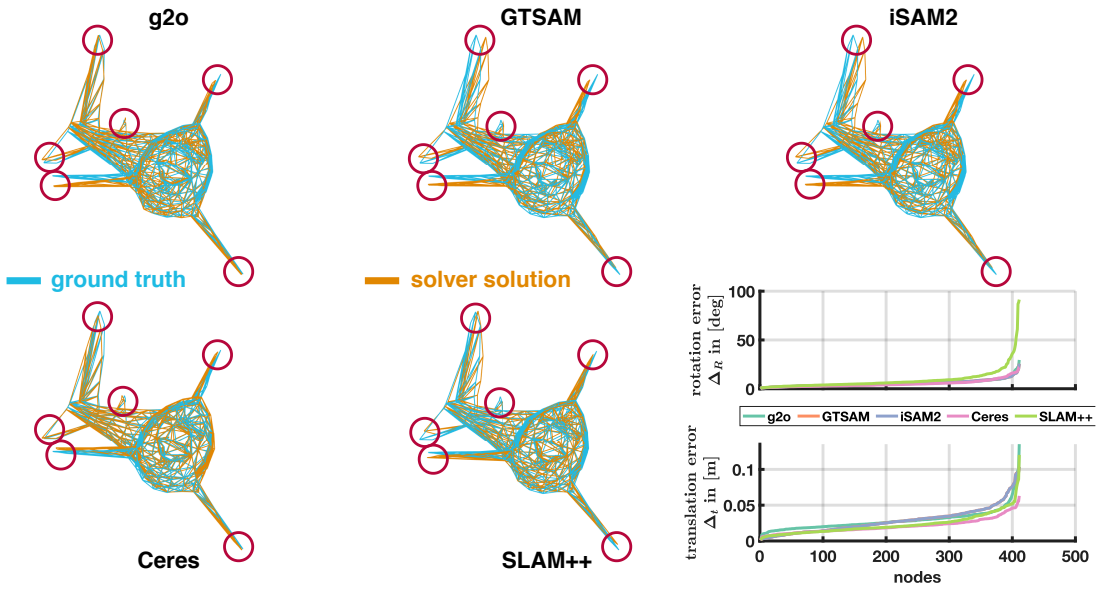

Fig. 8.4: Visualization of PGO results from pose graph *Shark*. This example is best suitable to hightlight salient points for a visual accuracy comparison. Plots on the right visualize Δ_R^i and $\Delta_\mathbf{t}^i$, which are sorted in ascending order to better compare minima and maxima values between the solvers.

Tab. 8.2: Pose graph solver optimization results for synthetic test data

Solver	**Mario** ($l_{\text{ref}} = 228.10m$)			**Shark** ($l_{\text{ref}} = 74.12m$)			**Teapot** ($l_{\text{ref}} = 222.54m$)			**Helix** ($l_{\text{ref}} = 5292.29m$)			**Sphere** ($l_{\text{ref}} = 85.06m$)		
	Δ^*_R [°]	$\Delta^*_\mathbf{t}$ [m]	Δ_l [m]	Δ^*_R [°]	$\Delta^*_\mathbf{t}$ [m]	Δ_l [m]	Δ^*_R [°]	$\Delta^*_\mathbf{t}$ [m]	Δ_l [m]	Δ^*_R [°]	$\Delta^*_\mathbf{t}$ [m]	Δ_l [m]	Δ^*_R [°]	$\Delta^*_\mathbf{t}$ [m]	Δ_l [m]
g2o	4.717	0.055	0.829	6.485	0.032	0.197	6.652	0.063	0.318	6.353	1.909	37.897	4.576	0.022	0.167
GTSAM	5.022	0.071	0.238	6.393	0.034	0.264	6.918	0.075	0.665	5.643	2.130	38.853	4.530	0.021	0.118
iSAM2	5.019	0.070	0.234	6.392	0.033	0.263	6.907	0.075	0.664	5.642	2.131	38.843	4.529	0.021	0.119
Ceres	4.811	0.045	0.424	6.204	0.023	0.443	5.159	0.072	1.999	6.030	1.202	24.454	4.645	0.019	0.535
SLAM++	5.214	0.043	0.163	13.771	0.027	0.324	8.440	0.094	0.812	4.582	1.697	13.213	5.767	0.020	0.620

the solvers obtain different optimization results, which are noticable at certain points. However each solver's accuracy concerning Δ^*_R, $\Delta^*_\mathbf{t}$ and Δ_l varies from model to model. There is no solver that cleary outperforms the other ones. *Ceres* in conjunction with MASAT provides overall satisfying results, followed by *g2o*. Accuracy results from GTSAM and iSAM2 are very close and do not distinguish noticeably. iSAM2 is mainly developed for faster processing in an online framework like visual SLAM. However this advantage plays a secondary role in the here proposed *offline* reconstruction pipeline. SLAM++ provide mixed accuracy results. It produces significant larger rotational errors except for the *Helix* pose graph. It can be supposed that SLAM++ provides better results concerning rotation for larger and densely connected pose graphs.
Furthermore, the presented results strongly depend on solver settings as well as alignment between optimized pose graphs and ground truth.

This analysis is mainly done to proof the proposed trajectory extraction workflow as well as to test the integration of standalone solvers by verifying optimization results.

Brief Chapter Summary

SCME incorporates PGO as back-end. Standalone solvers for PGO usually work in a Visual SLAM framework and thus require sequential pairwise transformation constraints from a camera trajectoy as input data. In contrast to that, the proposed SCME pipeline is designed to process unordered image collections and provides non-sequential pairwise transformations from scaled two-view transformations. A workflow was presented that extracts a camera trajectory from a set of given unordered transformation constraints. The workflow was applied to synthetic data from polygon models. Extracted pose graphs were solved via standalone PGO solvers to test the trajectory extraction workflow as well as the solver integration into the SCME pipeline.

9 Structureless Camera Motion Estimation

Brief Chapter Overview

This chapter explaines the proposed SCME (Structureless Camera Motion Estimation) pipeline and Section 9.1 introduces the pipeline's main elements, which include serveral findings from chapter Chapter 3 to chapter Chapter 8. Section 9.2 explains a novel approach to obtain global translation scale factors from up-to-scale two-view transformations using translation ratios from Chapter 7. This scaling approach is the last missing element that is required to incorporate PGO as global camera pose solver. Furthermore, the proposed scaling approach also integrates depth data from RGBD cameras as Section 9.3 comprehensively describes and extrinsic camera constraints, which is described in Section 9.4.

9.1 SCME Pipeline

SCME is a processing pipeline that combines existing ideas and novel insights to provide a general, modular and flexible camera motion estimation approach. It supports a wide range of central perspective and central omnidirectional projections via P2S-maps (general), incorporates different feature types and pose graph solvers (modular) and is not restricted to specific multi-camera configurations with overlapping FoVs or to certain camera combinations (flexible).

SCME is in the spirit of *global* SfM (Section 2.2, page 19) and solves for global camera poses utilizing up-to-scale two-view transformations from feature matching as Fig. 9.1 illustrates. Unlike extising approaches, SCME is non-incremental and separates global two-view translation scaling from global rotation estimation. Consequently translation scaling becomes independent of global rotation estimates and this circumstance enables SCME to solve for all global translation scales simultaneously without using any triangulation. This process is the core element of SCME that is split into two parts. The first one determines translation ratio factors from available pairs of up-to-scale two-view transformations and their point correspondences (Section 7.5, page 134). The second one obtaines all global two-view translation scale factors using derived translation ratio factors (Section 9.2, page 155).

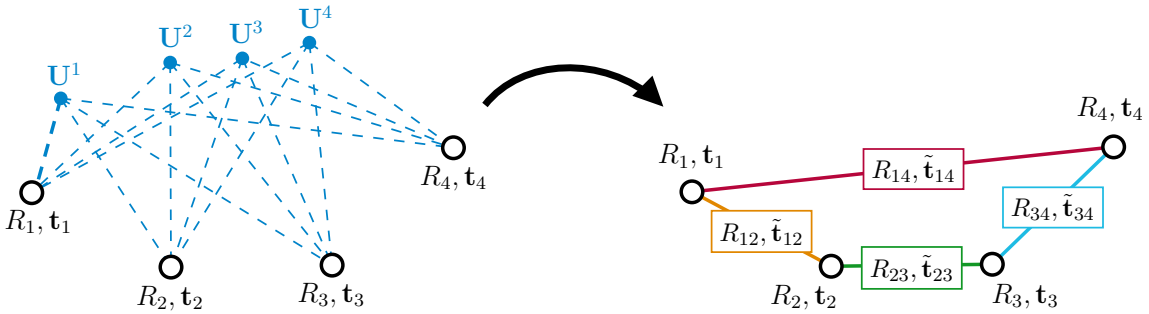

Fig. 9.1: Simple illustration of the main idea behind SCME that replaces the 3d structure $\mathbf{U}^1$, $\mathbf{U}^2$, $\mathbf{U}^3$, $\mathbf{U}^4$ in BA with globally scaled two-view transformations $[R_{12}, \tilde{\mathbf{t}}_{12}]$, $[R_{23}, \tilde{\mathbf{t}}_{23}]$, $[R_{34}, \tilde{\mathbf{t}}_{34}]$, $[R_{14}, \tilde{\mathbf{t}}_{14}]$ to solve for global camera poses using PGO.

Furthermore, SCME incorporates state-of-the-art solvers for PGO that obtain global camera poses based on provided globally scaled two-view transformations. As a consequence the proposed pipeline is an intermediate calculation step, since the optimization does not include the actual measured image points. This circumstance is comparable to Visual SLAM, where PGO optimizes camera poses in case of loop closures but global BA is still required for further optimization in post-processing. SCME also provides camera motion estimates that are close to the final solution and may deal as initial guess for further optimization using structureless BA. In order to use BA for optimization, 3d points need to be triangulated from global camera pose estimates and known image point correspondences.

Fig. 9.2 gives an overview of the presented pipeline that uses P2S-maps to incorporate a wide range of central projections from **intrinsic camera calibration** and from stitched full **omnidirectional images**, e.g. in standard equirectangular format. The pipeline provides **projection conversion** that allows to convert images from the source projection to a target projection with less distortion effects to improve feature matching. **Feature matching and transformation optimization** provide up-to-scale two-view transformations $[R_{kl}, \hat{\mathbf{t}}_{kl}]$, $[R_{lm}, \hat{\mathbf{t}}_{lm}]$, $[R_{mk}, \hat{\mathbf{t}}_{mk}]$ and point correspondences $\hat{\mathbf{X}}_k \leftrightarrow \hat{\mathbf{X}}_l$, $\hat{\mathbf{X}}_l \leftrightarrow \hat{\mathbf{X}}_m$, $\hat{\mathbf{X}}_m \leftrightarrow \hat{\mathbf{X}}_k$ from matched image pairs $k \leftrightarrow l$, $l \leftrightarrow m$, $m \leftrightarrow k$. **Ratio factor determination** is a novel method determining translation ratio factors r_{klm}, r_{lkm}, r_{kml} between adjoining two-view transformations and point correspondences as introduced in Section 7.5, page 134. They describe the relation of translation magnitudes between adjoining two-view transformations in global as well as in real-world scale. **Scale factor determination** uses these translation ratio factors to calculate the global translation scale factors $\tilde{s}_{kl}$, $\tilde{s}_{lm}$, $\tilde{s}_{mk}$. The derived results globally scale two-view transformations which then PGO uses to solve for global camera poses $[R_k, \mathbf{t}_k]$, $[R_l, \mathbf{t}_l]$, $[R_m, \mathbf{t}_m]$. Furthermore, SCME allows to integrate **optional constraints** such as depth values from RGBD cameras and extrinsics from multi-camera calibration to obtain real-world scale. Depth data - e.g. from the l^{th} camera - is used to obtain the real-world scale factors s_{lk}, s_{lm} directly as introduced in

Section 6.7, page 113, which then can be integrated into the **scale factor calculation** process. Extrinsic calibration - e.g. between the k^{th} and m^{th} camera - provides the normalized two-view transformation $[R_{mk}, \hat{\mathbf{t}}_{mk}]$ and the corresponding real-world scale factor s_{mk}. The latter one can be also integrated into the **scale factor calculation**. The up-to-scale two-view transformation $[R_{mk}, \hat{\mathbf{t}}_{mk}]$ is already optimized. Thus it is used to verify point correspondences $\hat{\mathbf{X}}_m \leftrightarrow \hat{\mathbf{X}}_k$ from feature matching and is part of the **ratio factor determination** process.

For a better understanding the following describes the translation scaling for a general case and at hand of the shown example in Fig. 9.1, which is taken from Chapter 2. This example does not use all available two-view transformations between unknown global camera poses in order to show how global translation scale factors can be derived from sparse image matching.

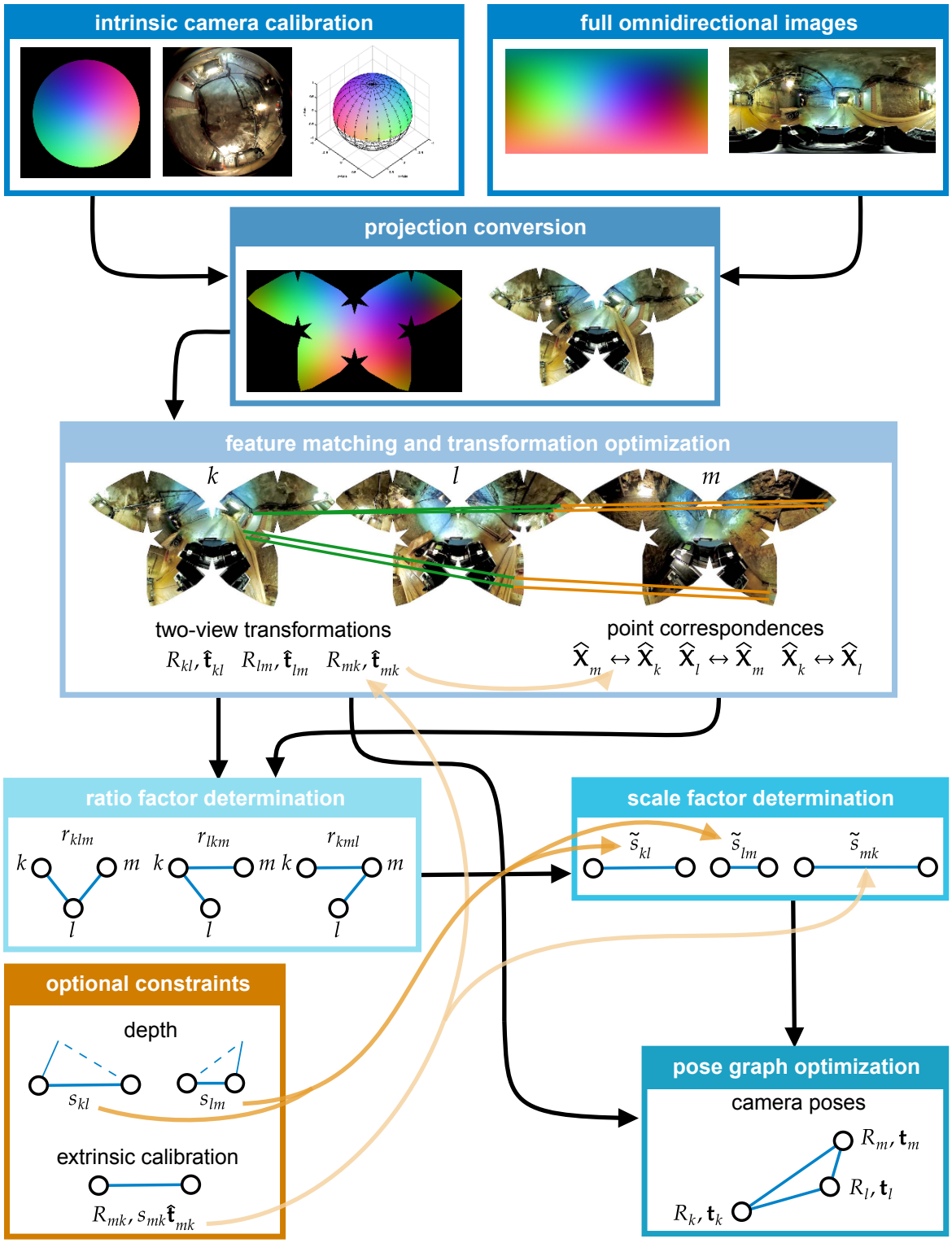

Fig. 9.2: Overview of the proposed SCME pipeline. Depth constraints from RGBD cameras as well as extrinsic calibration constrains from multi-camera rigs are optional and can be used to achieve real-world scaling.

9.2 Determination of Two-View Translation Scale Factors

As described in Section 6.7, page 113 the two-view translation $\mathbf{t}_{kl}$ between the k^{th} and l^{th} camera is a combination of an normalized translation $\|\hat{\mathbf{t}}_{kl}\| = 1$ and a real-world scale factor s_{kl} with

$$\mathbf{t}_{kl} = s_{kl}\hat{\mathbf{t}}_{kl}\,. \tag{9.1}$$

Since a monocular image-based reconstruction is always up to a certain scale, the obtained global scale factor $\tilde{s}_{kl}$ is also up to a certain scale such that

$$s_{kl} = \tau\tilde{s}_{kl}\,. \tag{9.2}$$

τ represents an aligning factor that adjusts the entire reconstruction to real-world dimensions. This is comparable to SfM where known reference points are used for scaling purpose. In this work τ can be obtained from known relative real-world constraints as shown in Sections 9.3 and 9.4. Applying Eq. (9.2) to Eq. (9.1) yields

$$\mathbf{t}_{kl} = \tau\tilde{s}\hat{\mathbf{t}}_{kl}\,. \tag{9.3}$$

Section 7.5, page 134 introduces the translation ratio factor r, whose relations are summarized:

$$r_{klm} = \frac{\|\mathbf{t}_{kl}\|}{\|\mathbf{t}_{lm}\|} = \frac{s_{kl}\|\hat{\mathbf{t}}_{kl}\|}{s_{lm}\|\hat{\mathbf{t}}_{lm}\|} = \frac{\tau\tilde{s}_{kl}\|\hat{\mathbf{t}}_{kl}\|}{\tau\tilde{s}_{lm}\|\hat{\mathbf{t}}_{lm}\|}$$

with $\|\hat{\mathbf{t}}_k\|{=}\|\hat{\mathbf{t}}_l\| = 1$ such that

$$r_{klm} = \frac{\|\mathbf{t}_{kl}\|}{\|\mathbf{t}_{lm}\|} = \frac{s_{kl}}{s_{lm}} = \frac{\tilde{s}_{kl}}{\tilde{s}_{lm}} \tag{9.4}$$

and finally yields

$$0 = r_{klm}\tilde{s}_{lm} - \tilde{s}_{kl}\,. \tag{9.5}$$

The scale factors are independent of the two-view transformation direction, thus $\tilde{s}_{kl} = \tilde{s}_{lk}$ and $\tilde{s}_{lm} = \tilde{s}_{ml}$. Furthermore, r_{klm}, r_{lkm}, r_{kml} describe the three translation ratio factors within a camera triplet k, l, m as Fig. 9.3 clarifies.

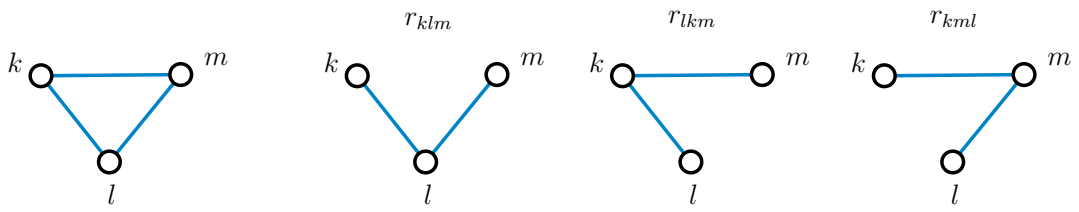

Fig. 9.3: The camera triplet k, l, m consists of three tranlsation ratio factors r_{klm}, r_{lkm}, r_{kml}.

The proposed translation scaling method requires at least one translation ratio factor from each available camera triplet and thus is suitable for sparse image matching.

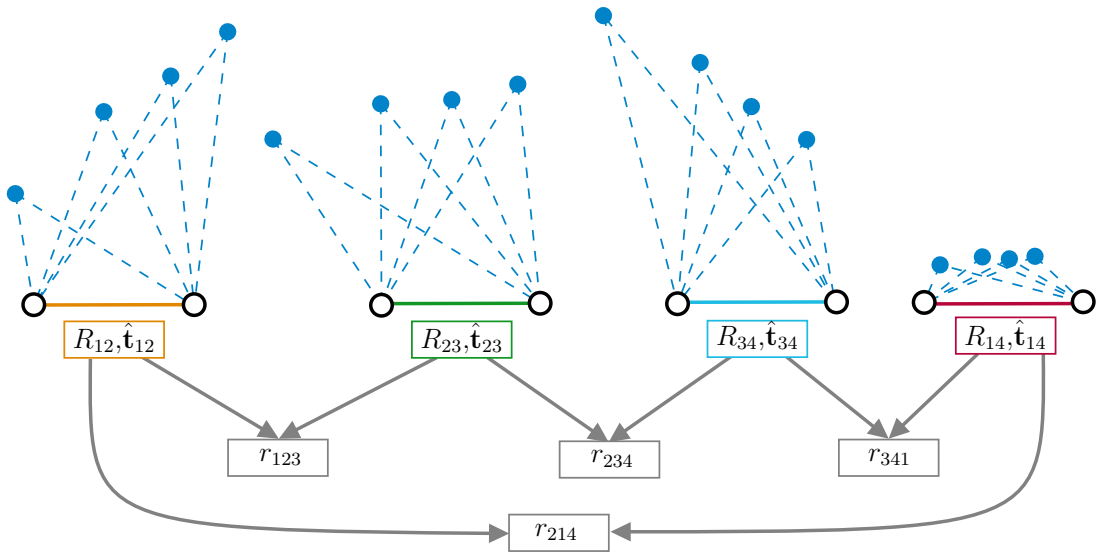

Fig. 9.4: The translation ratio factors r_{214}, r_{123}, r_{143}, r_{234} are obtained from given two-view transformations $[R_{12}, \hat{\mathbf{t}}_{12}]$, $[R_{23}, \hat{\mathbf{t}}_{23}]$, $[R_{34}, \hat{\mathbf{t}}_{34}]$, $[R_{14}, \hat{\mathbf{t}}_{14}]$ and image point correspondences $\hat{\mathbf{X}}_1 \leftrightarrow \hat{\mathbf{X}}_2$, $\hat{\mathbf{X}}_2 \leftrightarrow \hat{\mathbf{X}}_3$, $\hat{\mathbf{X}}_3 \leftrightarrow \hat{\mathbf{X}}_4$, $\hat{\mathbf{X}}_1 \leftrightarrow \hat{\mathbf{X}}_4$, which are obtained trough feature matching. As 3d points do not play any role in the entire SCME pipeline they are indicated here to better visualize the image point correspondences and the relation between up-to-scale two-view transformations.

Fig. 9.4 illustrates the set of translation ratio factors $r_{214}, r_{123}, r_{143}, r_{234}$ which are obtained from available up-to-scale two-view transformations $[R_{12}, \hat{\mathbf{t}}_{12}]$, $[R_{23}, \hat{\mathbf{t}}_{23}]$, $[R_{34}, \hat{\mathbf{t}}_{34}]$, $[R_{14}, \hat{\mathbf{t}}_{14}]$ and point correspondences $\hat{\mathbf{X}}_1 \leftrightarrow \hat{\mathbf{X}}_2$, $\hat{\mathbf{X}}_2 \leftrightarrow \hat{\mathbf{X}}_3$, $\hat{\mathbf{X}}_3 \leftrightarrow \hat{\mathbf{X}}_4$, $\hat{\mathbf{X}}_1 \leftrightarrow \hat{\mathbf{X}}_4$ as explained in Section 7.5, page 134. Considering Eq. (9.5), the derived translation ratio factors and unknown global scale factors form a system of linear equations:

$$
\begin{aligned}
r_{214}\tilde{s}_{14} - \tilde{s}_{12} &= 0 \\
r_{123}\tilde{s}_{23} - \tilde{s}_{12} &= 0 \\
r_{143}\tilde{s}_{34} - \tilde{s}_{14} &= 0 \\
r_{234}\tilde{s}_{34} - \tilde{s}_{23} &= 0\,,
\end{aligned}
$$

which is solved using a DLT that finds the solution for $B\mathbf{x} = \mathbf{0}$ in least squares sense subjecting $\|\mathbf{x}\| = 1$. Rearranging the equations leads to

$$
\underbrace{\begin{bmatrix} -1 & r_{214} & 0 & 0 \\ -1 & 0 & r_{123} & 0 \\ 0 & -1 & 0 & r_{143} \\ 0 & 0 & -1 & r_{234} \end{bmatrix}}_{B}
\underbrace{\begin{pmatrix} \tilde{s}_{12} \\ \tilde{s}_{14} \\ \tilde{s}_{23} \\ \tilde{s}_{34} \end{pmatrix}}_{\mathbf{x}}
= \mathbf{0}\,. \tag{9.6}
$$

Applying an SVD to B with $B = U\Sigma V^T$ determines V. The column $V_{(:,\min)}$ corresponds to the smallest singular value $\sigma_{\min}$ in Σ and represents the solution $\mathbf{x} = V_{(:,\min)}$. The derived global scale factor solutions are used as initial estimates for a non-linear LM optimization, which is presented here in a general expression

$$\underset{\tilde{s}_{kl},\tilde{s}_{lm}}{\operatorname{argmin}} \sum_{(k,l,m\in\mathcal{E})} \left(\frac{\tilde{s}_{kl}}{\tilde{s}_{lm}} - r_{klm} \right)^2 \tag{9.7}$$

with

$$\begin{aligned} J_{klm} &= \begin{bmatrix} \dfrac{\partial}{\partial \tilde{s}_{kl}} & \dfrac{\partial}{\partial \tilde{s}_{lm}} \end{bmatrix} \\ &= \begin{bmatrix} \dfrac{1}{\tilde{s}_{lm}} & -\dfrac{\tilde{s}_{kl}}{(\tilde{s}_{lm})^2} \end{bmatrix} \end{aligned} \tag{9.8}$$

considering $\tilde{s}_{kl} = \tilde{s}_{lk}$ and $\tilde{s}_{lm} = \tilde{s}_{ml}$. That expression applied to the described example yields

$$\underset{\tilde{s}_{12},\tilde{s}_{14},\tilde{s}_{23},\tilde{s}_{34}}{\operatorname{argmin}} \|\boldsymbol{\epsilon}\|^2$$

with

$$\boldsymbol{\epsilon} = \begin{pmatrix} \dfrac{\tilde{s}_{12}}{\tilde{s}_{14}} - r_{214} \\ \dfrac{\tilde{s}_{12}}{\tilde{s}_{23}} - r_{123} \\ \dfrac{\tilde{s}_{14}}{\tilde{s}_{34}} - r_{143} \\ \dfrac{\tilde{s}_{23}}{\tilde{s}_{34}} - r_{234} \end{pmatrix}$$

and the corresponding Jacobian

$$J = \frac{\partial \boldsymbol{\epsilon}}{\partial \mathbf{x}} = \begin{bmatrix} \dfrac{1}{\tilde{s}_{14}} & -\dfrac{\tilde{s}_{12}}{(\tilde{s}_{14})^2} & 0 & 0 \\ \dfrac{1}{\tilde{s}_{23}} & 0 & -\dfrac{\tilde{s}_{12}}{(\tilde{s}_{23})^2} & 0 \\ 0 & \dfrac{1}{\tilde{s}_{34}} & 0 & -\dfrac{\tilde{s}_{14}}{(\tilde{s}_{34})^2} \\ 0 & 0 & \dfrac{1}{\tilde{s}_{34}} & -\dfrac{\tilde{s}_{23}}{(\tilde{s}_{34})^2} \end{bmatrix}.$$

Using translation ratio uncertainties $\sigma_{r_{klm}}$ from Section 7.5.1, page 134 leads to the minimization of the sum of squared Mahalanobis distances

$$\underset{\tilde{s}_{kl},\tilde{s}_{lm}}{\operatorname{argmin}} \sum_{(k,l,m\in\mathcal{E})} \left(\frac{\tilde{s}_{kl}}{\tilde{s}_{lm}} - r_{klm} \right) \frac{1}{\sigma^2_{r_{klm}}} \left(\frac{\tilde{s}_{kl}}{\tilde{s}_{lm}} - r_{klm} \right). \tag{9.9}$$

The obtained results globally scale the given two-view transformations that are now suitable to be solved via PGO for global camera poses as Chapter 8 explains.

9.3 Integration of Depth Data

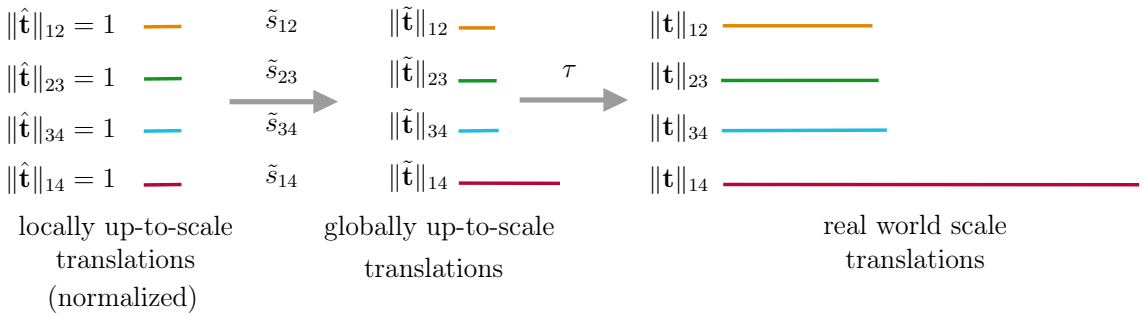

Fig. 9.5: Illustration of different two-view translation scale levels. Normalized translation scale factors are provided by feature matching. Globally scaled translations are obtain from translation ratio factors. Real-world scaled translations are derived by including depth information from RGBD cameras or by including extrinsic camera constraints.

The results from the previous section do not match real-world scale. The following shows how depth data - e.g. from an RGBD camera - can be integrated into the **scale factor calculation** process straightforwardly. This example assumes the 2nd camera to be an RGBD camera. The provided depth data is used to obtain the real-world

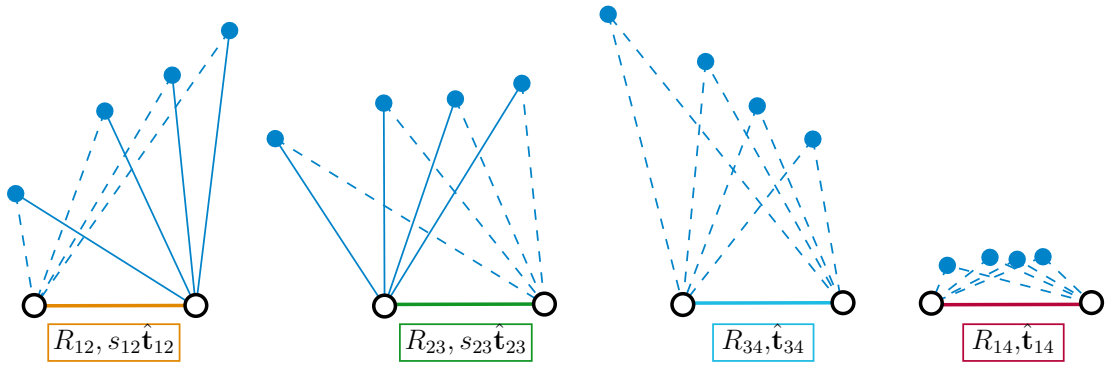

Fig. 9.6: Available two-view transformation from feature matching are up-to-scale. Since camera 2 provides depth information the two-view transformations $[R_{12}, \hat{\mathbf{t}}_{12}]$, $[R_{23}, \hat{\mathbf{t}}_{23}]$ are real-world scaled by s_{12}, s_{23}.

scale factors s_{12}, s_{23} using the proposed two-view translation scaling approach from Section 6.7, page 113.

Following the workflow from the previous section inital ratio factors $r_{214}, r_{123}, r_{143}, r_{234}$ and global scale factor estimates $\tilde{s}_{12}$, $\tilde{s}_{14}$, $\tilde{s}_{23}$, $\tilde{s}_{34}$ are obtained. The aligning factor τ is calculated from the correspondence set between global scale factors $\tilde{s}_{12}, \tilde{s}_{23}$ and

real-world scale factors s_{12}, s_{23} such that

$$\tau = \frac{\mathbf{a}^T\mathbf{b}}{\|\mathbf{a}\|^2}$$

with $\mathbf{a} = (\tilde{s}_{12}, \tilde{s}_{23})^T$ and $\mathbf{b} = (s_{12}, s_{23})^T$. The remaining real-world scale factors $s_{14} = \tau\tilde{s}_{14}$ and $s_{34} = \tau\tilde{s}_{34}$ are simply calculated. They are further non-linear optimized using Eq. (9.7). Since s_{12}, s_{23} are already optimized in real-world scale, they require to remain unchanged during LM optimization. This is achieved by zeroing the corresponding column entries in the Jacobian such that

$$J = \begin{bmatrix} 0 & -\frac{\tilde{s}_{12}}{(\tilde{s}_{14})^2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{\tilde{s}_{34}} & 0 & -\frac{s_{14}}{(s_{34})^2} \\ 0 & 0 & 0 & -\frac{s_{23}}{(s_{34})^2} \end{bmatrix}.$$

This procedure is inspired by sparse BA [172, 173], where certain parameters can be excluded from optimization, e.g. to only optimize motion or structure.

9.4 Integration of Extrinsic Camera Constraints

Assuming the 1st and the 2nd camera to be fixed in a stereo setup. Extrinsic camera calibration (Section 4.4, page 58) provides the transformation $[R_{12}, \mathbf{t}_{12}]$ between both cameras, which can be decomposed into a normalized transformation $[R_{12}, \hat{\mathbf{t}}_{12}]$ and a corresponding real-world translation scale factor s_{12}. Thus the real-world translation scale factor s_{12} can be intergrated into the translation **scale factor calculation** process as described in the previous section.

The transformation $[R_{12}, \hat{\mathbf{t}}_{12}]$ verifies feature matches $\hat{\mathbf{X}}_1 \leftrightarrow \hat{\mathbf{X}}_2$ and is used for the translation **ratio factor determination** process. As can be seen, depth information as well as extrinsic camera constraints in various combinations can be straightforwardly integrated into the translation scaling process.

Brief Chapter Summary

This chapter presented the complete SCME pipeline and comprehensively explained a novel, non-incremental approach that globally scales up-to-scale two-view transformations to be used with PGO. Unlike existing approaches, this new approach is independent of global rotation estimates or triangulated 3d points. Translation ratio factors are obtained between adjoining up-to-scale two-view transformations and their point correspondences. Based on these ratio factors, global translation scale factors are estimated and further optimized. Depth data from RGBD cameras as well as ex-

trinsic camera constraints can be simply integrated to achieve real-world scaling. The proposed modular pipeline has a simple structure and provides a general and flexible camera motion estimation approach, which incorporates a wide range of central directional and omnidirectional projections.

10 Camera Motion Estimation Results

SCME incorporates different image feature types such as SIFT (Scale Invariant Feature Transform) [174], SURF (Speeded Up Robust Features) [12], BRISK (Binary Robust Invariant Scalable Keypoints) [159], ORB (Oriented FAST and Rotated BRIEF) [212] and KAZE (Japanese word for *Wind*) [6]. This work utilizes *Root*SIFT [8] as a robust variant of SIFT.
Unlike many other SfM, VO or Visual SLAM implementations, SCME uses these mentioned feature types jointly as proposed in [63] and in any combination. This allows to make use of different feature types at the same time without selection a preferred/suitable one in advance. It is an elegant way to combine the feature's strenghts as well as to reduce their weaknesses.

The SCME pipeline is tested with image data from an underground mapping compaign of an old ore mine using a mobile robot, which is equipped with a *Kinect V2*, *Kodak SP 360 4K* and a *Ricoh Theta S* camera. The corresponding calibrations are shown in Section 4.3.1, page 55. The scanned region is approx. $65m$ long amd consists of narrow cross sections, a flat underground, external illumination and provides a loop closure to test SCME in a less challenging environment for the first time.

Image acquisition is taken during different measuring runs for each camera, thus obtained trajectories from camera motion cannot be directly compared between mentioned cameras.

10.1 Directional Camera Images

Kinect V2 RGB

SCME uses RGB data from the *Kinect V2* camera consisting of 734 images and data association relies on BRISK, ORB and SIFT features. The estimated camera motion is compared to results from standard SfM implementations like *Metashape*, *Colmap* and *Visual* SfM as Fig. 10.1 illustrates. *Metashape's* trajectory is set as reference and the remaining ones are transformed via ICP (Iterative Closest Point) with scaling capabilities. The SfM implementations obtain similar results and are also able to manage the loop closure. SCME also obtaines a closed model, but the trajectory shape differs noticeably from the SfM solutions.

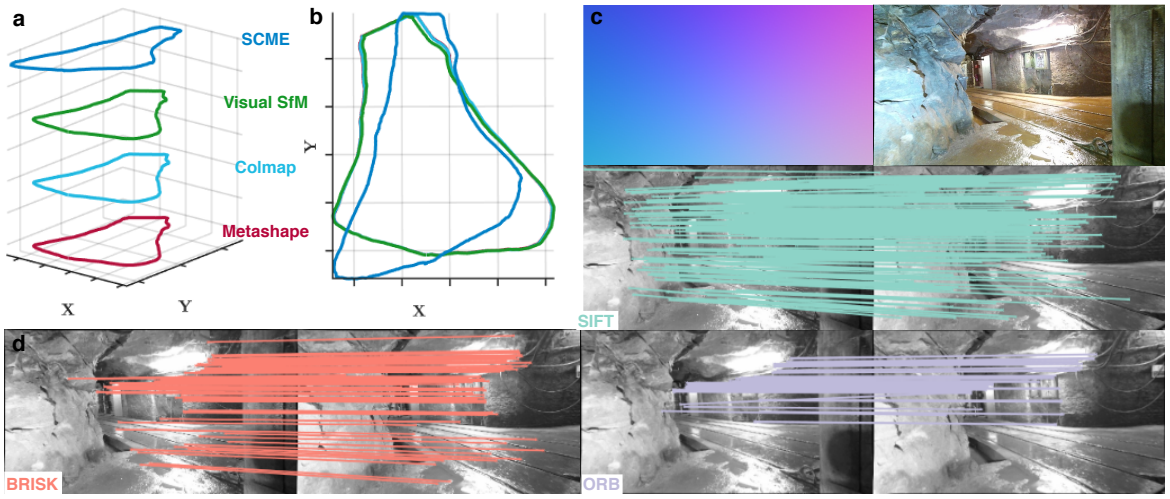

Fig. 10.1: SCME results from directional RGB images. The following is shown: **a)** camera motion (trajectory) obtained from *Metashape*, *Colmap*, *Visual* SfM and SCME, **b)** top view of camera motion results, **c)** camera calibration as P2S-map with an example image, **d)** point correspondences between two images from geometric verified BRISK, ORB and SIFT feature matches.

Kinect V2 RGBD

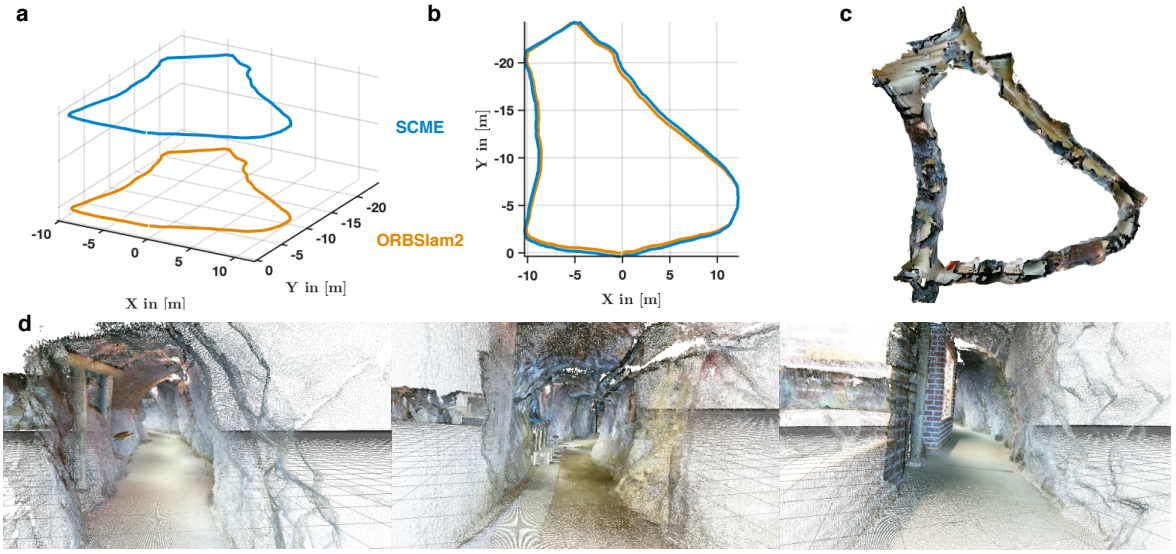

Fig. 10.2: SCME results from directional RGBD images. The following is shown: **a)** camera motion (trajectory) obtained from *ORBSlam2* and SCME, **b)** top view of camera motion results, **c)** top view of resulting point cloud from point cloud fusion via *Open3D*, which is obtained from SCME pose estimates and corresponding RGBD data without further pose optimization, **d)** selected insight views of the point cloud.

The forementioned RGB image set is used in combination with depth data. SCME utilizes this depth data to scale two-view transformations but do not use them for image alignment. Thus up-to-scale two-view transformations are used from the previous test and the additional depth data are used to calculate the real-world translation scale factors directly. *ORBSlam2*[96] [190] is a popular Visual SLAM algorithm and obtains

[96] https://github.com/raulmur/ORB_SLAM2

a reference trajectory as shown in Fig. 10.2. As can be seen both trajectories are nearly identicial, which shows that SCME obtaines comparable results for the RGBD data. Thus it can be stated that SCME calculates up-to-scale two-view transformations correctly from RGB images and also determines global scale factors correctly from depth data. In order to show SCME's accuracy concerning pose estimates, *Open3D*[97] [290] fuses RGBD data based on their corresponding camera poses to build an entire point cloud model of the scanned underground section. Fig. 10.2**c, d)** show sample views from the resulting points cloud indicating accurate results.

10.2 Omnidirectional Camera Images

Ricoh Theta S

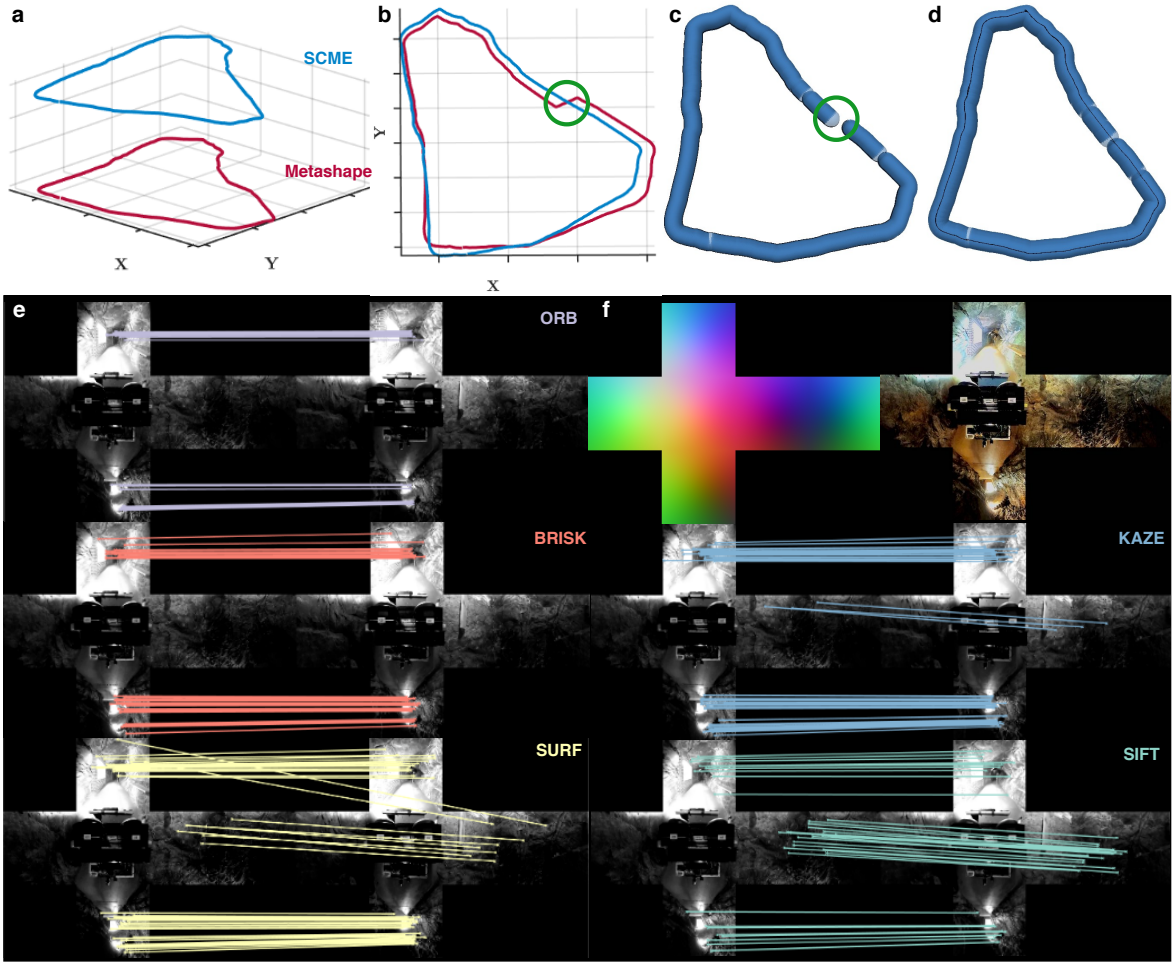

Fig. 10.3: SCME results from full omnirectional RGB images. The following is shown: **a)** camera motion (trajectory) obtained from *Metashape* and SCME, **b)** top view of camera motion results, where (**O**) highlights the break in the *Metashape* model leading to an open loop, **c)** open loop trajectory in *Metashape*, **d)** SCME's trajectory imported to *Metashape*, **e)** point correspondences between two images from geometrically verified ORB, BRISK, KAZE, SURF and SIFT feature matches, **f)** cube map as P2S-map with an example image.

[97] http://www.open3d.org

This camera provides stitched full omnidirectional images in equirectangular format, which are converted to a cube map projection. Equirectangular and cube map P2S-maps are generated via the map projection generator from Section 5.3, page 79 and images are converted using the projection conversion workflow as proposed in Section 5.4, page 81. A total of 1473 images are captured. Data association is based on all available feature types namely ORB, BRISK, KAZE, SURF and SIFT. Fig. 10.3 illustrates the results of the obtained camera poses between *Metashape* and SCME. As can be seen both achieve comparable results, however *Metashape*'s trajectory is non-closed. The open loop trajectory is in contrast to *Metashape*'s behavior in case of directional images. SfM in case of directional images can balance alignment inaccuracies by moving the camera center or by adjusting intrinsics, e.g slightly change focal length. In case of omnidirectional images this process becomes more difficult. Omnidirectional images in equirectangular format have a predefined intriniscs comparable to P2S-maps. Thus all intrinsic parameters remain unchanged since the geometry is fixed. Furthermore, omnidirectional images observe features from all directions and consequently there is no clear decision where the camera should move during optimization.

In contrast to *Metashape* SCME is able to reconstruct a closed camera trajectory and yields more confident results.

Kodak SP360 4k

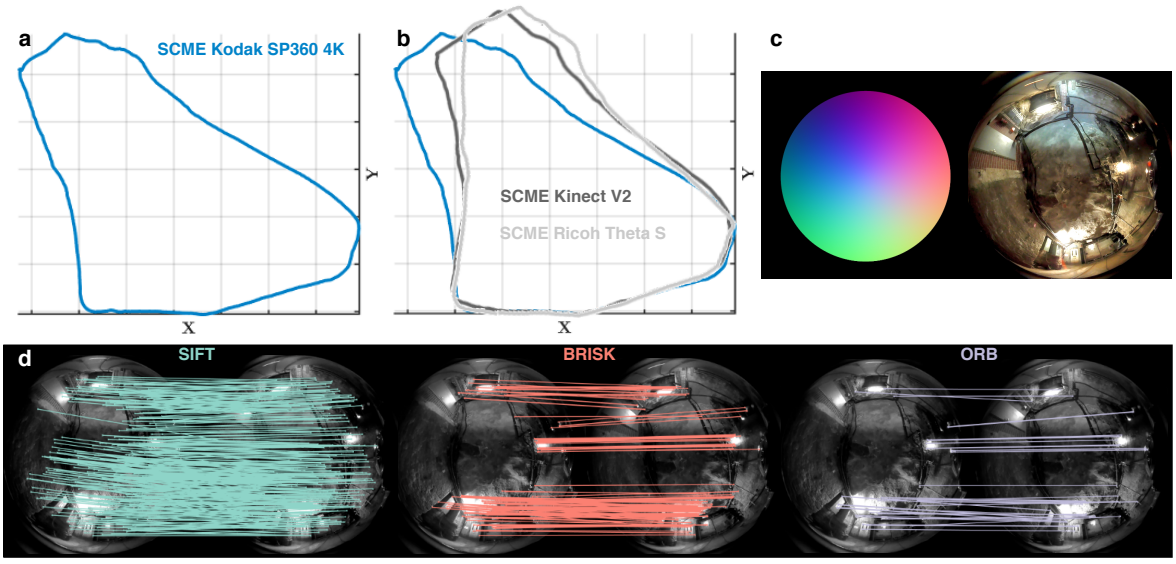

Fig. 10.4: SCME results from omnirectional (fisheye) RGB images. The following is shown: **a)** camera motion (trajectory) obtained from SCME, **b)** top view of camera motion results obtained from SCME using *Kodak SP360 4K*, *Kinect V2* and *Ricoh Theta S*. Please note, camera trajectories are from different runs. Due to the norrow environment the shape of the trajectories is similar, which is used to align the data for a visual comparison. **c)** camera calibration as P2S-map with an example image, **d)** point correspondences between two images from geometric verified SIFT, BRISK and ORB feature matches.

The *Kodak SP360 4k* is a fisheye camera that is intrinsically calibrated using the

described routine from Section 4.3, page 54. It provides 1033 images and relies on a feature combination consisting of BRISK, ORB and SIFT. SCME uses the original images with strong distortion effects. This camera is used to show SCME's compatibility and it is not used with any other SfM or or Visual SLAM implementation. The motion estimation results are illustrated in Fig. 10.4, which show promising results concerning the reconstructed camera trajectory and the ability to close loops.

SCME is tested with image data from different cameras. It obtaines camera poses from omnidirectional (fishey) and from full omnidirectional images and yield better results than *Metashape* does. It successfully obtaines camera poses from RGBD data and achieves similar results to *ORBSlam2*. In case of directional RGB images SCME's estimated camera trajectory shows significant differences, but still provides a closed model. SCME uses the same up-to-scale two-view transformation for RGB and RGBD data, this leads to the conclusion, that global translation scale factor determination provides slightly different relations, which might be caused by noise. Thus erroneous ratio factors should be removed from optimization.

11 Conclusion

SCME is a novel non-incremental gobal approach to obtain camera poses from up-to-scale two-view transformations for spherical cameras. Compared to other approaches SCME's processing steps are kept simple. SCME builds up on existing processing tasks (feature matching and geometric verification) from SfM and incorporates available pose graph solvers. Thus the integration of the proposed workflow can be done straightforwardly.

11.1 Summary

SCME represent an entire pipeline with various novelties and findings, which are summarized in the follwoing.

P2S-maps are an equation-free representation of the projection geometry. SCME uses P2S-maps to incorporate any central directional and central omnidirectional projection from cameras and world maps. P2S-maps are color-coded lookups, which store mappings from pixels to corresponding positions onto unit sphere. It is a central element of this proposed work and paves the way for a general SCM (Spherical Camera Model). P2S-maps also enable to interpolate mappings for intermediate pixels and they can be used to convert image data between projections. A routine is presented to convert images between projections based on their P2S-maps. As a consequence, this rountine does not need any prior knowledge of the underlying projections.

Intrinsic camera calibration incorporates UCM (Unified Camera Model) as well as PCM (Polynomal Camera Model) and generates P2S-maps from intrinsic camera parameters. It utilizes different target detection and calibration algorithms to improve the calibration process. The intrisic calibration routine is tested with different cameras.

Extrinsic camera calibration utilizes P2S-maps to incorporate a wide range of central directional and omnidirectional camera types. It uses PGO as back-end to solve for a non-restricted number of camera and target poses from relative target-camera transformations. PnP based on spherical coordinates is used to obtain relative target-camera transformations from 3d-2d and 2d-2d point correspondences. Finally, BA uses the results from PGO for further refinement. The extrinsic calibration is tested at hand of an RGBD camera, a four fisheye camera setup, a stereo setup consisting of persepective cameras and an eight camera setup also consisting of perspective cameras.

World map projections are used as target projections for undistortion purposes. A target world map projection and its corresponding P2S-map are obtained from a geograhic library. Camera images are then converted to the target projection based on their corresponding P2S-maps. This represents a felxible and general method for image undistortion.

Two camera relations are extensively described in this work. The *alternative* midpoint method is presented, which directly obtaines depth values from point correspondences. Based on these findings an *improved* procedure is developed, which straightforwardly resolves ambiguity from essential matrix deocomposition within two calculation steps only. Different two-view estimation algorithms are compared, where the *classic* 8-point algorithms achieves best results. Two-view optimization is based on a derived circumferential distance in combination with a simple epipolar distance. Furthermore, the identificaiton of degeneracies is explained based on homography for spherical cameras. Additionally the calculation of two-view translation scale factors from depth data is described.

Three camera relations are comprehensively described. During this topic, the three-view geometry is derived, a new *crossing epipolar planes* geometry is developed and the trifocal geometry is reformulated. The relations between these three geometries are shown, which are also able to obtain translation ratio factors between up-to-scale two-view transformations. This is another central element of this work, which allows to develop a global and structureless pose estimation algorithm. As an evaluation result the trifocal geometry achieves the best translation ratio factor estimates.

Global translation scaling uses translation ratio factors to obtain translation scale factors that correspond to up-to-scale two-view transformations. This is a novel approach which is non-incremental and doesn't require any triangulation. Instead it solves for all translation scale factors at once. As a consequence translation scale estimation becomes independent of global rotation estimation.

PGO is used as back-end in many Visual SLAM implementations to efficiently solve for camera poses from scaled two-view transformations, thus solvers require a sequence of scaled transformations. This work presents a strategy to extract a sequence of transformations from an unordered set of scaled two-view transformations. PGO is then used to solve for camera poses.

SCME results from different camera types show promising perspectives. SCME is able to obtain camera poses from RGBD data sets comparable to *ORBSlam2*. It also obtaines pose estimates from provided omnidirectional as well as from directional RGB images. As mentioned several times in this work, SCME is an estimation approach that tries to find ininital poses for further optimization. It is not aimed at replacing

BA or structureless BA. Taking this into account shows that SCME provides reliable pose estimaes.

11.2 Outlook and Future Work

As always with new algorithms, more tests need to be done under varying conditions to better evaluate results and to see strengths and weaknesses of this approach. Furthermore, this allows to improve certain routines and to obtain thresholds to set upper and lower bounds in the optimization regime.

As stated in Chapter 9 SCME obtaines camera poses from scaled two-view transformations using PGO, which does not take any measured features into account. Scaled two-view transformations are an intermediate result from two-view optimization and translation scaling. Thus the entire estimation represents an intermediate calculation step. Further improvements can be achieved by directly using structureless BA or by using BA if feature points are triangulated. Unfortunately up-to-now there is no ready-to-use implementation neither for BA nor structureless BA based on SCM. An implementation using the *Ceres* library might be a possible solution.

The determination of translation ratio factors und translation scaling factors could be implemented into existing *incremental* SfM pipelines to be used in conjunction with calibrated cameras. In doing so this would integrate a *global* SfM approach into existing *incremental* SfM implementations. As already explained *incremental* SfM obtaines up-to-scale two-view transformations to geometrically verify point correspondences. These transformtions can be simply used to determine translation ratio factors and consequently translation scale factors. Using PGO provides camera pose estimates and thus would reduce computational cost and the risk to fail due to SfM's incremental reconstruction approach.

SCME might be also integrated into monocular VO to scale two-view transformations between sequential frames. Translation scale factor determination is based on translation ratio factors that can be obtained during data association between consecutive frames. Hence translation scaling is a sparse linear problem that keeps manageable even in large-scale scenarios and would allow for real-time translation scaling.

Appendices

A.1 Additional Extrinsic Calibration Results

This section presents additional extrinsic camera results, which are obtained using the proposed pipeline from Section 4.4, page 58.

Stereo Camera Setup

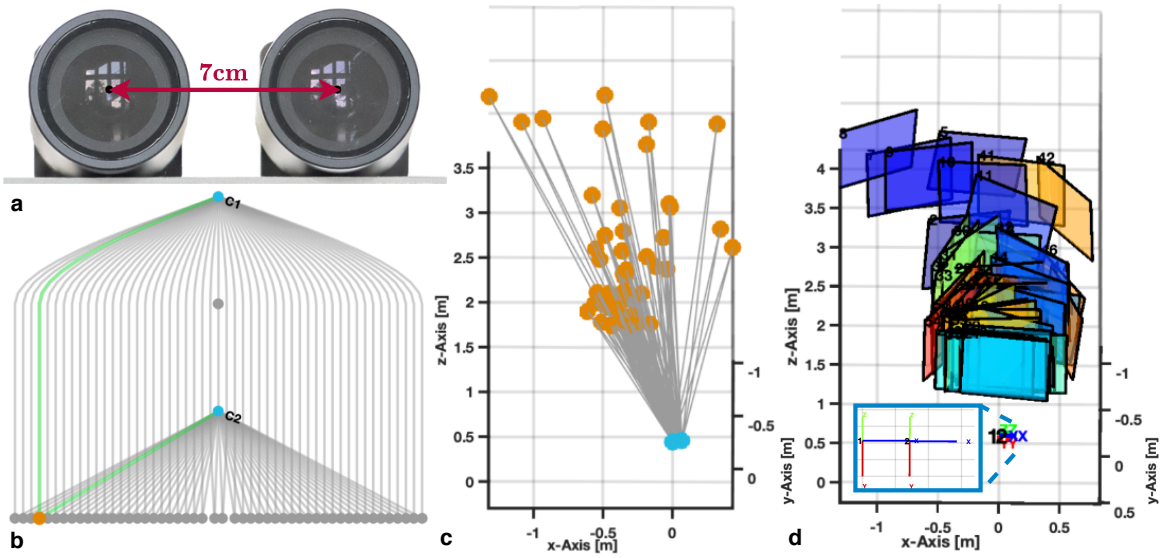

Fig. A.1.1: Extrinsic calibration of a camera setup consisting of two *Baumer* cameras. The following is shown: **a)** stereo camera setup with a $7cm$ baseline [258], **b)** a graph view visualizes the connections between cameras and targets (▬) and highlights the sub-pose graph (▬), which connects the two camera nodes (●, C1 and C2) using one target node (●), **c)** optimized pose graph with all target nodes, **d)** camera and target alignment from BA-refinement.

This classic stereo setup consists of two *Baumer VLG-12C.I* cameras, each equipped with a *Ricoh FL-HC0416X-VG 4.2mm f/1.6* lens. This setup was used on a mobile robotic platform in order to generate virtual 3d models from underground mines [82]. Fig. A.1.1**a** illustrates the stereo setup consisting of a $7cm$ baseline (along X-axis) and **b-d** illustrate the extrinsic calibration results. **Bouguet's Camera Calibration Toolbox for Matlab** [19] (explained in Section 4.3, page 54) is used to provide a reference calibration. The derived extrinsic results for the 2$^{\text{nd}}$ camera from the proposed extrinsic calibration workflow and the reference calibration (simply named Bouguet) are listed in Tab. A.1.1. As can be seen, both calibration routines obtain similar rotation parameters. Bouguet's calibration routine derives a smaller baseline ($t_x = 6.835cm$), whereas the proposed routine obtains a larger baseline ($t_x = 7.166cm$). Interestingly Bouguet's routine additionally derives an offset $t_z = 0.727cm$, which is a magnitude larger compared to the proposed routines with $t_z = 0.075cm$. Usually, the offset along Z-axis should be negligibly small. As a consequence, the proposed calibration routine obtains better translation estimates in viewing direction (Z-axis).

Tab. A.1.1: Comparison of extrinsic calibration results for the Baumer stereo setup. Extrinsics are listed for the 2nd camera with respect to the 1st one. Rotation is represented in Euler angles (roll, pitch, yaw) for better comparison.

Extrinsics	**Bouguet**	**Proposed**
γ	$-0.002° \pm 0.009°$	$-0.002° \pm 0.022°$
β	$-0.008° \pm 0.009°$	$-0.008° \pm 0.028°$
α	$-0.005° \pm 0.009°$	$-0.005° \pm 0.019°$
t_x	$6.835cm \pm 0.246cm$	$7.166cm \pm 0.023cm$
t_y	$0.085cm \pm 0.233cm$	$0.023cm \pm 0.020cm$
t_z	$0.727cm \pm 0.251cm$	$0.075cm \pm 0.055cm$

RGBD Camera Setup

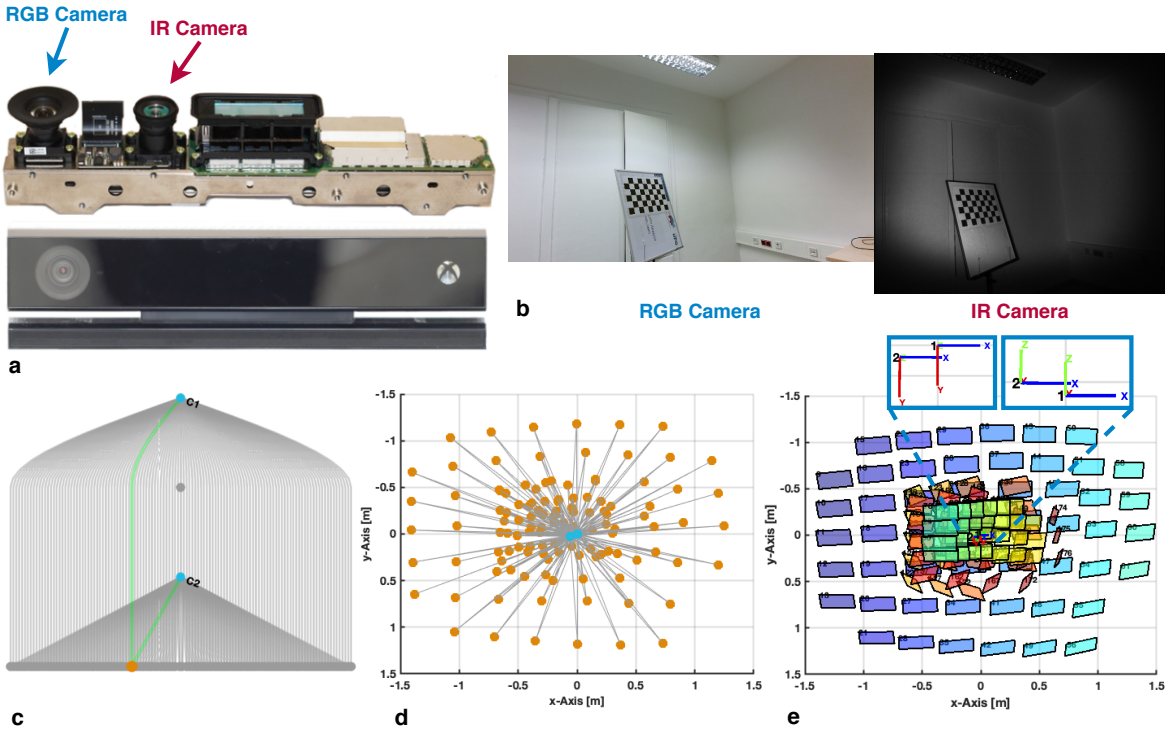

Fig. A.1.2: Extrinsic calibration of a *Kinect v2* camera consisting of a color camera and an infrared one. The following is shown: **a)** setup with indicated cameras [258], **b)** sample calibration images from both cameras, **c)** a graph view visualizes the connections between cameras and targets (▬) and highlights the sub-pose graph (▬), which connects the two camera nodes (●, C1 and C2) using one target node (●), **d)** optimized pose graph with all target nodes, **e)** camera and target alignment from BA-refinement.

The *Kinect v2* is an RGBD-camera consisting of a color camera and an infrared one as illustrated in Fig. A.1.2**a**. It also represents a stereo camera system but in contrast to the *Baumer* setup it consists of two different different sensors. The reference calibration parameters are taken from [123], which are provided without uncertainties. The derived extrinsic results for the 2$^{\text{nd}}$ camera from the proposed extrinsic calibration workflow and the reference calibration (simply named Reference) are listed in Tab. A.1.2.

Both calibrations yield similar results with negligible small rotation differences. The most noticable discrepancy is given for the translation t_z in viewing direction with $\approx 0.7cm$. However these results do not allow to draw direct conclusions concerining the question which one performs better, since each calibration uses its own image data set. Finally it can be said, the proposed calibration workflow delivers satisfying results for the tested RGBD-camera.

Tab. A.1.2: Comparison of extrinsic calibration results for the *Kinect v2* camera. Extrinsics are listed for the 2nd camera (infrared) with respect to the 1st one (color). Rotation is represented in Euler angles (roll, pitch, yaw) for better comparison.

Extrinsics	**Reference**	**Proposed**
γ	$-0.379°$	$-0.437° \pm 0.048°$
β	$-0.060°$	$-0.716° \pm 0.053°$
α	$-0.410°$	$-0.096° \pm 0.044°$
t_x	$-6.015cm$	$-6.135cm \pm 0.018cm$
t_y	$0.221cm$	$0.334cm \pm 0.013cm$
t_z	$2.714cm$	$1.972cm \pm 0.029cm$

A.2 Linear Least Squares Scaling

Assuming a linear scaling between two vectors $\mathbf{a}$ and $\mathbf{b}$ with a scaling factor λ such that

$$\begin{pmatrix} \mathbf{a}_{(1)} \\ \vdots \\ \mathbf{a}_{(m)} \end{pmatrix} \lambda = \begin{pmatrix} \mathbf{b}_{(1)} \\ \vdots \\ \mathbf{b}_{(m)} \end{pmatrix} .$$

The optimization in least scares sense is defined as

$$\min \sum_{i=1}^{m} \left\| \mathbf{a}_{(i)} \lambda - \mathbf{b}_{(i)} \right\|^2 ,$$

which turns into the following equation:

$$\left(\mathbf{a}_{(1)} \lambda - \mathbf{b}_{(1)} \right)^2 + \cdots + \left(\mathbf{a}_{(m)} \lambda - \mathbf{b}_{(m)} \right)^2 = 0$$

$$\lambda^2 \left(\mathbf{a}_{(1)}^2 + \cdots + \mathbf{a}_{(m)}^2 \right) - 2\lambda \left(\mathbf{a}_{(1)} \mathbf{b}_{(1)} + \cdots + \mathbf{a}_{(m)} \mathbf{b}_{(m)} \right) + \mathbf{b}_{(1)}^2 + \cdots + \mathbf{b}_{(m)}^2 = 0 .$$

Obtaining the derivative function ∂ with respect to λ and setting it to zero leads to a least squares approximation of λ:

$$\frac{\partial}{\partial \lambda} = 0 = 2\lambda \left(\mathbf{a}_{(1)}^2 + \cdots + \mathbf{a}_{(m)}^2 \right) - 2 \left(\mathbf{a}_{(1)} \mathbf{b}_{(1)} + \cdots + \mathbf{a}_{(m)} \mathbf{b}_{(m)} \right)$$

$$\lambda = \frac{\mathbf{a}_{(1)} \mathbf{b}_{(1)} + \cdots + \mathbf{a}_{(m)} \mathbf{b}_{(m)}}{\mathbf{a}_{(1)}^2 + \cdots + \mathbf{a}_{(m)}^2}$$

$$\boxed{\lambda = \frac{\mathbf{a} \bullet \mathbf{b}}{\|\mathbf{a}\|^2} = \frac{\mathbf{a}^T \mathbf{b}}{\|\mathbf{a}\|^2}} \tag{A.1}$$

Speaking in a geometric context this presents the orthogonal projection of $\mathbf{b}$ onto $\mathbf{a}$.

A.3 Proof Rank Deficiency

This section proofs B's rank deficiency described in Section 6.2.1, page 89 by showing that the third row is linearly dependent. Let's revisit:

$$B = \left[\ [\mathring{\mathbf{X}}_i]_\times \ \middle|\ \mathbf{t}_i \times \mathring{\mathbf{X}}_i\ \right] \in \mathbb{R}^{3\times 4}$$

$$B = \begin{bmatrix} 0 & -\mathring{\mathbf{X}}_{i(3)} & \mathring{\mathbf{X}}_{i(2)} & \mathbf{t}_{i(2)}\mathring{\mathbf{X}}_{i(3)} - \mathbf{t}_{i(3)}\mathring{\mathbf{X}}_{i(2)} \\ \mathring{\mathbf{X}}_{i(3)} & 0 & -\mathring{\mathbf{X}}_{i(1)} & \mathbf{t}_{i(3)}\mathring{\mathbf{X}}_{i(1)} - \mathbf{t}_{i(1)}\mathring{\mathbf{X}}_{i(3)} \\ -\mathring{\mathbf{X}}_{i(2)} & \mathring{\mathbf{X}}_{i(1)} & 0 & \mathbf{t}_{i(1)}\mathring{\mathbf{X}}_{i(2)} - \mathbf{t}_{i(2)}\mathring{\mathbf{X}}_{i(1)} \end{bmatrix}.$$

For the sake of convenience let

$$\mathring{\mathbf{t}}_i = \mathbf{t}_i \times \mathring{\mathbf{X}}_i \tag{A.2}$$

such that

$$B = \begin{bmatrix} 0 & -\mathring{\mathbf{X}}_{i(3)} & \mathring{\mathbf{X}}_{i(2)} & \mathring{\mathbf{t}}_{i(1)} \\ \mathring{\mathbf{X}}_{i(3)} & 0 & -\mathring{\mathbf{X}}_{i(1)} & \mathring{\mathbf{t}}_{i(2)} \\ -\mathring{\mathbf{X}}_{i(2)} & \mathring{\mathbf{X}}_{i(1)} & 0 & \mathring{\mathbf{t}}_{i(3)} \end{bmatrix}.$$

Rewriting the third row of B as linear combination of the previous ones:

$$\lambda_1 B^T_{(1,:)} + \lambda_2 B^T_{(2,:)} = B^T_{(3,:)}$$

$$\lambda_1 \begin{pmatrix} 0 \\ -\mathring{\mathbf{X}}_{i(3)} \\ \mathring{\mathbf{X}}_{i(2)} \\ \mathring{\mathbf{t}}_{i(1)} \end{pmatrix} + \lambda_2 \begin{pmatrix} \mathring{\mathbf{X}}_{i(3)} \\ 0 \\ -\mathring{\mathbf{X}}_{i(1)} \\ \mathring{\mathbf{t}}_{i(2)} \end{pmatrix} = \begin{pmatrix} -\mathring{\mathbf{X}}_{i(2)} \\ \mathring{\mathbf{X}}_{i(1)} \\ 0 \\ \mathring{\mathbf{t}}_{i(3)} \end{pmatrix}$$

$$0 + \lambda_2 \mathring{\mathbf{X}}_{i(3)} = -\mathring{\mathbf{X}}_{i(2)} \tag{A.3}$$

$$-\lambda_1 \mathring{\mathbf{X}}_{i(3)} + 0 = \mathring{\mathbf{X}}_{i(1)} \tag{A.4}$$

$$\lambda_1 \mathring{\mathbf{X}}_{i(2)} - \lambda_2 \mathring{\mathbf{X}}_{i(1)} = 0 \tag{A.5}$$

$$\lambda_1 \mathring{\mathbf{t}}_{i(1)} + \lambda_2 \mathring{\mathbf{t}}_{i(2)} = \mathring{\mathbf{t}}_{i(3)}\,. \tag{A.6}$$

Obviously, solving Eq. (A.4) for λ_1 yields

$$\lambda_1 = -\frac{\mathring{\mathbf{X}}_{i(1)}}{\mathring{\mathbf{X}}_{i(3)}} \tag{A.7}$$

and similarily solving Eq. (A.3) for λ_2 gives

$$\lambda_2 = -\frac{\mathring{\mathbf{X}}_{i(2)}}{\mathring{\mathbf{X}}_{i(3)}}\,. \tag{A.8}$$

Applying Eq. (A.7) and Eq. (A.8) to Eq. (A.5) shows that:

$$-\frac{\mathring{\mathbf{X}}_{i(1)}}{\mathring{\mathbf{X}}_{i(3)}}\mathring{\mathbf{X}}_{i(2)} + \frac{\mathring{\mathbf{X}}_{i(2)}}{\mathring{\mathbf{X}}_{i(3)}}\mathring{\mathbf{X}}_{i(1)} = 0$$
$$0 = 0\,.$$

Applying Eq. (A.7) and Eq. (A.8) to Eq. (A.6) yields

$$-\frac{\mathring{\mathbf{X}}_{i(1)}}{\mathring{\mathbf{X}}_{i(3)}}\mathring{\mathbf{t}}_{i(1)} - \frac{\mathring{\mathbf{X}}_{i(2)}}{\mathring{\mathbf{X}}_{i(3)}}\mathring{\mathbf{t}}_{i(2)} = \mathring{\mathbf{t}}_{i(3)}\,.$$

Going on by substracting $\mathring{\mathbf{t}}_{i(3)}$ from the right-hand side and multiplying $-\mathring{\mathbf{X}}_{i(3)}$ results to

$$\mathring{\mathbf{X}}_{i(1)}\mathring{\mathbf{t}}_{i(1)} + \mathring{\mathbf{X}}_{i(2)}\mathring{\mathbf{t}}_{i(2)} + \mathring{\mathbf{X}}_{i(3)}\mathring{\mathbf{t}}_{i(3)} = 0\,.$$

Transforming the previous equation into vector mutliplication form leads to

$$\mathring{\mathbf{X}}_i^T\mathring{\mathbf{t}}_i = \mathring{\mathbf{X}}_i \bullet \mathring{\mathbf{t}}_i = 0\,.$$

Finally, re-entering Eq. (A.2) in order to replace $\mathring{\mathbf{t}}_i$ confirms that:

$$\underbrace{\mathring{\mathbf{X}}_i \bullet \left(\mathbf{t}_i \times \mathring{\mathbf{X}}_i\right)}_{\mathbf{a}\bullet(\mathbf{b}\times\mathbf{a})=0} = 0$$
$$0 = 0\,.$$

A.4 Alternative Derivation Midpoint Method

This alternative approach derives Eqs. (6.6) and (6.7), page 92. It combines the back-projection function Eq. (6.2) on page 88 and the parallelism constraint given by Eq. (6.4) on page 89.
Assuming Eq. (6.2) to be camera k

$$\mathbf{U} = R_k \hat{\mathbf{X}}_k \lambda_k + \mathbf{t}_k \tag{A.9}$$

and Eq. (6.4) to be camera l

$$R_l \hat{\mathbf{X}}_l \times \big(\mathbf{U} - \mathbf{t}_l\big) = \mathbf{0}\,. \tag{A.10}$$

Applying Eq. (A.9) to Eq. (A.10) results to the following equation:

$$\begin{aligned}
R_l \hat{\mathbf{X}}_l \times \big(R_k \hat{\mathbf{X}}_k \lambda_k + \mathbf{t}_k - \mathbf{t}_l\big) &= \mathbf{0} \\
R_l \hat{\mathbf{X}}_l \times \big(\mathbf{t}_k - \mathbf{t}_l\big) + \lambda_k \big(\underbrace{R_l \hat{\mathbf{X}}_l \times R_k \hat{\mathbf{X}}_k}_{\mathbf{a}\times\mathbf{b}=-(\mathbf{b}\times\mathbf{a})}\big) &= \mathbf{0} \\
R_l \hat{\mathbf{X}}_l \times \big(\mathbf{t}_k - \mathbf{t}_l\big) - \lambda_k \big(R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l\big) &= \mathbf{0}\,,
\end{aligned}$$

which is the same as Eq. (6.6) on page 92. Switching k and l produces the same result as described in Eq. (6.7) on page 92, namely:

$$\begin{aligned}
R_k \hat{\mathbf{X}}_k \times \big(R_l \hat{\mathbf{X}}_l \lambda_l + \mathbf{t}_l - \mathbf{t}_k\big) &= \mathbf{0} \\
\underbrace{R_k \hat{\mathbf{X}}_k \times \big(\mathbf{t}_l - \mathbf{t}_k\big)}_{-(\mathbf{a}\times\mathbf{b})=\mathbf{a}\times-\mathbf{b}} + \lambda_l \big(R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l\big) &= \mathbf{0} \\
-\big(R_k \hat{\mathbf{X}}_k \times \big(\mathbf{t}_k - \mathbf{t}_l\big)\big) + \lambda_l \big(R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l\big) &= \mathbf{0} \\
R_k \hat{\mathbf{X}}_k \times \big(\mathbf{t}_k - \mathbf{t}_l\big) - \lambda_l \big(R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l\big) &= \mathbf{0}\,.
\end{aligned}$$

A.5 Simplification of Depth Calculation

This section contains the simplification of the depth calculation mentioned in Section 6.2.2, page 91 by considering the collinearity between $R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l$ and $R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)$ as well as between $R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l$ and $R_l\hat{\mathbf{X}}_l \times \left(\mathbf{t}_k - \mathbf{t}_l\right)$. Taking these constraints into account reduces Eq. (6.8) on page 93 to:

$$
\begin{aligned}
\lambda_k &= \frac{\overbrace{\underbrace{\left(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right)}_{\mathbf{a}} \cdot \underbrace{\left(R_l\hat{\mathbf{X}}_l \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right)}_{\mathbf{b}}}^{\mathbf{a}\cdot\mathbf{b}=\|\mathbf{a}\|\,\|\mathbf{b}\|}}{\underbrace{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\|^2}_{\|\mathbf{a}\|^2}} \\
\lambda_k &= \frac{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\| \left\|R_l\hat{\mathbf{X}}_l \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right\|}{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\|^2} \\
\lambda_k &= \frac{\left\|R_l\hat{\mathbf{X}}_l \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right\|}{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\|},
\end{aligned}
$$

and similarily Eq. (6.9) on page 93 to:

$$
\begin{aligned}
\lambda_l &= \frac{\overbrace{\underbrace{\left(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right)}_{\mathbf{a}} \cdot \underbrace{\left(R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right)}_{\mathbf{b}}}^{\mathbf{a}\cdot\mathbf{b}=\|\mathbf{a}\|\,\|\mathbf{b}\|}}{\underbrace{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\|^2}_{\|\mathbf{a}\|^2}} \\
\lambda_l &= \frac{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\| \left\|R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right\|}{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\|^2} \\
\lambda_l &= \frac{\left\|R_k\hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l\right)\right\|}{\left\|R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\right\|}.
\end{aligned}
$$

A.6 Relation between Epipolar and Circumferential Constraint

This section describes the relation between circumferential constraint and epipolar constraint. In Section 6.6.2, page 109 the circumferential distance is introduced, which is based on on Eq. (6.5), page 92

$$0 = \mathbf{t} + \lambda_k R\hat{\mathbf{X}}_k - \lambda_l \hat{\mathbf{X}}_l \,.$$

Applying the simplified depth factor calculation from Eqs. (6.10) and (6.11), page 93 with:

$$\lambda_k^i = \frac{\|\hat{\mathbf{X}}_l \times \mathbf{t}\|}{\|R\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l\|} \qquad \lambda_l = \frac{\|R\hat{\mathbf{X}}_k \times \mathbf{t}\|}{\|R\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l\|}$$

and multiplying $\|R\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l\|$ from the right yields the proposed circumferential constraint

$$\mathbf{0} = \|R\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l\|\mathbf{t} + \|\hat{\mathbf{X}}_l \times \mathbf{t}\|R\hat{\mathbf{X}}_k - \|R\hat{\mathbf{X}}_k \times \mathbf{t}\|\hat{\mathbf{X}}_l \,.$$

Scalar multiplying $\left(\mathbf{t} \times R\hat{\mathbf{X}}_k\right)$ from the right yields the epipolar geometry from Section 6.3, page 94:

$$\begin{aligned}
\mathbf{0} &= \|R\hat{\mathbf{X}}_k \times \hat{\mathbf{X}}_l\| \underbrace{\mathbf{t} \bullet \left(\mathbf{t} \times R\hat{\mathbf{X}}_k\right)}_{\mathbf{a}\bullet(\mathbf{a}\times\mathbf{b})=0} + \|\hat{\mathbf{X}}_l \times \mathbf{t}\| \underbrace{R\hat{\mathbf{X}}_k \bullet \left(\mathbf{t} \times R\hat{\mathbf{X}}_k\right)}_{\mathbf{a}\bullet(\mathbf{b}\times\mathbf{a})=0} \\
&\quad - \|R\hat{\mathbf{X}}_k \times \mathbf{t}\|\hat{\mathbf{X}}_l \bullet \left(\mathbf{t} \times R\hat{\mathbf{X}}_k\right) \\
\mathbf{0} &= -\|R\hat{\mathbf{X}}_k \times \mathbf{t}\|\hat{\mathbf{X}}_l \bullet \left(\mathbf{t} \times R\hat{\mathbf{X}}_k\right) \\
\mathbf{0} &= \hat{\mathbf{X}}_l \bullet \left(\mathbf{t} \times R\hat{\mathbf{X}}_k\right) \\
\mathbf{0} &= \hat{\mathbf{X}}_l^T \left(\mathbf{t}_{[\times]} R\hat{\mathbf{X}}_k\right) .
\end{aligned}$$

As shown the epipolar constraint can be also developed from the circumferential constraint.

A.7 Covariance Estimation

This section describes the estimation of the covariance matrix S from corresponding optimized parameters as it is used in this work. Assuming a set of measurements x^i and y^i, $i = 1, \ldots, p$ and a mapping function $f(\cdot)$ with the parameters α and β such that $\epsilon^i = \mathbf{d}\big(y^i, f(x^i, \alpha, \beta)\big)$. Here, $\mathbf{d}(\cdot, \cdot)$ denotes a distance function with ϵ^i denoting the i^{th} residual. Both parameters α and β are optimized by minimizing the sum of squared distances

$$\underset{\alpha,\beta}{\operatorname{argmin}} \sum_{i=1}^{p} \mathbf{d}\big(y^i, f(x^i, \alpha, \beta)\big)^2 .$$

The uncertainties of α and β are embedded in the covariance matrix

$$S = \begin{bmatrix} \sigma_\alpha^2 & \sigma_{\alpha\beta}^2 \\ \sigma_{\alpha\beta}^2 & \sigma_\beta^2 \end{bmatrix} , \tag{A.11}$$

which is determined by

$$S = H^{-1}\sigma^2$$

with H denoting the Hessian matrix of the distance function $\mathbf{d}(\cdot, \cdot)$. $H \approx J^T J$ is an approximation, if either the second derivatives of the residuals ϵ^i are small or the residuals ϵ^i themselves are small such that

$$S = \big[J^T J\big]^{-1}\sigma^2 . \tag{A.12}$$

J denotes the Jacobian of the residuals ϵ^i with respect to α and β

$$J = \begin{bmatrix} \frac{\partial \epsilon^1}{\partial \alpha} & \frac{\partial \epsilon^1}{\partial \beta} \\ \vdots & \vdots \\ \frac{\partial \epsilon^p}{\partial \alpha} & \frac{\partial \epsilon^p}{\partial \beta} \end{bmatrix}$$

and

$$\sigma^2 = \frac{1}{p-1} \sum_{i=1}^{p} \big(\epsilon^i\big)^2 \tag{A.13}$$

denotes the sample variance of the residuals.

A.8 Uncertainty Estimation from Epipolar Geometry

The following describes the covariance estimation for the epipolar constraint from Eq. (6.19), page 108

$$\epsilon^i = \left(\hat{\mathbf{X}}_l^i\right)^T \left(\mathbf{t} \times R\hat{\mathbf{X}}_k^i\right),$$

which is used to optimize R and $\mathbf{t}$ from a set of point correspondences $\hat{\mathbf{X}}_k^i \leftrightarrow \hat{\mathbf{X}}_l^i$, $i = 1, \ldots, p$ by minimizing the sum of squared epipolar distances

$$\underset{R,\mathbf{t}}{\operatorname{argmin}} \sum_{i=1}^{m} \left(\epsilon^i\right)^2.$$

The corresponding Jacobian J is defined

$$J = \begin{bmatrix} \frac{\partial \epsilon^1}{\partial R} & \frac{\partial \epsilon^1}{\partial \mathbf{t}} \\ \vdots & \vdots \\ \frac{\partial \epsilon^p}{\partial R} & \frac{\partial \epsilon^p}{\partial \mathbf{t}} \end{bmatrix}$$

with

$$\frac{\partial \epsilon^i}{\partial R} = \left(\hat{\mathbf{X}}_k^i \otimes \left(\hat{\mathbf{X}}_l^i \times \mathbf{t}\right)\right)^T$$
$$\frac{\partial \epsilon^i}{\partial \mathbf{t}} = \left(R\hat{\mathbf{X}}_k^i \times \hat{\mathbf{X}}_l^i\right)^T$$

Referring to Appendix A.7, page 180 the covariance S can be approximated such that

$$\boxed{S = \left[J^T J\right]^{-1} \frac{1}{p-1} \sum_{i=1}^{p} \left(\epsilon^i\right)^2}. \tag{A.14}$$

$\left[J^T J\right]^{-1}$ becomes rank-deficient if the three dimensional rotation is overparameterized as it is the case for a rotation matrix (nine parameters) or a quaternion (four parameters) representation. In order to prevent this circumstance it is recommended to use a rotation representation such as Euler angles or Rodrigues vector. Later the covariance matrix can be transferred into rotation matrix or quaternion representation as proposed in [17].

A.9 Two-View Scaling Factor Estimation: Uncertainty Estimation

This section explains the uncertainty estimation for the linear two-view translation scaling from Section 6.7.1, page 114 and is based on the covariance approximation from Appendix A.7, page 180. Error propagation and uncertainty estimation are developed as exemplary described in [54]. The cost function represents the simple distances $\boldsymbol{\delta} - \tilde{\boldsymbol{\lambda}} s$ between given depth information

$$\boldsymbol{\delta} = \left(\delta_k^1, \ldots, \delta_k^m, \delta_l^1, \ldots, \delta_l^n\right)^T \qquad \in \mathbb{R}^{(m+n)\times 1}$$

and recalculated ones

$$\tilde{\boldsymbol{\lambda}} = \left(\tilde{\lambda}_k'^1, \ldots, \tilde{\lambda}_k'^m, \tilde{\lambda}_l''^1, \ldots, \tilde{\lambda}_l''^n\right)^T \qquad \in \mathbb{R}^{(m+n)\times 1}$$

with:

$$\tilde{\lambda}_k'^i = \frac{\left(\hat{\mathbf{X}}_l'^i \times \hat{\mathbf{t}}\right)^T \left(R\hat{\mathbf{X}}_k'^i \times \hat{\mathbf{X}}_l'^i\right)}{\left\| R\hat{\mathbf{X}}_k'^i \times \hat{\mathbf{X}}_l'^i \right\|^2}, \qquad i = 1, \ldots, m$$

$$\tilde{\lambda}_l''^j = \frac{\left(R\hat{\mathbf{X}}_k''^j \times \hat{\mathbf{t}}\right)^T \left(R\hat{\mathbf{X}}_k''^j \times \hat{\mathbf{X}}_l''^j\right)}{\left\| R\hat{\mathbf{X}}_k''^j \times \hat{\mathbf{X}}_l''^j \right\|^2}, \quad j = 1, \ldots, n$$

from Eqs. (6.6) and (6.7), page 92. Describing the error function as linear least squares formulation yields

$$s_{\text{init}} = \min_s \|\boldsymbol{\delta} - \tilde{\boldsymbol{\lambda}} s\|^2 .$$

The solution for the scaling factor's variance is given by

$$\sigma_{s_{\text{init}}}^2 = (J^T J)^{-1} \sigma_{\boldsymbol{\epsilon}}^2$$

with J denoting the Jacobian $J \triangleq -\tilde{\boldsymbol{\lambda}}$. Since $\tilde{\boldsymbol{\lambda}}^T \tilde{\boldsymbol{\lambda}} = \|\tilde{\boldsymbol{\lambda}}\|^2$ yields a scalar value, it simplifies the solution

$$\sigma_{s_{\text{init}}}^2 = \frac{1}{\|\tilde{\boldsymbol{\lambda}}\|^2} \sigma_{\boldsymbol{\epsilon}}^2 .$$

$\sigma_{\boldsymbol{\epsilon}}^2$ denotes the sample variance

$$\sigma_{\boldsymbol{\epsilon}}^2 = \frac{1}{m+n-1} \|\boldsymbol{\delta} - \tilde{\boldsymbol{\lambda}} s\|^2 ,$$

which is an approximator for the unknown noise variance. The scaling factor's standard deviation is finally obtained by

$$\boxed{\sigma_{s_{\text{init}}} = \frac{\|\boldsymbol{\delta} - \tilde{\boldsymbol{\lambda}} s_{\text{init}}\|}{\sqrt{m+n-1}\|\tilde{\boldsymbol{\lambda}}\|}}. \tag{A.15}$$

A.10 Two-View Scaling Factor Optimization: Uncertainty Estimation

This section explains the uncertainty estimation for the non-linear two-view translation scaling from Section 6.7.2, page 116 and is based on the covariance approximation from Appendix A.7, page 180. Error propagation and uncertainty estimation are similar to Appendix A.9, however the Jacobians are more complex with:

$$\frac{\partial \hat{\mathbf{Y}}_k^i}{\partial \mathbf{Y}_k^i} = \frac{I - \hat{\mathbf{Y}}_k^i \left(\hat{\mathbf{Y}}_k^i\right)^T}{\|\mathbf{Y}_k^i\|} \qquad \frac{\partial \mathbf{Y}_k^i}{\partial s} = -R^T \hat{\mathbf{t}}$$
$$\frac{\partial \hat{\mathbf{Y}}_l^j}{\partial \mathbf{Y}_l^j} = \frac{I - \hat{\mathbf{Y}}_l^j \left(\hat{\mathbf{Y}}_l^j\right)^T}{\|\mathbf{Y}_l^j\|} \qquad \frac{\partial \mathbf{Y}_l^j}{\partial s} = \hat{\mathbf{t}}\,.$$

$$J_k^i = \frac{\partial \hat{\mathbf{Y}}_k^i}{\partial \mathbf{Y}_k^i}\frac{\partial \mathbf{Y}_k^i}{\partial s} = -\frac{I - \frac{\delta_l^i R^T \hat{\mathbf{X}}_l'^i - sR^T\hat{\mathbf{t}}}{\|\delta_l^i R^T \hat{\mathbf{X}}_l'^i - sR^T\hat{\mathbf{t}}\|}\left(\frac{\delta_l^i R^T \hat{\mathbf{X}}_l'^i - sR^T\hat{\mathbf{t}}}{\|\delta_l^i R^T \hat{\mathbf{X}}_l'^i - sR^T\hat{\mathbf{t}}\|}\right)^T}{\|\delta_l^i R^T \hat{\mathbf{X}}_l'^i - sR^T\hat{\mathbf{t}}\|} R^T\hat{\mathbf{t}}$$
$$J_l^j = \frac{\partial \hat{\mathbf{Y}}_l^j}{\partial \mathbf{Y}_l^j}\frac{\partial \mathbf{Y}_l^j}{\partial s} = \frac{I - \frac{\delta_k^j R \hat{\mathbf{X}}_k''^j + s\hat{\mathbf{t}}}{\|\delta_k^j R \hat{\mathbf{X}}_k''^j + s\hat{\mathbf{t}}\|}\left(\frac{\delta_k^j R \hat{\mathbf{X}}_k''^j + s\hat{\mathbf{t}}}{\|\delta_k^j R \hat{\mathbf{X}}_k''^j + s\hat{\mathbf{t}}\|}\right)^T}{\|\delta_k^j R \hat{\mathbf{X}}_k''^j + s\hat{\mathbf{t}}\|}\hat{\mathbf{t}}$$

The Jacobians J_i^k and J_l^j are further stacked to form $\mathbf{J}$ with

$$\mathbf{J} = \left(J_k^1, \ldots, J_k^m, J_l^1, \ldots, J_l^n\right)^T \in \mathbb{R}^{3(m+n)\times 1},$$

that is used to calculate the scaling factor's variance

$$\sigma_{s_{\text{opt}}}^2 = \left(\mathbf{J}^T\mathbf{J}\right)^{-1}\frac{1}{m+n-1}\|\hat{\mathbf{X}} - \hat{\mathbf{Y}}\|^2\,.$$

Since $\boldsymbol{J}^T\boldsymbol{J} = \|\boldsymbol{J}\|^2$, the solution is further simplified in order to obtain the standard deviation of the optimized scaling factor

$$\boxed{\sigma_{s_{\text{opt}}} = \frac{\|\hat{\mathbf{X}} - \hat{\mathbf{Y}}\|}{\sqrt{m+n-1}\|\mathbf{J}\|}}. \tag{A.16}$$

A.11 Depth from Adjoining Two-View Geometries

Given a point correspondence between cameras $\hat{\mathbf{X}}_k \leftrightarrow \hat{\mathbf{X}}_l$ and $\hat{\mathbf{X}}_l \leftrightarrow \hat{\mathbf{X}}_m$ with known camera extrinsics $R_k\hat{\mathbf{X}}_k$, $R_l\hat{\mathbf{X}}_l$ and $R_m\hat{\mathbf{X}}_m$. This method obtains the depth λ_l between $k \leftrightarrow l$ and $m \leftrightarrow l$ as a least-squares solution.
Taking Eq. (6.7), page 92 and deriving the relation for $k \leftrightarrow l$ and $m \leftrightarrow l$ leads to:

$$\begin{aligned}\lambda_l\big(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\big) &= R_k\hat{\mathbf{X}}_k \times \big(\mathbf{t}_k - \mathbf{t}_l\big)\\ \lambda_l\big(R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l\big) &= R_m\hat{\mathbf{X}}_m \times \big(\mathbf{t}_m - \mathbf{t}_l\big)\,.\end{aligned}$$

Solving the problem for λ_l leads to the least-squares solution $\lambda_l = \mathbf{a}^T\mathbf{b}/\|\mathbf{a}\|^2$ with:

$$\begin{aligned}\mathbf{a} &= \begin{bmatrix} R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l \\ R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l \end{bmatrix}\\ \mathbf{b} &= \begin{bmatrix} R_k\hat{\mathbf{X}}_k \times \big(\mathbf{t}_k - \mathbf{t}_l\big) \\ R_m\hat{\mathbf{X}}_m \times \big(\mathbf{t}_m - \mathbf{t}_l\big) \end{bmatrix}.\end{aligned}$$

similiar to the presented one in [177].

A.12 Alternative Three-View Derivation

The following shows a second and third derivation approach to obtain the three-view constraint as presented in Section 7.1, page 125.

A.12.1 Second Derivation Approach

This derivation is based on Eq. (6.7), page 92 applied to cameras $k \leftrightarrow l$ and $m \leftrightarrow l$:

$$R_k\hat{\mathbf{X}}_k \times \big(\mathbf{t}_k - \mathbf{t}_l\big) = \lambda_l\big(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\big) \tag{A.17}$$

$$R_m\hat{\mathbf{X}}_m \times \big(\mathbf{t}_m - \mathbf{t}_l\big) = \lambda_l\big(R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l\big)\,. \tag{A.18}$$

Scalar-multiplying $R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l$ to Eq. (A.17) and $R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l$ to Eq. (A.18) from the left yields:

$$\begin{aligned}\big(R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l\big) \bullet \big(R_k\hat{\mathbf{X}}_k \times \big(\mathbf{t}_k - \mathbf{t}_l\big)\big) &= \lambda_l\big(R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l\big) \bullet \big(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\big)\\ \big(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\big) \bullet \big(R_m\hat{\mathbf{X}}_m \times \big(\mathbf{t}_m - \mathbf{t}_l\big)\big) &= \lambda_l \underbrace{\big(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\big) \bullet \big(R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l\big)}_{\mathbf{a}\bullet\mathbf{b}=\mathbf{b}\bullet\mathbf{a}},\end{aligned}$$

which can be equalized to obtain the three-view constraint

$$\boxed{0 = \big(R_k\hat{\mathbf{X}}_k \times \big(\mathbf{t}_k - \mathbf{t}_l\big)\big) \bullet \big(R_m\hat{\mathbf{X}}_m \times R_l\hat{\mathbf{X}}_l\big) - \big(R_k\hat{\mathbf{X}}_k \times R_l\hat{\mathbf{X}}_l\big) \bullet \big(R_m\hat{\mathbf{X}}_m \times \big(\mathbf{t}_m - \mathbf{t}_l\big)\big)}.$$

A.12.2 Third Derivation Approach

This derivation uses the least-square depth approximation from Eq. (6.11), page 93 between cameras $k \leftrightarrow l$ and $m \leftrightarrow l$:

$$\lambda_l = \frac{\left\| R_k \hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l \right) \right\|}{\left\| R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l \right\|} \tag{A.19}$$

$$\lambda_l = \frac{\left\| R_m \hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l \right) \right\|}{\left\| R_m \hat{\mathbf{X}}_m \times R_l \hat{\mathbf{X}}_l \right\|} . \tag{A.20}$$

Equalizing Eqs. (A.19) and (A.20) yields Eq. (7.9), page 126

$$\left\| R_m \hat{\mathbf{X}}_m \times R_l \hat{\mathbf{X}}_l \right\| \left\| R_k \hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l \right) \right\| = \left\| R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l \right\| \left\| R_m \hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l \right) \right\| ,$$

which is further transformed as Section 7.1, page 125 describes.

A.13 Relation between Trifocal Geometry and Alternative Midpoint Method

Suppose A to be the trifocal constraint from Eq. (7.25), page 131 such that:

$$\begin{aligned} A &= \left(R_k \hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l \right) \right) \left(R_m \hat{\mathbf{X}}_m \times R_l \hat{\mathbf{X}}_l \right)^T - \left(R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l \right) \left(R_m \hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l \right) \right)^T \\ &\triangleq \mathbf{g}_k \mathbf{d}_m^T - \mathbf{d}_k \mathbf{g}_m^T . \end{aligned}$$

The depth functions from Eq. (6.7), page 92 between cameras $k \leftrightarrow l$ and $m \leftrightarrow l$ are given with:

$$\lambda_l = \frac{\left(R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l \right)^T \left(R_k \hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l \right) \right)}{\left(R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l \right)^T \left(R_k \hat{\mathbf{X}}_k \times R_l \hat{\mathbf{X}}_l \right)} \quad \triangleq \frac{\mathbf{d}_k^T \mathbf{g}_k}{\mathbf{d}_k^T \mathbf{d}_k} \tag{A.21}$$

$$\lambda_l = \frac{\left(R_m \hat{\mathbf{X}}_m \times R_l \hat{\mathbf{X}}_l \right)^T \left(R_m \hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l \right) \right)}{\left(R_m \hat{\mathbf{X}}_m \times R_l \hat{\mathbf{X}}_l \right)^T \left(R_m \hat{\mathbf{X}}_m \times R_l \hat{\mathbf{X}}_l \right)} \quad \triangleq \frac{\mathbf{d}_m^T \mathbf{g}_m}{\mathbf{d}_m^T \mathbf{d}_m} , \tag{A.22}$$

as well as the collinear constraint from Eq. (6.7), page 92 applied to both mentioned camera pairs:

$$0 = \left(R_k \hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l \right) \right) \lambda_l - \left(R_k \hat{\mathbf{X}}_k \times \left(\mathbf{t}_k - \mathbf{t}_l \right) \right) \quad \triangleq \mathbf{d}_k \lambda_l - \mathbf{g}_k \tag{A.23}$$

$$0 = \left(R_m \hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l \right) \right) \lambda_l - \left(R_m \hat{\mathbf{X}}_m \times \left(\mathbf{t}_m - \mathbf{t}_l \right) \right) \triangleq \mathbf{d}_m \lambda_l - \mathbf{g}_m . \tag{A.24}$$

Entering Eq. (A.21) in Eq. (A.24) yields:

$$\begin{aligned}
\mathbf{d}_m \underbrace{\mathbf{d}_k^T \mathbf{g}_k}_{\mathbf{a}^T\mathbf{b}=\mathbf{b}^T\mathbf{a}} - \mathbf{g}_m \mathbf{d}_k^T \mathbf{d}_k &= \mathbf{0} \\
\left(\mathbf{d}_m \mathbf{g}_k^T - \mathbf{g}_m \mathbf{d}_k^T\right) \mathbf{d}_k &= \mathbf{0} \\
\underbrace{\left(\mathbf{g}_k \mathbf{d}_m^T - \mathbf{d}_k \mathbf{g}_m^T\right)}_{\hat{=} A}{}^T \mathbf{d}_k &= \mathbf{0} \\
A^T \mathbf{d}_k &= \mathbf{0}\,. \qquad \text{(A.25)}
\end{aligned}$$

Applying Eq. (A.22) to Eq. (A.23) leads to:

$$\begin{aligned}
\mathbf{d}_k \underbrace{\mathbf{d}_m^T \mathbf{g}_m}_{\mathbf{a}^T\mathbf{b}=\mathbf{b}^T\mathbf{a}} - \mathbf{g}_k \mathbf{d}_m^T \mathbf{d}_m &= \mathbf{0} \\
\underbrace{\left(\mathbf{d}_k \mathbf{g}_m^T - \mathbf{g}_k \mathbf{d}_m^T\right)}_{\hat{=} -A} \mathbf{d}_m &= \mathbf{0} \\
A \mathbf{d}_m &= \mathbf{0}\,. \qquad \text{(A.26)}
\end{aligned}$$

Eqs. (A.25) and (A.26) show the direct connection between derived relations from the *alternative* midpoint method (Section 6.2.2, page 91) and the trifocal geometry (Section 7.3, page 131). Consequently, if the trifocal constraint is satisfied, the *alternative* midpoint triangulation is also satisfied.

A.14 Additional Pose Graph Generation Examples

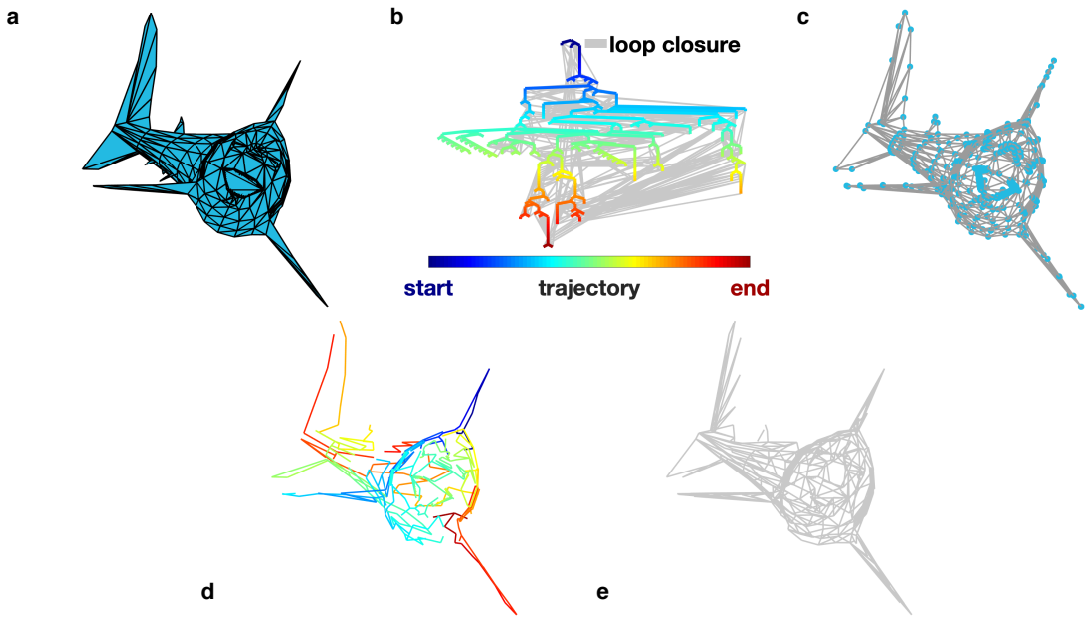

Fig. A.14.3: Pose graph extraction from the mesh of the polygon model *Shark*[98]. The following is shown in: **a)** original mesh, **b)** subgraph indicating the trajectory's pathway from **start** to **end** with branching, ▬ illustrate loop closures. **c)** final pose graph, **d** trajectory's pathway in 3d model, **e)** loop closure edges in 3d model.

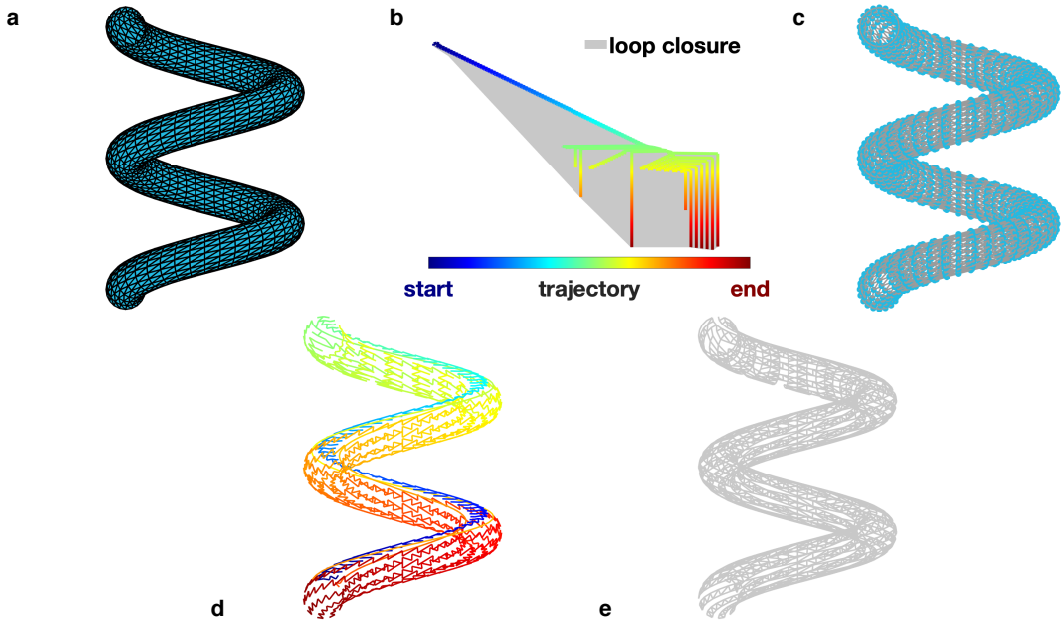

Fig. A.14.4: Pose graph extraction from the mesh of the polygon model *Helix*[99]. The following is shown in: **a)** original mesh, **b)** subgraph indicating the trajectory's pathway from **start** to **end** with branching, ▬ illustrate loop closures. **c)** final pose graph, **d** trajectory's pathway in 3d model, **e)** loop closure edges in 3d model.

[98] https://people.sc.fsu.edu/~jburkardt/data/ply/shark.ply
[99] https://people.sc.fsu.edu/~jburkardt/data/ply/helix.ply

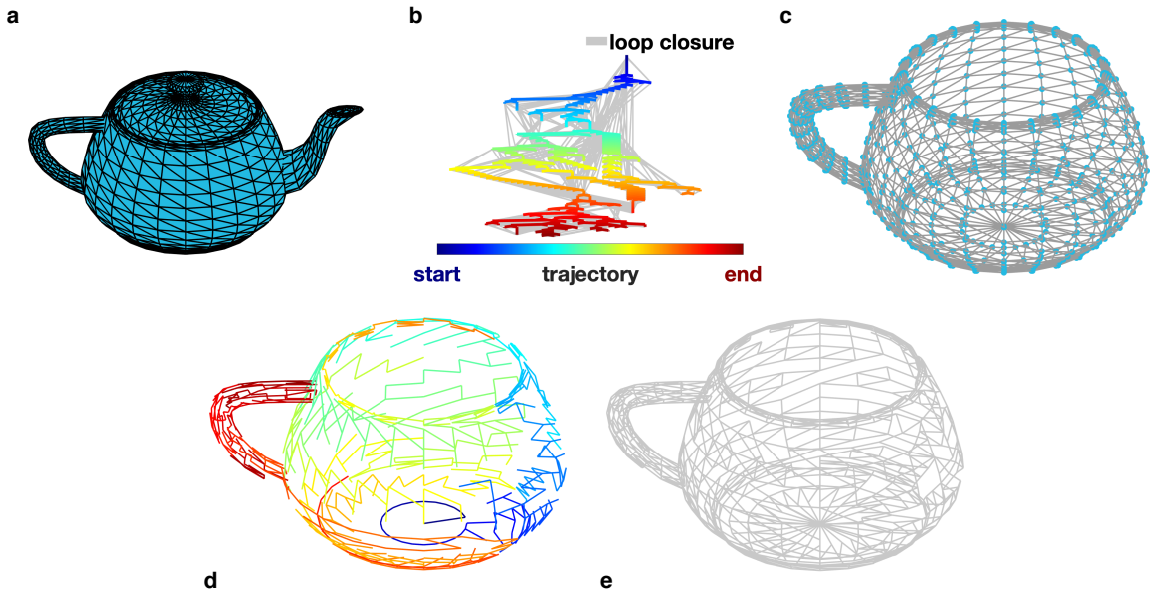

Fig. A.14.5: Pose graph extraction from the mesh of the polygon model *Teapot*[100]. The following is shown in: **a)** original mesh, **b)** subgraph indicating the trajectory's pathway from **start** to **end** with branching, illustrate loop closures. **c)** final pose graph, **d** trajectory's pathway in 3d model, **e)** loop closure edges in 3d model. Some parts of the *Teapot* model are rejected, due to unconnected vertices.

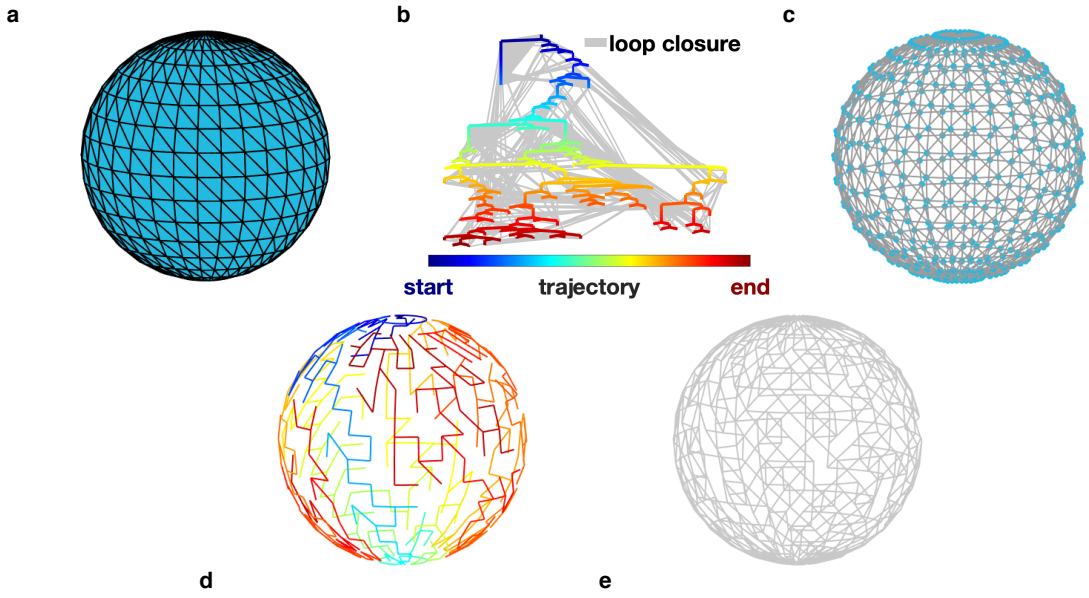

Fig. A.14.6: Pose graph extraction from the mesh of the polygon model *Sphere*[101]. The following is shown in: **a)** original mesh, **b)** subgraph indicating the trajectory's pathway from **start** to **end** with branching, illustrate loop closures. **c)** final pose graph, **d** trajectory's pathway in 3d model, **e)** loop closure edges in 3d model.

[100] https://people.sc.fsu.edu/~jburkardt/data/ply/teapot.ply
[101] https://people.sc.fsu.edu/~jburkardt/data/ply/sphere.ply

A.15 Pose Graph Solver Settings

Tab. A.15.3: Pose graph solver settings

Solver	Settings
g2o	'-v','-i',2000,'-update',1,'-robustKernel','Cauchy', '-solver','dl_var'
GTSAM	'LevenbergMarquardt','optimizeSafely','Verbosity', 'ERROR','MaxIterations',1000,'VerbosityDL','SILENT'
iSAM2	'ISAM2','iter',1000,'Factorization','CHOLESKY', 'RelinearizeSkip',1,'RelinearizeThreshold',0.01, 'CacheLinearizedFactors',1,'EnableDetailedResults',1, 'EnableRelinearization',1,'EvaluateNonlinearError',1, 'EnablePartialRelinearizationCheck',1,
SLAM++	'-mfnsi',100,'-mnsi',50,'-po','-fL','-us'

A.16 Additional Pose Graph Optimization Examples

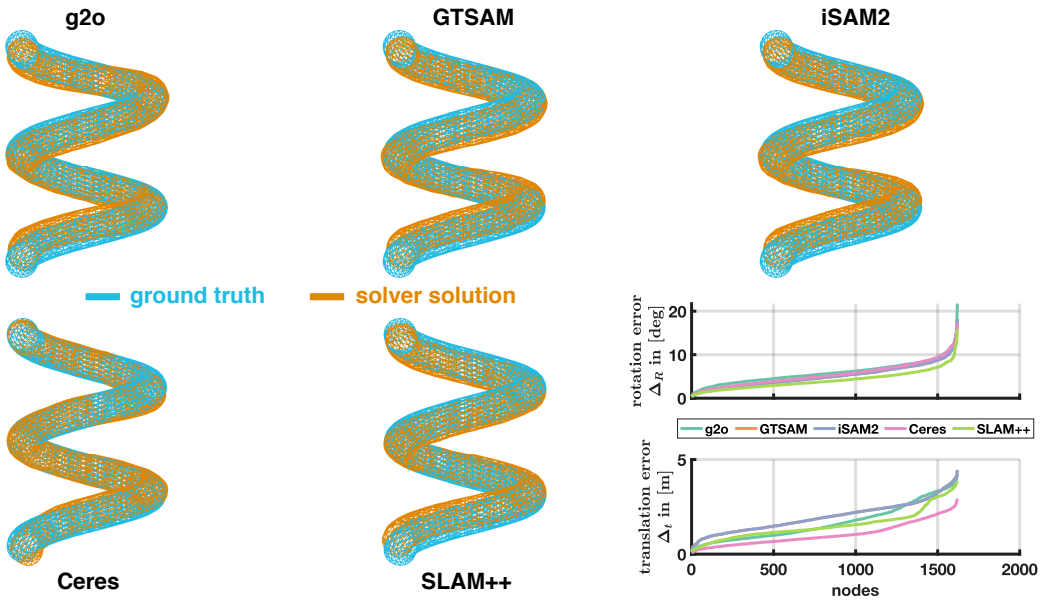

Fig. A.16.7: Pose Graph Optimization: Helix

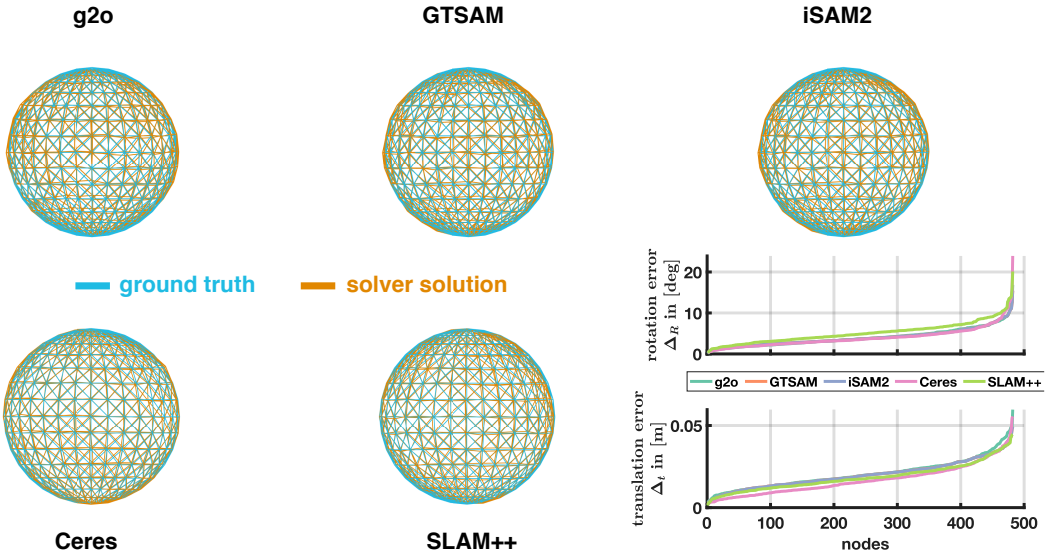

Fig. A.16.8: Pose Graph Optimization: Sphere

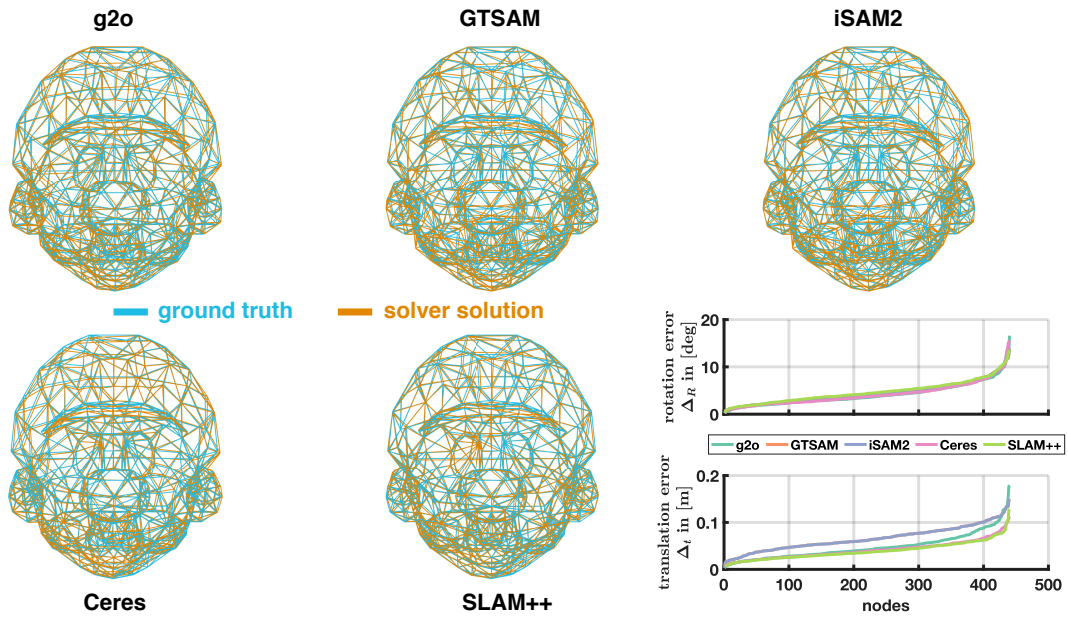

Fig. A.16.9: Pose Graph Optimization: Mario

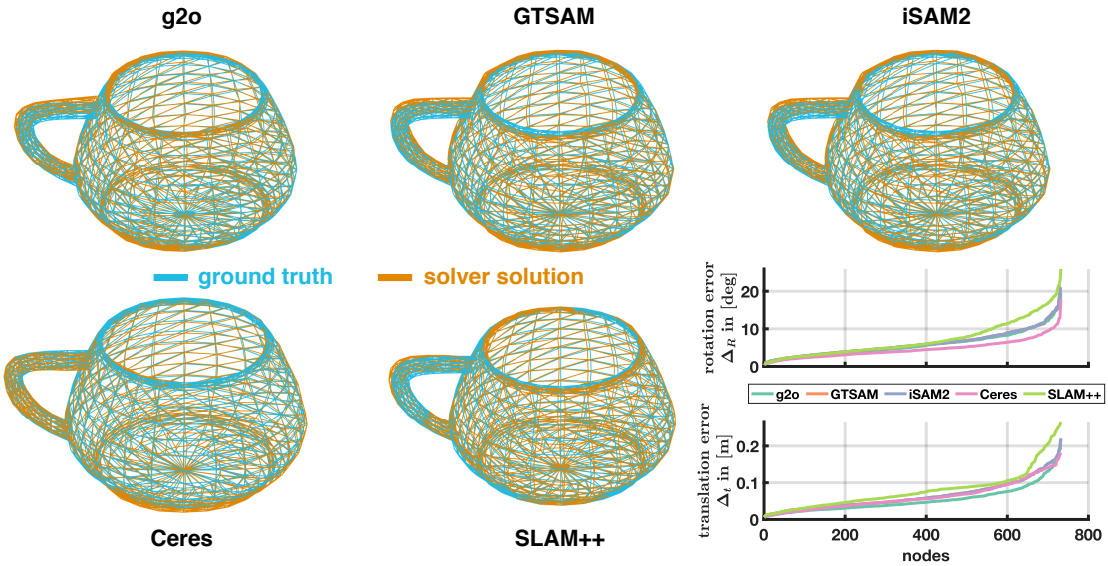

Fig. A.16.10: Pose Graph Optimization: Teapot

Bibliography

[1] P. Agarwal, G. D. Tipaldi, L. Spinello, C. Stachniss, and W. Burgard. "Robust Map Optimization Using Dynamic Covariance Scaling". In: *2013 IEEE International Conference on Robotics and Automation*. Karlsruhe, Germany: IEEE, May 2013, pp. 62–69. ISBN: 978-1-4673-5643-5 978-1-4673-5641-1. DOI: 10.1109/ICRA.2013.6630557 (cit. on pp. 144, 145).

[2] S. Agarwal, Y. Furukawa, N. Snavely, B. Curless, S. Seitz, and R. Szeliski. "Reconstructing Rome". In: *Computer* 43.6 (June 2010), pp. 40–47. ISSN: 0018-9162. DOI: 10.1109/MC.2010.175 (cit. on p. 19).

[3] S. Agarwal, K. Mierle, and Others. *Ceres Solver* (cit. on p. 141).

[4] S. Agarwal, N. Snavely, I. Simon, S. Seitz, and R. Szeliski. "Building Rome in a Day". In: *International Conference on Computer Vision*. Kyoto, Japan, 2009 (cit. on pp. 19, 21).

[5] S. Aghayari, M. Saadatseresht, M. Omidalizarandi, and I. Neumann. "GEOMETRIC CALIBRATION OF FULL SPHERICAL PANORAMIC RICOH-THETA CAMERA". In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* IV-1/W1 (May 2017), pp. 237–245. ISSN: 2194-9050. DOI: 10.5194/isprs-annals-IV-1-W1-237-2017 (cit. on p. 77).

[6] P. F. Alcantarilla, A. Bartoli, and A. J. Davison. "KAZE Features". In: *Proceedings of the 12th European Conference on Computer Vision - Volume Part VI*. ECCV'12. Berlin, Heidelberg: Springer-Verlag, Oct. 2012, pp. 214–227. ISBN: 978-3-642-33782-6. DOI: 10.1007/978-3-642-33783-3_16 (cit. on p. 161).

[7] I. Aloise and G. Grisetti. "Chordal Based Error Function for 3-D Pose-Graph Optimization". In: *IEEE Robotics and Automation Letters* 5.1 (Jan. 2020), pp. 274–281. ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2019.2956456 (cit. on p. 140).

[8] R. Arandjelovic and A. Zisserman. "Three Things Everyone Should Know to Improve Object Retrieval". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. Providence, RI: IEEE, June 2012, pp. 2911–2918. ISBN:

978-1-4673-1228-8 978-1-4673-1226-4 978-1-4673-1227-1. DOI: 10.1109/CVPR.2012.6248018 (cit. on p. 161).

[9] M. Arie-Nachimson, S. Z. Kovalsky, I. Kemelmacher-Shlizerman, A. Singer, and R. Basri. "Global Motion Estimation from Point Matches". In: IEEE, Oct. 2012, pp. 81–88. ISBN: 978-0-7695-4873-9 978-1-4673-4470-8. DOI: 10.1109/3DIMPVT.2012.46 (cit. on p. 22).

[10] S. Baker and S. K. Nayar. "A Theory of Single-Viewpoint Catadioptric Image Formation". In: *International Journal of Computer Vision* 35.2 (Nov. 1999), pp. 175–196. ISSN: 1573-1405. DOI: 10.1023/A:1008128724364 (cit. on p. 35).

[11] J. Barreto and H. Araujo. "Issues on the Geometry of Central Catadioptric Image Formation". In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*. Vol. 2. Kauai, HI, USA: IEEE Comput. Soc, 2001, pp. II–422–II–427. ISBN: 978-0-7695-1272-3. DOI: 10.1109/CVPR.2001.990992 (cit. on p. 35).

[12] H. Bay, T. Tuytelaars, and L. Van Gool. "SURF: Speeded Up Robust Features". In: *Computer Vision – ECCV 2006*. Ed. by A. Leonardis, H. Bischof, and A. Pinz. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2006, pp. 404–417. ISBN: 978-3-540-33833-8. DOI: 10.1007/11744023_32 (cit. on p. 161).

[13] P. A. Beardsley, A. Zisserman, and D. W. Murray. "Navigation Using Affine Structure from Motion". In: *Computer Vision — ECCV '94*. Ed. by J.-O. Eklundh. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 1994, pp. 85–96. ISBN: 978-3-540-48400-4. DOI: 10.1007/BFb0028337 (cit. on p. 91).

[14] P. Beardsley, A. Zisserman, and D. Murray. "Sequential Updating of Projective and Affine Structure from Motion". In: *International Journal of Computer Vision* 23.3 (June 1997), pp. 235–259. ISSN: 1573-1405. DOI: 10.1023/A:1007923216416 (cit. on p. 91).

[15] N. Benecke, H. Engelhardt, A. Heyduk, M. Jendryz, B. Jung, H. Kleta, P. Koch, J. König, S. Leighton, P. O'Leary, S. May, S. Rapp, M. Sastuba, A. Schischmanow, M. Weber, and S. Zuev. "iDeepMon – Intelligente Inspektion und Überwachung von tiefen Bergwerksschächten". In: *BHM Berg- und Hüttenmännische Monatshefte* 162.10 (Sept. 2017), pp. 430–433. ISSN: 0005-8912, 1613-7531. DOI: 10.1007/s00501-017-0648-x (cit. on p. 85).

[16] B. Bhowmick, S. Patra, A. Chatterjee, V. M. Govindu, and S. Banerjee. "Divide and Conquer: Efficient Large-Scale Structure from Motion Using Graph Partitioning". In: *Computer Vision – ACCV 2014*. Ed. by D. Cremers, I. Reid, H. Saito, and M.-H. Yang. Vol. 9004. Cham: Springer International Publishing, 2015, pp. 273–287. ISBN: 978-3-319-16807-4 978-3-319-16808-1. DOI: 10.1007/978-3-319-16808-1_19 (cit. on p. 24).

[17] J. L. Blanco-Claraco. "A Tutorial on SE(3) Transformation Parameterizations and on-Manifold Optimization". In: *arXiv:2103.15980 [cs]* (Mar. 2021). arXiv: 2103.15980 [cs] (cit. on pp. 67, 181).

[18] Y. Bok, D.-G. Choi, and I. Kweon. "Sensor Fusion of Cameras and a Laser for City-Scale 3D Reconstruction". In: *Sensors* 14.11 (Nov. 2014), pp. 20882–20909. ISSN: 1424-8220. DOI: 10.3390/s141120882 (cit. on p. 18).

[19] J.-Y. Bouguet. *Camera Calibration Toolbox for Matlab*. California Institute of Technology. 2004 (cit. on pp. 37, 38, 51, 53, 54, 58, 170).

[20] J. Briales, L. Kneip, and J. Gonzalez-Jimenez. "A Certifiably Globally Optimal Solution to the Non-minimal Relative Pose Problem". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. June 2018, pp. 145–154. DOI: 10.1109/CVPR.2018.00023 (cit. on p. 107).

[21] J. Briales and J. Gonzalez-Jimenez. "Cartan-Sync: Fast and Global SE(d)-Synchronization". In: *IEEE Robotics and Automation Letters* 2.4 (Oct. 2017), pp. 2127–2134. ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2017.2718661 (cit. on p. 144).

[22] D. C. Brown. "Decentering Distortion of Lenses". In: *PHOTOGRAMMETRIC ENGINEERING* (1966), p. 19 (cit. on pp. 37, 55).

[23] D. C. Brown. "Close-Range Camera Calibration". In: *Photogrammetric Engineering* 37.8 (1971), pp. 855–866 (cit. on pp. 37, 54).

[24] Q. Cai, Y. Wu, L. Zhang, and P. Zhang. "Equivalent Constraints for Two-View Geometry: Pose Solution/Pure Rotation Identification and 3D Reconstruction". In: *International Journal of Computer Vision* 127.2 (Feb. 2019), pp. 163–180. ISSN: 0920-5691, 1573-1405. DOI: 10.1007/s11263-018-1136-9 (cit. on pp. 97, 98, 119).

[25] G. Calafiore, L. Carlone, and F. Dellaert. "Pose Graph Optimization in the Complex Domain: Lagrangian Duality, Conditions For Zero Duality Gap, and Optimal Solutions". In: *arXiv:1505.03437 [cs]* (May 2015). arXiv: 1505.03437 [cs] (cit. on p. 139).

[26] M. Calonder, ed. *EKF SLAM vs. FastSLAM – A Comparison*. 2006 (cit. on p. 12).

[27] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós. "ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM". In: *IEEE Transactions on Robotics* (2021), pp. 1–17. ISSN: 1552-3098, 1941-0468. DOI: 10.1109/TRO.2021.3075644. arXiv: 2007.11898 (cit. on pp. 16, 18, 119).

[28] L. Carlone. *G2o versus Toro: Format and Cost Functions*. https://lucacarlone.mit.edu/notes/ (cit. on p. 139).

[29] L. Carlone, R. Aragues, J. Castellanos, and B. Bona. "A Linear Approximation for Graph-based Simultaneous Localization and Mapping". In: June 2011. DOI: 10.15607/RSS.2011.VII.006 (cit. on p. 142).

[30] L. Carlone, R. Aragues, J. A. Castellanos, and B. Bona. "A First-Order Solution to Simultaneous Localization and Mapping with Graphical Models". In: *2011 IEEE International Conference on Robotics and Automation*. May 2011, pp. 1764–1771. DOI: 10.1109/ICRA.2011.5979890 (cit. on p. 142).

[31] L. Carlone and A. Censi. "From Angular Manifolds to the Integer Lattice: Guaranteed Orientation Estimation With Application to Pose Graph Optimization". In: *IEEE Transactions on Robotics* 30.2 (Apr. 2014), pp. 475–492. ISSN: 1552-3098, 1941-0468. DOI: 10.1109/TRO.2013.2291626 (cit. on pp. 140, 142).

[32] L. Carlone, D. Rosen, G. Calafiore, J. Leonard, and F. Dellaert. "Lagrangian Duality in 3D SLAM: Verification Techniques and Optimal Solutions". In: *arXiv:1506.00746 [cs, math]* (July 2015). arXiv: 1506.00746 [cs, math] (cit. on pp. 139, 142, 147).

[33] L. Carlone, R. Tron, K. Daniilidis, and F. Dellaert. "Initialization Techniques for 3D SLAM: A Survey on Rotation Estimation and Its Use in Pose Graph Optimization". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. Seattle, WA, USA: IEEE, May 2015, pp. 4597–4604. ISBN: 978-1-4799-6923-4. DOI: 10.1109/ICRA.2015.7139836 (cit. on pp. 139, 142, 147).

[34] G. Caron and D. Eynard. "Multiple Camera Types Simultaneous Stereo Calibration". In: *IEEE International Conference on Robotics and Automation*. IEEE, May 2011, pp. 2933–2938. ISBN: 978-1-61284-386-5. DOI: 10.1109/ICRA.2011.5979975 (cit. on p. 58).

[35] G. Caron, E. M. Mouaddib, and E. Marchand. "Single Viewpoint Stereoscopic Sensor Calibration". In: *5th International Symposium On I/V Communications and Mobile Network*. IEEE, Sept. 2010, pp. 1–4. ISBN: 978-1-4244-5996-4. DOI: 10.1109/ISVC.2010.5656150 (cit. on p. 58).

[36] A. Chatterjee and V. M. Govindu. "Efficient and Robust Large-Scale Rotation Averaging". In: *2013 IEEE International Conference on Computer Vision*. Sydney, Australia: IEEE, Dec. 2013, pp. 521–528. ISBN: 978-1-4799-2840-8. DOI: 10.1109/ICCV.2013.70 (cit. on p. 22).

[37] J. Chen, D. Wu, P. Song, F. Deng, Y. He, and S. Pang. "Multi-View Triangulation: Systematic Comparison and an Improved Method". In: *IEEE Access* 8 (2020), pp. 21017–21027. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.2969082 (cit. on pp. 22, 91).

[38] Y. Chen, S. Shen, Y. Chen, and G. Wang. "Graph-Based Parallel Large Scale Structure from Motion". In: *arXiv:1912.10659 [cs]* (June 2020). arXiv: 1912.10659 [cs] (cit. on pp. 21, 24).

[39] A. Cohen, T. Sattler, and M. Pollefeys. "Merging the Unmatchable: Stitching Visually Disconnected SfM Models". In: IEEE, Dec. 2015, pp. 2129–2137. ISBN: 978-1-4673-8391-2. DOI: 10.1109/ICCV.2015.246 (cit. on p. 24).

[40] K. Connor and I. Reid. "Novel View Specification and Synthesis". In: *Procedings of the British Machine Vision Conference 2002*. Cardiff: British Machine Vision Association, 2002, pp. 22.1–22.10. ISBN: 978-1-901725-19-3. DOI: 10.5244/C.16.22 (cit. on p. 129).

[41] A. E. Conrady. "Decentred Lens-Systems". In: *Monthly Notices of the Royal Astronomical Society* 79.5 (Mar. 1919), pp. 384–390. ISSN: 0035-8711. DOI: 10.1093/mnras/79.5.384 (cit. on pp. 37, 55).

[42] S. Cortés, A. Solin, E. Rahtu, and J. Kannala. "ADVIO: An Authentic Dataset for Visual-Inertial Odometry". In: *arXiv:1807.09828 [cs]* (July 2018). arXiv: 1807.09828 [cs] (cit. on p. 2).

[43] J. Courbon and Y. Mezouar. "Evaluation of the Unified Model of the Sphere for Fisheye Cameras in Robotic Applications". In: *Advanced Robotics* (2013), p. 22 (cit. on p. 35).

[44] D. J. Crandall, A. Owens, N. Snavely, and D. P. Huttenlocher. "SfM with MRFs: Discrete-Continuous Optimization for Large-Scale Structure from Motion". In: *IEEE transactions on pattern analysis and machine intelligence* 35.12 (Dec. 2013), pp. 2841–2853. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2012.218 (cit. on p. 22).

[45] N. Crombez, R. Seulin, O. Morel, D. Fofi, and C. Demonceaux. "Multimodal 2D Image to 3D Model Registration via a Mutual Alignment of Sparse and Dense Visual Features". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. Brisbane, QLD: IEEE, May 2018, pp. 6316–6322. ISBN: 978-1-5386-3081-5. DOI: 10.1109/ICRA.2018.8461092 (cit. on p. 20).

[46] Z. Cui and P. Tan. "Global Structure-from-Motion by Similarity Averaging". In: *2015 IEEE International Conference on Computer Vision (ICCV)* (2015). DOI: 10.1109/ICCV.2015.105 (cit. on p. 22).

[47] F. Dellaert. *Factor Graphs and GTSAM: A Hands-on Introduction*. Technical Report GT-RIM-CP&R-2012-002. Sept. 2012, p. 27 (cit. on p. 140).

[48] F. Dellaert and M. Kaess. "Square Root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing". In: *The International Journal of Robotics Research* 25.12 (Dec. 2006), pp. 1181–1203. ISSN: 0278-3649, 1741-3176. DOI: 10.1177/0278364906072768 (cit. on pp. 12, 13, 141).

[49] F. Dellaert and M. Kaess. "Factor Graphs for Robot Perception". In: *Foundations and Trends in Robotics* 6.1-2 (2017), pp. 1–139. ISSN: 1935-8253, 1935-8261. DOI: 10.1561/2300000043 (cit. on pp. 13, 15, 140).

[50] T. Ding, Y. Yang, Z. Zhu, D. P. Robinson, R. Vidal, L. Kneip, and M. C. Tsakiris. "Robust Homography Estimation via Dual Principal Component Pursuit". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA: IEEE, June 2020, pp. 6079–6088. ISBN: 978-1-72817-168-5. DOI: 10.1109/CVPR42600.2020.00612 (cit. on p. 121).

[51] J. Dong and Z. Lv. "miniSAM: A Flexible Factor Graph Non-linear Least Squares Optimization Framework". In: *arXiv:1909.00903 [cs]* (Sept. 2019). arXiv: 1909.00903 [cs] (cit. on p. 143).

[52] G. Dubbelman and B. Browning. "Closed-Form Online Pose-chain SLAM". In: *2013 IEEE International Conference on Robotics and Automation*. Karlsruhe, Germany: IEEE, May 2013, pp. 5190–5197. ISBN: 978-1-4673-5643-5 978-1-4673-5641-1. DOI: 10.1109/ICRA.2013.6631319 (cit. on p. 143).

[53] T. El-Ganainy and M. Hefeeda. "Streaming Virtual Reality Content". In: *arXiv:1612.08350 [cs]* (Dec. 2016). arXiv: 1612.08350 [cs] (cit. on p. 77).

[54] B. S. Everitt and D. C. Howell, eds. *Encyclopedia of Statistics in Behavioral Science*. Vol. 2. Chichester, UK: John Wiley & Sons, Ltd, Oct. 2005. ISBN: 978-0-470-86080-9 978-0-470-01319-9. DOI: 10.1002/0470013192 (cit. on p. 182).

[55] M. Fang, T. Pollok, and C. Qu. "Merge-SfM: Merging Partial Reconstructions". In: *30th British Machine Vision Conference 2019, BMVC 2019, Cardiff, UK, September 9-12, 2019*. BMVA Press, 2019, p. 29 (cit. on p. 24).

[56] Q. Fang. "Vision-Aided Localization and Navigation Based on Trifocal Tensor". In: *arXiv:1611.03538 [cs]* (Nov. 2016). arXiv: 1611.03538 [cs] (cit. on p. 133).

[57] M. Farenzena, A. Fusiello, and R. Gherardi. "Structure-and-Motion Pipeline on a Hierarchical Cluster Tree". In: *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*. Kyoto, Japan: IEEE, Sept. 2009, pp. 1489–1496. ISBN: 978-1-4244-4442-7. DOI: 10.1109/ICCVW.2009.5457435 (cit. on p. 21).

[58] K. Fathian and N. R. Gans. "A New Approach for Solving the Five-Point Relative Pose Problem for Vision-Based Estimation and Control". In: *2014 American Control Conference*. Portland, OR, USA: IEEE, June 2014, pp. 103–109. ISBN: 978-1-4799-3274-0 978-1-4799-3272-6 978-1-4799-3271-9. DOI: 10.1109/ACC.2014.6859364 (cit. on p. 101).

[59] K. Fathian, J. P. Ramirez-Paredes, E. A. Doucette, J. W. Curtis, and N. R. Gans. "Quaternion Based Camera Pose Estimation From Matched Feature Points". In: *IEEE Robotics and Automation Letters* 3.2 (Apr. 2018), pp. 857–864. ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2018.2792142. arXiv: 1704.02672 (cit. on pp. 101, 105).

[60] K. Fathian, J. P. Ramirez-Paredes, E. A. Doucette, J. W. Curtis, and N. R. Gans. "QuEst: A Quaternion-Based Approach for Camera Motion Estimation From Minimal Feature Points". In: *IEEE Robotics and Automation Letters* 3.2 (Apr. 2018), pp. 857–864. ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2018.2792142 (cit. on pp. 101, 102, 104, 107).

[61] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. Cambridge, MA, USA: MIT Press, 1993. ISBN: 978-0-262-06158-2 (cit. on p. 101).

[62] O. D. Faugeras and S. Maybank. "Motion from Point Matches: Multiplicity of Solutions". In: *International Journal of Computer Vision* 4.3 (June 1990), pp. 225–246. ISSN: 1573-1405. DOI: 10.1007/BF00054997 (cit. on p. 101).

[63] M. Ferber, M. Sastuba, S. Grehl, and B. Jung. "Combining SURF and SIFT for Challenging Indoor Localization Using a Feature Cloud". In: *7th International Conference on Indoor Positioning and Indoor Navigation*. 2016, p. 4 (cit. on p. 161).

[64] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza. "SVO: Semidirect Visual Odometry for Monocular and Multicamera Systems". In: *IEEE Transactions on Robotics* 33.2 (Apr. 2017), pp. 249–265. ISSN: 1552-3098, 1941-0468. DOI: 10.1109/TRO.2016.2623335 (cit. on pp. 17, 18).

[65] C.-W. Fu, L. Wan, T.-T. Wong, and C.-S. Leung. "The Rhombic Dodecahedron Map: An Efficient Scheme for Encoding Panoramic Video". In: *IEEE Transactions on Multimedia* 11.4 (June 2009), pp. 634–644. ISSN: 1941-0077. DOI: 10.1109/TMM.2009.2017626 (cit. on p. 77).

[66] J. Fujiki. "Three Types of Reprojection Error on Spherical Epipolar Geometry". In: *The 8th Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras - OMNIVIS*. Oct. 2008 (cit. on p. 108).

[67] J. Fujiki, A. Torii, and S. Akaho. "Epipolar Geometry via Rectification of Spherical Images". In: *Proceedings of the 3rd International Conference on Computer Vision/Computer Graphics Collaboration Techniques*. MIRAGE'07. Berlin, Heidelberg: Springer-Verlag, Mar. 2007, pp. 461–471. ISBN: 978-3-540-71456-9 (cit. on p. 108).

[68] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. "Towards Internet-scale Multi-View Stereo". In: IEEE, June 2010, pp. 1434–1441. ISBN: 978-1-4244-6984-0. DOI: 10.1109/CVPR.2010.5539802 (cit. on p. 21).

[69] Y. Furukawa and C. Hernández. "Multi-View Stereo: A Tutorial". In: *Foundations and Trends® in Computer Graphics and Vision* 9.1-2 (2015), pp. 1–148. ISSN: 1572-2740, 1572-2759. DOI: 10.1561/0600000052 (cit. on p. 21).

[70] Y. Furukawa and J. Ponce. "Accurate, Dense, and Robust Multi-View Stereopsis". In: 1.1 (2008), p. 14 (cit. on p. 21).

[71] P. V. Gakne. "Tackling the Scale Factor Issue in a Monocular Visual Odometry Using a 3D City Model". In: *International Technical Symposium on Navigation and Timing 2018*. ENAC, Nov. 2018. DOI: 10.31701/itsnt2018.20 (cit. on p. 18).

[72] G. Gallego, E. Mueggler, and P. Sturm. “Translation of "Zur Ermittlung Eines Objektes Aus Zwei Perspektiven Mit Innerer Orientierung" by Erwin Kruppa (1913)”. In: *arXiv:1801.01454 [cs]* (Dec. 2017). arXiv: 1801.01454 [cs] (cit. on pp. 95, 101).

[73] A. Geiger, F. Moosmann, O. Car, and B. Schuster. “Automatic Camera and Range Sensor Calibration Using a Single Shot”. In: *International Conference on Robotics and Automation*. IEEE, May 2012, pp. 3936–3943. ISBN: 978-1-4673-1405-3 978-1-4673-1403-9 978-1-4673-1578-4 978-1-4673-1404-6. DOI: 10.1109/ICRA.2012.6224570 (cit. on pp. 53, 54).

[74] A. Geiger, J. Ziegler, and C. Stiller. “StereoScan: Dense 3d Reconstruction in Real-Time”. In: IEEE, June 2011, pp. 963–968. ISBN: 978-1-4577-0890-9. DOI: 10.1109/IVS.2011.5940405 (cit. on p. 18).

[75] C. Geyer and K. Daniilidis. “A Unifying Theory for Central Panoramic Systems and Practical Implications”. In: *Computer Vision — ECCV 2000*. Ed. by D. Vernon. Vol. 1843. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 445–461. ISBN: 978-3-540-67686-7 978-3-540-45053-5. DOI: 10.1007/3-540-45053-X_29 (cit. on p. 35).

[76] R. Gherardi, M. Farenzena, and A. Fusiello. “Improving the Efficiency of Hierarchical Structure-and-Motion”. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. San Francisco, CA, USA: IEEE, June 2010, pp. 1594–1600. ISBN: 978-1-4244-6984-0. DOI: 10.1109/CVPR.2010.5539782 (cit. on p. 21).

[77] D. Ghosh and N. Kaabouch. “A Survey on Image Mosaicing Techniques”. In: *Journal of Visual Communication and Image Representation* 34 (Jan. 2016), pp. 1–11. ISSN: 10473203. DOI: 10.1016/j.jvcir.2015.10.014 (cit. on p. 76).

[78] T. Goldstein, P. Hand, C. Lee, V. Voroninski, and S. Soatto. “ShapeFit and ShapeKick for Robust, Scalable Structure from Motion”. In: *arXiv:1608.02165 [cs, math]* (Aug. 2016). arXiv: 1608.02165 [cs, math] (cit. on p. 22).

[79] V. Govindu. “Combining Two-View Constraints for Motion Estimation”. In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*. Vol. 2. Kauai, HI, USA: IEEE Comput. Soc, 2001, pp. II–218–II–225. ISBN: 978-0-7695-1272-3. DOI: 10.1109/CVPR.2001.990963 (cit. on p. 26).

[80] J. Grater, T. Schwarze, and M. Lauer. "Robust Scale Estimation for Monocular Visual Odometry Using Structure from Motion and Vanishing Points". In: *2015 IEEE Intelligent Vehicles Symposium (IV)*. Seoul, South Korea: IEEE, June 2015, pp. 475–480. ISBN: 978-1-4673-7266-4. DOI: 10.1109/IVS.2015.7225730 (cit. on p. 18).

[81] S. Grehl, M. Ferber, C. Buhl, M. Sastuba, M. Donner, P. Poschmann, F. Schreiter, M. Herrmann, E. Berger, D. Vogt, and B. Jung. "A Comparing View on Robot Odometry in Underground Mining". In: *Proceedings the International Conference on Intelligent Robots and Systems*. 2016, pp. 1–2. ISBN: 978-3-319-32552-1. DOI: 10.13140/RG.2.2.25921.15209 (cit. on p. 2).

[82] S. Grehl, M. Sastuba, M. Donner, M. Ferber, F. Schreiter, H. Mischo, and B. Jung. "Towards Virtualization of Underground Mines Using Mobile Robots – from 3D Scans to Virtual Mines". In: *23rd International Symposium on Mine Planing & Equipment Selection*. 2015, p. 12 (cit. on pp. 2, 170).

[83] G Grisetti, R Kummerle, C Stachniss, and W Burgard. "A Tutorial on Graph-Based SLAM". In: *IEEE Intelligent Transportation Systems Magazine* 2.4 (24), pp. 31–43. ISSN: 1939-1390. DOI: 10.1109/MITS.2010.939925 (cit. on p. 13).

[84] G. Grisetti, C. Stachniss, and W. Burgard. "Nonlinear Constraint Network Optimization for Efficient Map Learning". In: *IEEE Transactions on Intelligent Transportation Systems* 10.3 (Sept. 2009), pp. 428–439. ISSN: 1524-9050, 1558-0016. DOI: 10.1109/TITS.2009.2026444 (cit. on p. 142).

[85] G. Grisetti, S. Grzonka, C. Stachniss, P. Pfaff, and W. Burgard. "Efficient Estimation of Accurate Maximum Likelihood Maps in 3D". In: *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. San Diego, CA, USA: IEEE, Oct. 2007, pp. 3472–3478. ISBN: 978-1-4244-0911-2 978-1-4244-0912-9. DOI: 10.1109/IROS.2007.4399030 (cit. on p. 142).

[86] G. Grisetti, T. Guadagnino, I. Aloise, M. Colosi, B. Della Corte, and D. Schlegel. "Least Squares Optimization: From Theory to Practice". In: *Robotics* 9.3 (July 2020), p. 51. ISSN: 2218-6581. DOI: 10.3390/robotics9030051 (cit. on p. 140).

[87] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. "A Tree Parameterization for Efficiently Computing Maximum Likelihood Maps Using Gradient Descent". In: June 2007. DOI: 10.15607/RSS.2007.III.009 (cit. on p. 142).

[88] C. Griwodz, S. Gasparini, L. Calvet, P. Gurdjos, F. Castan, B. Maujean, Y. Lanthony, and G. De Lillo. "AliceVision Meshroom: An Open-Source 3D Reconstruction Pipeline". In: *12th ACM Multimedia Systems Conference (MMSys 2021)*. Istanbul, Turkey: ACM: Association for Computing Machinery, Sept. 2021, pp. 241–247. DOI: 10.1145/3458305.3478443 (cit. on p. 20).

[89] L. E. Gurrieri and E. Dubois. "Acquisition of Omnidirectional Stereoscopic Images and Videos of Dynamic Scenes: A Review". In: *Journal of Electronic Imaging* 22.3 (July 2013), p. 030902. ISSN: 1017-9909, 1560-229X. DOI: 10.1117/1.JEI.22.3.030902 (cit. on p. 77).

[90] C. Häne, L. Heng, G. H. Lee, F. Fraundorfer, P. Furgale, T. Sattler, and M. Pollefeys. "3D Visual Perception for Self-Driving Cars Using a Multi-Camera System: Calibration, Mapping, Localization, and Obstacle Detection". In: *arXiv:1708.09839 [cs]* (Aug. 2017). arXiv: 1708.09839 [cs] (cit. on p. 2).

[91] M. J. Harker and P. L. O'Leary. "First Order Geometric Distance (The Myth of Sampsonus)". In: *Procedings of the British Machine Vision Conference 2006*. Edinburgh: British Machine Vision Association, 2006, pp. 10.1–10.10. ISBN: 978-1-901725-32-2. DOI: 10.5244/C.20.10 (cit. on p. 108).

[92] K. Harsányi, A. Kiss, T. Szirányi, and A. Majdik. "MASAT: A Fast and Robust Algorithm for Pose-Graph Initialization". In: *Pattern Recognition Letters* 129 (Jan. 2020), pp. 131–136. ISSN: 01678655. DOI: 10.1016/j.patrec.2019.11.010 (cit. on p. 142).

[93] R. Hartley, R. Gupta, and T. Chang. "Stereo from Uncalibrated Cameras". In: *Proceedings 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Champaign, IL, USA: IEEE Comput. Soc. Press, 1992, pp. 761–764. ISBN: 978-0-8186-2855-9. DOI: 10.1109/CVPR.1992.223179 (cit. on p. 89).

[94] R. Hartley. "A Linear Method for Reconstruction from Lines and Points". In: *Proceedings of IEEE International Conference on Computer Vision*. June 1995, pp. 882–887. DOI: 10.1109/ICCV.1995.466843 (cit. on p. 131).

[95] R. Hartley, J. Trumpf, Y. Dai, and H. Li. "Rotation Averaging". In: *International Journal of Computer Vision* 103.3 (July 2013), pp. 267–305. ISSN: 0920-5691, 1573-1405. DOI: 10.1007/s11263-012-0601-0 (cit. on p. 140).

[96] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge: Cambridge University Press, 2004. ISBN: 978-0-511-18618-9 (cit. on pp. 13, 22, 88, 90, 91, 96, 97, 104, 108, 122, 128–131).

[97] R. I. Hartley. "Estimation of Relative Camera Positions for Uncalibrated Cameras". In: *Computer Vision — ECCV'92*. Ed. by G. Goos, J. Hartmanis, and G. Sandini. Vol. 588. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 579–587. ISBN: 978-3-540-55426-4 978-3-540-47069-4. DOI: 10.1007/3-540-55426-2_62 (cit. on p. 95).

[98] R. I. Hartley. "Cheirality Invariants". In: *In Proc. Darpa Image Understanding Workshop*. 1993, pp. 745–753 (cit. on p. 97).

[99] R. I. Hartley. "Lines and Points in Three Views and the Trifocal Tensor". In: *International Journal of Computer Vision* 22.2 (Mar. 1997), pp. 125–140. ISSN: 1573-1405. DOI: 10.1023/A:1007936012022 (cit. on p. 131).

[100] R. I. Hartley. "Chirality". In: *International Journal of Computer Vision* 26.1 (Jan. 1998), pp. 41–61. ISSN: 1573-1405. DOI: 10.1023/A:1007984508483 (cit. on p. 97).

[101] R. I. Hartley and P. Sturm. "Triangulation". In: *Computer Vision and Image Understanding* 68.2 (Nov. 1997), pp. 146–157. ISSN: 1077-3142. DOI: 10.1006/cviu.1997.0547 (cit. on pp. 22, 88, 90, 91).

[102] M. Havlena and K. Schindler. "VocMatch: Efficient Multiview Correspondence for Structure from Motion". In: *Computer Vision – ECCV 2014*. Ed. by D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars. Vol. 8691. Cham: Springer International Publishing, 2014, pp. 46–60. ISBN: 978-3-319-10577-2 978-3-319-10578-9. DOI: 10.1007/978-3-319-10578-9_4 (cit. on p. 21).

[103] M. Havlena, A. Torii, J. Knopp, and T. Pajdla. "Randomized Structure from Motion Based on Atomic 3D Models from Camera Triplets". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. Miami, FL: IEEE, June 2009, pp. 2874–2881. ISBN: 978-1-4244-3992-8. DOI: 10.1109/CVPR.2009.5206677 (cit. on p. 24).

[104] M. Havlena, A. Torii, and T. Pajdla. "Efficient Structure from Motion by Graph Optimization". In: *Computer Vision – ECCV 2010*. Ed. by K. Daniilidis, P. Maragos, and N. Paragios. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2010, pp. 100–113. ISBN: 978-3-642-15552-9. DOI: 10.1007/978-3-642-15552-9_8 (cit. on p. 24).

[105] J. Heikkila and O. Silven. "A Four-Step Camera Calibration Procedure with Implicit Image Correction". In: *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. San Juan, Puerto Rico:

IEEE Comput. Soc, 1997, pp. 1106–1112. ISBN: 978-0-8186-7822-6. DOI: 10.1109/CVPR.1997.609468 (cit. on pp. 38, 54).

[106] J. Heinly, J. L. Schonberger, E. Dunn, and J.-M. Frahm. "Reconstructing the World* in Six Days". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA, USA: IEEE, June 2015, pp. 3287–3295. ISBN: 978-1-4673-6964-0. DOI: 10.1109/CVPR.2015.7298949 (cit. on p. 19).

[107] D. Helmick, Yang Cheng, and S. Roumeliotis. "Path Following Using Visual Odometry for a Mars Rover in High-Slip Environments". In: *2004 IEEE Aerospace Conference Proceedings (IEEE Cat. No.04TH8720)*. Vol. 2. Big Sky, MT, USA: IEEE, 2004, pp. 772–789. ISBN: 978-0-7803-8155-1. DOI: 10.1109/AERO.2004.1367679 (cit. on p. 18).

[108] B. K. P. Horn. "Relative Orientation". In: *International Journal of Computer Vision* 4.1 (Jan. 1990), pp. 59–78. ISSN: 1573-1405. DOI: 10.1007/BF00137443 (cit. on p. 95).

[109] G. Hu, K. Khosoussi, and Shoudong Huang. "Towards a Reliable SLAM Back-End". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Tokyo: IEEE, Nov. 2013, pp. 37–43. ISBN: 978-1-4673-6358-7 978-1-4673-6357-0. DOI: 10.1109/IROS.2013.6696329 (cit. on p. 142).

[110] T. Huang and O. Faugeras. "Some Properties of the E Matrix in Two-View Motion Estimation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11.12 (Dec./1989), pp. 1310–1312. ISSN: 01628828. DOI: 10.1109/34.41368 (cit. on p. 95).

[111] M. Humenberger, Y. Cabon, N. Guerin, J. Morat, J. Revaud, P. Rerole, N. Pion, C. de Souza, V. Leroy, and G. Csurka. "Robust Image Retrieval-based Visual Localization Using Kapture". In: *arXiv:2007.13867 [cs]* (Aug. 2020). arXiv: 2007.13867 [cs] (cit. on p. 20).

[112] J. Iglhaut, C. Cabo, S. Puliti, L. Piermattei, J. O'Connor, and J. Rosette. "Structure from Motion Photogrammetry in Forestry: A Review". In: *Current Forestry Reports* 5.3 (Sept. 2019), pp. 155–168. ISSN: 2198-6436. DOI: 10.1007/s40725-019-00094-3 (cit. on p. 2).

[113] V. Ila, L. Polok, M. Solony, and K. Istenic. "Fast Incremental Bundle Adjustment with Covariance Recovery". In: *2017 International Conference on 3D Vision (3DV)*. Qingdao: IEEE, Oct. 2017, pp. 175–184. ISBN: 978-1-5386-2610-8. DOI: 10.1109/3DV.2017.00029 (cit. on p. 141).

[114] V. Ila, L. Polok, M. Solony, and P. Svoboda. "Highly Efficient Compact Pose SLAM with SLAM++". In: *arXiv:1608.03037 [cs]* (Aug. 2016). arXiv: 1608.03037 [cs] (cit. on p. 141).

[115] V. Indelman. "Navigation Performance Enhancement Using Online Mosaicking". PhD thesis. 2011 (cit. on pp. 26, 125, 126, 128).

[116] V. Indelman. "Bundle Adjustment without Iterative Structure Estimation and Its Application to Navigation". In: *Proceedings of the 2012 IEEE/ION Position, Location and Navigation Symposium*. Myrtle Beach, SC, USA: IEEE, Apr. 2012, pp. 748–756. ISBN: 978-1-4673-0387-3 978-1-4673-0385-9 978-1-4673-0386-6. DOI: 10.1109/PLANS.2012.6236952 (cit. on p. 125).

[117] V. Indelman and F. Dellaert. "Incremental Light Bundle Adjustment: Probabilistic Analysis and Application to Robotic Navigation". In: *New Development in Robot Vision*. Ed. by Y. Sun, A. Behal, and C.-K. R. Chung. Vol. 23. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 111–136. ISBN: 978-3-662-43858-9 978-3-662-43859-6. DOI: 10.1007/978-3-662-43859-6_7 (cit. on pp. 13, 26, 27, 125, 128).

[118] V. Indelman, R. Roberts, C. Beall, and F. Dellaert. "Incremental Light Bundle Adjustment". In: *Procedings of the British Machine Vision Conference 2012*. Surrey: British Machine Vision Association, 2012, pp. 134.1–134.11. ISBN: 978-1-901725-46-9. DOI: 10.5244/C.26.134 (cit. on pp. 26, 27, 125, 133).

[119] V. Indelman, R. Roberts, and F. Dellaert. "Incremental Light Bundle Adjustment for Structure from Motion and Robotics". In: *Robotics and Autonomous Systems* 70 (Aug. 2015), pp. 63–82. ISSN: 09218890. DOI: 10.1016/j.robot.2015.03.009 (cit. on pp. 23, 26, 27, 125, 133).

[120] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof. "From Structure-from-Motion Point Clouds to Fast Location Recognition". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. June 2009, pp. 2599–2606. DOI: 10.1109/CVPR.2009.5206587 (cit. on p. 20).

[121] Y.-D. Jian, D. Balcan, and F. Dellaert. "Generalized Subgraph Preconditioners for Large-Scale Bundle Adjustment". In: *Proceedings / IEEE International Conference on Computer Vision. IEEE International Conference on Computer Vision*. Vol. 7474. Nov. 2011, pp. 295–302. ISBN: 978-3-642-34090-1. DOI: 10.1109/ICCV.2011.6126255 (cit. on p. 140).

[122] N. Jiang, Z. Cui, and P. Tan. "A Global Linear Method for Camera Pose Registration". In: *2013 IEEE International Conference on Computer Vision*. Sydney, Australia: IEEE, Dec. 2013, pp. 481–488. ISBN: 978-1-4799-2840-8. DOI: 10.1109/ICCV.2013.66 (cit. on p. 22).

[123] J. Jiao, L. Yuan, W. Tang, Z. Deng, and Q. Wu. "A Post-Rectification Approach of Depth Images of Kinect v2 for 3D Reconstruction of Indoor Scenes". In: *ISPRS International Journal of Geo-Information* 6.11 (Nov. 2017), p. 349. ISSN: 2220-9964. DOI: 10.3390/ijgi6110349 (cit. on p. 172).

[124] L. F. Julià and P. Monasse. "A Critical Review of the Trifocal Tensor Estimation". In: *Image and Video Technology*. Ed. by M. Paul, C. Hitoshi, and Q. Huang. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 337–349. ISBN: 978-3-319-75786-5. DOI: 10.1007/978-3-319-75786-5_28 (cit. on p. 131).

[125] A. Jurić, F. Kendeš, I. Marković, and I. Petrović. "A Comparison of Graph Optimization Approaches for Pose Estimation in SLAM". In: *International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)* (2021), p. 1 (cit. on pp. 140, 141, 144).

[126] M. Kaess, Kai Ni, and F. Dellaert. "Flow Separation for Fast and Robust Stereo Odometry". In: *2009 IEEE International Conference on Robotics and Automation*. Kobe: IEEE, May 2009, pp. 3539–3544. ISBN: 978-1-4244-2788-8. DOI: 10.1109/ROBOT.2009.5152333 (cit. on pp. 18, 114).

[127] M. Kaess, A. Ranganathan, and F. Dellaert. "iSAM: Incremental Smoothing and Mapping". In: *IEEE Transactions on Robotics* 24.6 (Dec. 2008), pp. 1365–1378. ISSN: 1552-3098, 1941-0468. DOI: 10.1109/TRO.2008.2006706 (cit. on p. 141).

[128] M. Kaess and F. Dellaert. "Covariance Recovery from a Square Root Information Matrix for Data Association". In: *Robotics and Autonomous Systems* 57.12 (Dec. 2009), pp. 1198–1210. ISSN: 09218890. DOI: 10.1016/j.robot.2009.06.008 (cit. on p. 67).

[129] M. Kaess, V. Ila, R. Roberts, and F. Dellaert. "The Bayes Tree: An Algorithmic Foundation for Probabilistic Robot Mapping". In: *Algorithmic Foundations of Robotics IX*. Ed. by B. Siciliano, O. Khatib, F. Groen, D. Hsu, V. Isler, J.-C. Latombe, and M. C. Lin. Vol. 68. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 157–173. ISBN: 978-3-642-17451-3 978-3-642-17452-0. DOI: 10.1007/978-3-642-17452-0_10 (cit. on pp. 13, 141).

[130] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert. "iSAM2: Incremental Smoothing and Mapping with Fluid Relinearization and Incremental Variable Reordering". In: *2011 IEEE International Conference on Robotics and Automation*. Shanghai, China: IEEE, May 2011, pp. 3281–3288. ISBN: 978-1-61284-386-5. DOI: 10.1109/ICRA.2011.5979641 (cit. on p. 141).

[131] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert. "iSAM2: Incremental Smoothing and Mapping Using the Bayes Tree". In: *The International Journal of Robotics Research* 31.2 (Feb. 2012), pp. 216–235. ISSN: 0278-3649, 1741-3176. DOI: 10.1177/0278364911430419 (cit. on pp. 15, 141).

[132] M. Kaess, A. Ranganathan, and F. Dellaert. "Fast Incremental Square Root Information Smoothing". In: *In IEEE International Conference on Robotics and Automation*. 2007 (cit. on pp. 15, 141).

[133] A. Kagan and Z. Landsman. "Relation between the Covariance and Fisher Information Matrices". In: *Statistics & Probability Letters* 42.1 (Mar. 1999), pp. 7–13. ISSN: 01677152. DOI: 10.1016/S0167-7152(98)00178-3 (cit. on p. 67).

[134] L. Kang, L. Wu, and Y.-H. Yang. "Robust Multi-View L2 Triangulation via Optimal Inlier Selection and 3D Structure Refinement". In: *Pattern Recognition* 47.9 (Sept. 2014), pp. 2974–2992. ISSN: 00313203. DOI: 10.1016/j.patcog.2014.03.022 (cit. on p. 22).

[135] J. Kannala and S. S. Brandt. "A Generic Camera Model and Calibration Method for Conventional, Wide-Angle, and Fish-Eye Lenses". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.8 (Aug. 2006), pp. 1335–1340. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2006.153 (cit. on p. 37).

[136] A. Kasyanov, F. Engelmann, J. Stückler, and B. Leibe. "Keyframe-Based Visual-Inertial Online SLAM with Relocalization". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (Sept. 2017), pp. 6662–6669. DOI: 10.1109/IROS.2017.8206581. arXiv: 1702.02175 (cit. on p. 16).

[137] T. Kazik, L. Kneip, J. Nikolic, M. Pollefeys, and R. Siegwart. "Real-Time 6D Stereo Visual Odometry with Non-Overlapping Fields of View". In: IEEE, June 2012, pp. 1529–1536. ISBN: 978-1-4673-1228-8 978-1-4673-1226-4 978-1-4673-1227-1. DOI: 10.1109/CVPR.2012.6247843 (cit. on p. 18).

[138] B. Khomutenko, G. Garcia, and P. Martinet. "An Enhanced Unified Camera Model". In: *IEEE Robotics and Automation Letters* 1.1 (Jan. 2016), pp. 137–144. ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2015.2502921 (cit. on p. 38).

[139] B. Kitt, J. Rehder, A. Chambers, M. Schönbein, H. Lategahn, and S. Singh. "Monocular Visual Odometry Using a Planar Road Model to Solve Scale Ambiguity". In: *ECMR*. 2011 (cit. on p. 18).

[140] B. Kitt, A. Geiger, and H. Lategahn. "Visual Odometry Based on Stereo Image Sequences with RANSAC-based Outlier Rejection Scheme". In: *2010 IEEE Intelligent Vehicles Symposium*. La Jolla, CA, USA: IEEE, June 2010, pp. 486–492. ISBN: 978-1-4244-7866-8. DOI: 10.1109/IVS.2010.5548123 (cit. on p. 18).

[141] B. Klingner, D. Martin, and J. Roseborough. "Street View Motion-from-Structure-from-Motion". In: IEEE, Dec. 2013, pp. 953–960. ISBN: 978-1-4799-2840-8. DOI: 10.1109/ICCV.2013.122 (cit. on pp. 20, 22).

[142] L. Kneip, R. Siegwart, and M. Pollefeys. "Finding the Exact Rotation between Two Images Independently of the Translation". In: *Computer Vision – ECCV 2012*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid. Vol. 7577. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 696–709. ISBN: 978-3-642-33782-6 978-3-642-33783-3. DOI: 10.1007/978-3-642-33783-3_50 (cit. on p. 107).

[143] J. Kopf, M. F. Cohen, and R. Szeliski. "First-Person Hyper-Lapse Videos". In: *ACM Transactions on Graphics* 33.4 (July 2014), pp. 1–10. ISSN: 07300301. DOI: 10.1145/2601097.2601195 (cit. on p. 20).

[144] J. Kopf, M. Uyttendaele, O. Deussen, and M. F. Cohen. "Capturing and Viewing Gigapixel Images". In: *ACM Transactions on Graphics* 26.3 (July 2007), 93–es. ISSN: 0730-0301. DOI: 10.1145/1276377.1276494 (cit. on p. 75).

[145] Ľ. Kovanič, P. Blišťan, V. Zelizňaková, and J. Palková. "Surveying of Open Pit Mine Using Low-Cost Aerial Photogrammetry". In: *The Rise of Big Spatial Data*. Ed. by I. Ivan, A. Singleton, J. Horák, and T. Inspektor. Cham: Springer International Publishing, 2017, pp. 121–129. ISBN: 978-3-319-45122-0 978-3-319-45123-7. DOI: 10.1007/978-3-319-45123-7_9 (cit. on p. 2).

[146] T. Kroeger and L. Gool. "Video Registration to SfM Models". In: *ECCV*. 2014. DOI: 10.1007/978-3-319-10602-1_1 (cit. on pp. 20, 38).

[147] B. Krolla, C. Gava, A. Pagani, and D. Stricker. "Consistent Pose Uncertainty Estimation for Spherical Cameras". In: June 2014 (cit. on pp. 61, 66).

[148] E. Kruppa. "Zur Ermittlung eines Objektes aus zwei Perspektiven mit innerer Orientierung". In: *Sitzungsberichte der Mathematisch-Naturwissenschaftlichen Kaiserlichen Akademie der Wissenschaften, Abteilung II a.* Vol. 122. 1913, pp. 1939–1948 (cit. on p. 94).

[149] F. R. Kschischang and H.-A. Loeliger. "Factor Graphs and the Sum-Product Algorithm". In: *IEEE TRANSACTIONS ON INFORMATION THEORY* 47.2 (2001), p. 22 (cit. on pp. 13, 15).

[150] Z. Kukelova, M. Bujnak, and T. Pajdla. "Polynomial Eigenvalue Solutions to the 5-Pt and 6-Pt Relative Pose Problems". In: *Procedings of the British Machine Vision Conference 2008*. Leeds: British Machine Vision Association, 2008, pp. 56.1–56.10. ISBN: 978-1-901725-36-0. DOI: 10.5244/C.22.56 (cit. on pp. 101, 102).

[151] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. "G2o: A General Framework for Graph Optimization". In: *2011 IEEE International Conference on Robotics and Automation*. Shanghai, China: IEEE, May 2011, pp. 3607–3613. ISBN: 978-1-61284-386-5. DOI: 10.1109/ICRA.2011.5979949 (cit. on p. 141).

[152] Y. Latif, C. Cadena, and J. Neira. "Robust Loop Closing Over Time". In: July 2012. DOI: 10.15607/RSS.2012.VIII.030 (cit. on p. 144).

[153] Y. Latif, C. Cadena, and J. Neira. "Robust Loop Closing over Time for Pose Graph SLAM". In: *The International Journal of Robotics Research* 32.14 (Dec. 2013), pp. 1611–1626. ISSN: 0278-3649, 1741-3176. DOI: 10.1177/0278364913498910 (cit. on p. 144).

[154] Y. Latif, C. Cadena, and J. Neira. "Robust Graph SLAM Back-Ends: A Comparative Analysis". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Chicago, IL, USA: IEEE, Sept. 2014, pp. 2683–2690. ISBN: 978-1-4799-6934-0 978-1-4799-6931-9. DOI: 10.1109/IROS.2014.6942929 (cit. on p. 145).

[155] S. Laveau and O. Faugeras. "Oriented Projective Geometry for Computer Vision". In: *Computer Vision — ECCV '96*. Ed. by G. Goos, J. Hartmanis, J. van Leeuwen, B. Buxton, and R. Cipolla. Vol. 1064. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, pp. 147–156. ISBN: 978-3-540-61122-6 978-3-540-49949-7. DOI: 10.1007/BFb0015531 (cit. on p. 97).

[156] S. H. Lee and J. Civera. "Closed-Form Optimal Two-View Triangulation Based on Angular Errors". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, Oct. 2019, pp. 2681–2689. ISBN: 978-1-72814-803-8. DOI: 10.1109/ICCV.2019.00277 (cit. on pp. 22, 89, 93).

[157] S. H. Lee and J. Civera. "Triangulation: Why Optimize?" In: *arXiv:1907.11917 [cs]* (Aug. 2019). arXiv: 1907.11917 [cs] (cit. on pp. 22, 89, 92–94, 110).

[158] V. Lepetit, F. Moreno-Noguer, and P. Fua. "EPnP: An Accurate O(n) Solution to the PnP Problem". In: *International Journal of Computer Vision* 81.2 (Feb. 2009), pp. 155–166. ISSN: 0920-5691, 1573-1405. DOI: 10.1007/s11263-008-0152-6 (cit. on p. 63).

[159] S. Leutenegger, M. Chli, and R. Y. Siegwart. "BRISK: Binary Robust Invariant Scalable Keypoints". In: *2011 International Conference on Computer Vision*. Nov. 2011, pp. 2548–2555. DOI: 10.1109/ICCV.2011.6126542 (cit. on p. 161).

[160] S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, K. Konolige, and R. Siegwart. "Keyframe-Based Visual-Inertial SLAM Using Nonlinear Optimization". In: *Robotics: Science and Systems IX*. Robotics: Science and Systems Foundation, June 2013. ISBN: 978-981-07-3937-9. DOI: 10.15607/RSS.2013.IX.037 (cit. on p. 16).

[161] B. Li, L. Heng, K. Koser, and M. Pollefeys. "A Multiple-Camera System Calibration Toolbox Using a Feature Descriptor-Based Calibration Pattern". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Tokyo: IEEE, Nov. 2013, pp. 1301–1307. ISBN: 978-1-4673-6358-7 978-1-4673-6357-0. DOI: 10.1109/IROS.2013.6696517 (cit. on pp. 51, 53, 55, 59, 71).

[162] H. Li. "A Simple Solution to the Six-Point Two-View Focal-Length Problem". In: *Computer Vision – ECCV 2006*. Ed. by A. Leonardis, H. Bischof, and A. Pinz. Vol. 3954. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 200–213. ISBN: 978-3-540-33838-3 978-3-540-33839-0. DOI: 10.1007/11744085_16 (cit. on pp. 101, 102).

[163] H. Li and R. Hartley. "Five-Point Motion Estimation Made Easy". In: *Proceedings of the 18th International Conference on Pattern Recognition - Volume 01*. ICPR '06. USA: IEEE Computer Society, Aug. 2006, pp. 630–633. ISBN: 978-0-7695-2521-1. DOI: 10.1109/ICPR.2006.579 (cit. on pp. 101, 102).

[164] S. Li, X. Wang, and T. Kosaki. "Computation of Homography between a Spherical Image and a Perspective Image". In: *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*. Munich, Germany: IEEE, Aug. 2018, pp. 439–444. ISBN: 978-1-5386-3593-3. DOI: 10.1109/COASE.2018.8560371 (cit. on p. 121).

[165] Li Ling, Eva Cheng, and I. S. Burnett. "Eight Solutions of the Essential Matrix for Continuous Camera Motion Tracking in Video Augmented Reality". In: *2011 IEEE International Conference on Multimedia and Expo*. Barcelona, Spain: IEEE, July 2011, pp. 1–6. ISBN: 978-1-61284-348-3. DOI: 10.1109/ICME.2011.6011989 (cit. on pp. 97, 98).

[166] P. Lindstrom. "Triangulation Made Easy". In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. San Francisco, CA, USA: IEEE, June 2010, pp. 1554–1561. ISBN: 978-1-4244-6984-0. DOI: 10.1109/CVPR.2010.5539785 (cit. on p. 22).

[167] A. Liu, S. Marschner, and N. Snavely. "Caliber: Camera Localization and Calibration Using Rigidity Constraints". In: *International Journal of Computer Vision* 118.1 (May 2016), pp. 1–21. ISSN: 0920-5691, 1573-1405. DOI: 10.1007/s11263-015-0866-1 (cit. on p. 59).

[168] H. C. Longuet-Higgins. "A Computer Algorithm for Reconstructing a Scene from Two Projections". In: *Nature* 293.5828 (Sept. 1981), pp. 133–135. ISSN: 1476-4687. DOI: 10.1038/293133a0 (cit. on pp. 95, 101, 102).

[169] R. Losch, S. Grehl, M. Donner, C. Buhl, and B. Jung. "Design of an Autonomous Robot for Mapping, Navigation, and Manipulation in Underground Mines". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid: IEEE, Oct. 2018, pp. 1407–1412. ISBN: 978-1-5386-8094-0. DOI: 10.1109/IROS.2018.8594190 (cit. on p. 2).

[170] Y. Lou, N. Snavely, and J. Gehrke. "MatchMiner: Efficient Spanning Structure Mining in Large Image Collections". In: *Computer Vision – ECCV 2012*. Ed. by A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid. Vol. 7573. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 45–58. ISBN: 978-3-642-33708-6 978-3-642-33709-3. DOI: 10.1007/978-3-642-33709-3_4 (cit. on p. 21).

[171] M. Lourakis and A. Argyros. "Is Levenberg-Marquardt the Most Efficient Optimization Algorithm for Implementing Bundle Adjustment?." In: *Proceedings*

of the Tenth IEEE International Conference on Computer Vision. Vol. 2. Jan. 2005, pp. 1526–1531. DOI: 10.1109/ICCV.2005.128 (cit. on p. 24).

[172] M. I. A. Lourakis and A. A. Argyros. *The Design and Implementation of a Generic Sparse Bundle Adjustment Software Package Based on the Levenberg-Marquardt Algorithm*. Tech. rep. 2004 (cit. on pp. 23, 159).

[173] M. I. A. Lourakis and A. A. Argyros. "SBA: A Software Package for Generic Sparse Bundle Adjustment". In: *ACM Transactions on Mathematical Software* 36.1 (Mar. 2009), pp. 1–30. ISSN: 00983500. DOI: 10.1145/1486525.1486527 (cit. on pp. 23, 159).

[174] D. Lowe. "Object Recognition from Local Scale-Invariant Features". In: IEEE, 1999, 1150–1157 vol.2. ISBN: 978-0-7695-0164-2. DOI: 10.1109/ICCV.1999.790410 (cit. on p. 161).

[175] F. Lu and E. Milios. "Globally Consistent Range Scan Alignment for Environment Mapping". In: *Autonomous Robots* 4.4 (Oct. 1997), pp. 333–349. ISSN: 1573-7527. DOI: 10.1023/A:1008854305733 (cit. on p. 13).

[176] Y. Ma, K. Huang, R. Vidal, J. Košecká, and S. Sastry. "Rank Conditions on the Multiple-View Matrix". In: *International Journal of Computer Vision* 59.2 (Sept. 2004), pp. 115–137. ISSN: 0920-5691. DOI: 10.1023/B:VISI.0000022286.53224.3d (cit. on pp. 95, 128, 131).

[177] Y. Ma, S. Soatto, J. Košecká, and S. S. Sastry. *An Invitation to 3-D Vision*. Ed. by S. S. Antman, J. E. Marsden, L. Sirovich, and S. Wiggins. Vol. 26. Interdisciplinary Applied Mathematics. New York, NY: Springer New York, 2004. ISBN: 978-1-4419-1846-8 978-0-387-21779-6. DOI: 10.1007/978-0-387-21779-6 (cit. on pp. 95, 128, 131, 184).

[178] M. T. Malinowski, A. Richards, and M. Woods. "Fusion of Visual and Wheel Odometry with Integrated Slip Estimation". In: *AIAA Scitech 2021 Forum*. AIAA SciTech Forum. American Institute of Aeronautics and Astronautics, Jan. 2021. DOI: 10.2514/6.2021-1757 (cit. on p. 18).

[179] K. Matzen, M. F. Cohen, B. Evans, J. Kopf, and R. Szeliski. "Low-Cost 360 Stereo Photography and Video Capture". In: *ACM Transactions on Graphics* 36.4 (July 2017), pp. 1–12. ISSN: 07300301. DOI: 10.1145/3072959.3073645 (cit. on p. 77).

[180] H. Mayer. "RPBA – Robust Parallel Bundle Adjustment Based on Covariance Information". In: *arXiv:1910.08138 [cs]* (Oct. 2019). arXiv: 1910.08138 [cs] (cit. on p. 23).

[181] C. Mei and P. Rives. "Single View Point Omnidirectional Camera Calibration from Planar Grids". In: *Proceedings 2007 IEEE International Conference on Robotics and Automation*. Rome, Italy: IEEE, Apr. 2007, pp. 3945–3950. ISBN: 978-1-4244-0602-9 978-1-4244-0601-2. DOI: 10.1109/ROBOT.2007.364084 (cit. on pp. 35, 51, 55).

[182] B. Micusik and T. Pajdla. "Estimation of Omnidirectional Camera Model from Epipolar Geometry". In: *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.* Madison, WI, USA: IEEE Comput. Soc, 2003, pp. I–485–I–490. ISBN: 978-0-7695-1900-5. DOI: 10.1109/CVPR.2003.1211393 (cit. on p. 39).

[183] B. Micusik and T. Pajdla. "Structure from Motion with Wide Circular Field of View Cameras". In: *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE* 28.7 (2006), p. 15 (cit. on p. 22).

[184] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. "FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem". In: *In Proceedings of the AAAI National Conference on Artificial Intelligence*. AAAI, 2002, pp. 593–598 (cit. on p. 12).

[185] M. Montemerlo, S. Thrun, D. Roller, and B. Wegbreit. "FastSLAM 2.0: An Improved Particle Filtering Algorithm for Simultaneous Localization and Mapping That Provably Converges". In: *Proceedings of the 18th International Joint Conference on Artificial Intelligence*. IJCAI'03. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., Aug. 2003, pp. 1151–1156 (cit. on p. 12).

[186] F. Moreno-Noguer, V. Lepetit, and P. Fua. "Accurate Non-Iterative O(n) Solution to the PnP Problem". In: *2007 IEEE 11th International Conference on Computer Vision*. Rio de Janeiro, Brazil: IEEE, 2007, pp. 1–8. ISBN: 978-1-4244-1630-1. DOI: 10.1109/ICCV.2007.4409116 (cit. on p. 63).

[187] P. Moulon, P. Monasse, and R. Marlet. "Global Fusion of Relative Motions for Robust, Accurate and Scalable Structure from Motion". In: *Proceedings of the IEEE International Conference on Computer Vision*. Dec. 2013. DOI: 10.1109/ICCV.2013.403 (cit. on p. 22).

[188] P. Moulon, P. Monasse, R. Perrot, and R. Marlet. "OpenMVG: Open Multiple View Geometry". In: *Reproducible Research in Pattern Recognition*. Ed. by B. Kerautret, M. Colom, and P. Monasse. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2017, pp. 60–74. ISBN: 978-3-319-56414-2. DOI: 10.1007/978-3-319-56414-2_5 (cit. on p. 20).

[189] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. "ORB-SLAM: A Versatile and Accurate Monocular SLAM System". In: *IEEE Transactions on Robotics* 31.5 (Oct. 2015), pp. 1147–1163. ISSN: 1552-3098, 1941-0468. DOI: 10.1109/TRO.2015.2463671. arXiv: 1502.00956 (cit. on p. 119).

[190] R. Mur-Artal and J. D. Tardos. "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras". In: *IEEE Transactions on Robotics* 33.5 (Oct. 2017), pp. 1255–1262. ISSN: 1552-3098, 1941-0468. DOI: 10.1109/TRO.2017.2705103. arXiv: 1610.06475 (cit. on pp. 16, 119, 162).

[191] D. Nister. "An Efficient Solution to the Five-Point Relative Pose Problem". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26.6 (June 2004), pp. 756–770. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2004.17 (cit. on pp. 101, 102, 105).

[192] D. Nister, O. Naroditsky, and J. Bergen. "Visual Odometry". In: *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.* Vol. 1. June 2004, pp. I–I. DOI: 10.1109/CVPR.2004.1315094 (cit. on p. 17).

[193] G. Nützi, S. Weiss, D. Scaramuzza, and R. Siegwart. "Fusion of IMU and Vision for Absolute Scale Estimation in Monocular SLAM". In: *Journal of Intelligent and Robotic Systems* 61 (Mar. 2011), pp. 287–299. ISSN: 978-94-007-1109-9. DOI: 10.1007/s10846-010-9490-z (cit. on p. 18).

[194] O. Ozyesil, V. Voroninski, R. Basri, and A. Singer. "A Survey of Structure from Motion". In: *arXiv:1701.08493 [cs]* (May 2017). arXiv: 1701.08493 [cs] (cit. on p. 20).

[195] A. Pagani, C. Gava, Y. Cui, B. Krolla, J.-M. Hengen, and D. Stricker. *Dense 3D Point Cloud Generation from Multiple High-resolution Spherical Images.* The Eurographics Association, 2011. ISBN: 978-3-905674-34-7. DOI: 10.2312/VAST/VAST11/017-024 (cit. on pp. 96, 108).

[196] A. Pagani and D. Stricker. "Structure from Motion Using Full Spherical Panoramic Cameras". In: IEEE, Nov. 2011, pp. 375–382. ISBN: 978-1-4673-0063-6 978-1-4673-0062-9 978-1-4673-0061-2. DOI: 10.1109/ICCVW.2011.6130266 (cit. on pp. 22, 61, 62, 96, 104, 108, 109).

[197] A. Pandey and U. C. Pati. "Image Mosaicing: A Deeper Insight". In: *Image and Vision Computing* 89 (Sept. 2019), pp. 236–257. ISSN: 02628856. DOI: 10.1016/j.imavis.2019.07.002 (cit. on p. 76).

[198] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988. ISBN: 978-1-55860-479-7 (cit. on p. 13).

[199] S. Peleg, M. Ben-Ezra, and Y. Pritch. "Omnistereo: Panoramic Stereo Imaging". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23.3 (Mar. 2001), pp. 279–290. ISSN: 01628828. DOI: 10.1109/34.910880 (cit. on p. 77).

[200] J. Philip. "A Non-Iterative Algorithm for Determining All Essential Matrices Corresponding to Five Point Pairs". In: *The Photogrammetric Record* 15.88 (Oct. 1996), pp. 589–599. ISSN: 0031-868X, 1477-9730. DOI: 10.1111/0031-868X.00066 (cit. on p. 101).

[201] J. Philip. *Critical Point Configurations of the 5-, 6-, 7-, and 8-Point Algorithms for Relative Orientation*. Department of Mathematics, Royal Institute of Technology, 1998 (cit. on pp. 105, 106).

[202] B. Ponitz, M. Sastuba, and C. Brücker. "4D Visualization Study of a Vortex Ring Life Cycle Using Modal Analyses". In: *Journal of Visualization* 19.2 (May 2016), pp. 237–259. ISSN: 1343-8875, 1875-8975. DOI: 10.1007/s12650-015-0314-x (cit. on p. 34).

[203] B. Ponitz, M. Sastuba, C. Brücker, and J. Kitzhofer. "Volumetric Velocimetry via Scanning Back-Projection and Least-Squares-Matching Algorithms of a Vortex Ring". In: *16th Int. Symp. on Appl. of Laser Techniques to Fluid Mechanics* (2012), pp. 9–12 (cit. on p. 34).

[204] C. Richardt, Y. Pritch, H. Zimmer, and A. Sorkine-Hornung. "Megastereo: Constructing High-Resolution Stereo Panoramas". In: IEEE, June 2013, pp. 1256–1263. ISBN: 978-0-7695-4989-7. DOI: 10.1109/CVPR.2013.166 (cit. on p. 77).

[205] V. Rodehorst. "EVALUATION OF THE METRIC TRIFOCAL TENSOR FOR RELATIVE THREE-VIEW ORIENTATION". In: *20th International Conference on the Application of Computer Science and Mathematics in Architecture and Civil Engineering*. Weimar, Germany, July 2015, p. 7 (cit. on p. 131).

[206] V. Rodehorst, M. Heinrichs, and O. Hellwich. "Evaluation of Relative Pose Estimation Methods for Multi-Camera Setups". In: *International Archives of Photogrammetry and Remote Sensing (ISPRS '08)* (Jan. 2008) (cit. on pp. 104, 108).

[207] A. L. Rodríguez, P. E. López-de-Teruel, and A. Ruiz. "Reduced Epipolar Cost for Accelerated Incremental SfM". In: *CVPR 2011*. June 2011, pp. 3097–3104. DOI: 10.1109/CVPR.2011.5995569 (cit. on pp. 24, 26).

[208] A. Rodríguez, P. López-de-Teruel, and A. Ruiz. "GEA Optimization for Live Structureless Motion Estimation". In: *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*. Nov. 2011, pp. 715–718. DOI: 10.1109/ICCVW.2011.6130320 (cit. on p. 26).

[209] D. M. Rosen, L. Carlone, A. S. Bandeira, and J. J. Leonard. "SE-Sync: A Certifiably Correct Algorithm for Synchronization over the Special Euclidean Group". In: *arXiv:1612.07386 [cs]* (Feb. 2017). arXiv: 1612.07386 [cs] (cit. on p. 143).

[210] D. M. Rosen, L. Carlone, A. S. Bandeira, and J. J. Leonard. "SE-Sync: A Certifiably Correct Algorithm for Synchronization over the Special Euclidean Group". In: *The International Journal of Robotics Research* 38.2-3 (Mar. 2019), pp. 95–125. ISSN: 0278-3649, 1741-3176. DOI: 10.1177/0278364918784361 (cit. on p. 143).

[211] A. Rosinol, M. Abate, Y. Chang, and L. Carlone. "Kimera: An Open-Source Library for Real-Time Metric-Semantic Localization and Mapping". In: *arXiv:1910.02490 [cs]* (Mar. 2020). arXiv: 1910.02490 [cs] (cit. on pp. 16, 143).

[212] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. "ORB: An Efficient Alternative to SIFT or SURF". In: *2011 International Conference on Computer Vision*. Barcelona, Spain: IEEE, Nov. 2011, pp. 2564–2571. ISBN: 978-1-4577-1102-2 978-1-4577-1101-5 978-1-4577-1100-8. DOI: 10.1109/ICCV.2011.6126544 (cit. on p. 161).

[213] M. Rufli, D. Scaramuzza, and R. Siegwart. "Automatic Detection of Checkerboards on Blurred and Distorted Images". In: *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Nice: IEEE, Sept. 2008, pp. 3121–3126. ISBN: 978-1-4244-2057-5 978-1-4244-2058-2. DOI: 10.1109/IROS.2008.4650703 (cit. on p. 53).

[214] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison. "SLAM++: Simultaneous Localisation and Mapping at the Level of Objects". In: *2013 IEEE Conference on Computer Vision and Pattern Recognition*. Portland, OR, USA: IEEE, June 2013, pp. 1352–1359. ISBN: 978-0-7695-4989-7. DOI: 10.1109/CVPR.2013.178 (cit. on p. 142).

[215] D. Scaramuzza. "Omnidirectional Vision: From Calibration to Root Motion Estimation". PhD thesis. 2007 (cit. on p. 39).

[216] D. Scaramuzza, A. Martinelli, and R. Siegwart. "A Toolbox for Easily Calibrating Omnidirectional Cameras". In: *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Oct. 2006, pp. 5695–5701. DOI: 10.1109/IROS.2006.282372 (cit. on pp. 37, 39, 53).

[217] D. Scaramuzza and R. Siegwart. "Appearance-Guided Monocular Omnidirectional Visual Odometry for Outdoor Ground Vehicles". In: *IEEE Transactions on Robotics* 24.5 (Oct. 2008), pp. 1015–1026. ISSN: 1552-3098, 1941-0468. DOI: 10.1109/TRO.2008.2004490 (cit. on p. 120).

[218] D. Scaramuzza and F. Fraundorfer. "Visual Odometry Part I: The First 30 Years and Fundamentals". In: *IEEE Robotics & Automation Magazine* 18.4 (Dec. 2011), pp. 80–92. ISSN: 1070-9932. DOI: 10.1109/MRA.2011.943233 (cit. on pp. 17, 18, 25, 31, 107, 114, 135).

[219] D. Scaramuzza, A. Martinelli, and R. Siegwart. "A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion". In: *Proceedings of the Fourth IEEE International Conference on Computer Vision Systems*. ICVS '06. USA: IEEE Computer Society, Jan. 2006, p. 45. ISBN: 978-0-7695-2506-8. DOI: 10.1109/ICVS.2006.3 (cit. on p. 39).

[220] D. Scaramuzza and R. Siegwart. "A Practical Toolbox for Calibrating Omnidirectional Cameras". In: *Vision Systems: Applications*. Ed. by G. Obinata and A. Dutt. I-Tech Education and Publishing, June 2007. ISBN: 978-3-902613-01-1. DOI: 10.5772/4994 (cit. on pp. 39, 51, 53, 55).

[221] J. Schneider, F. Schindler, T. Läbe, and W. Förstner. "Bundle Adjustment for Multi-Camera Systems with Points at Infinity". In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* I-3 (July 2012), pp. 75–80. ISSN: 2194-9050. DOI: 10.5194/isprsannals-I-3-75-2012 (cit. on p. 22).

[222] M. Schönbein, T. Strauß, and A. Geiger. "Calibrating and Centering Quasi-Central Catadioptric Cameras". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. May 2014, pp. 4443–4450. DOI: 10.1109/ICRA.2014.6907507 (cit. on pp. 51, 58).

[223] J. L. Schonberger, A. C. Berg, and J.-M. Frahm. "PAIGE: PAirwise Image Geometry Encoding for Improved Efficiency in Structure-from-Motion". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA, USA: IEEE, June 2015, pp. 1009–1018. ISBN: 978-1-4673-6964-0. DOI: 10.1109/CVPR.2015.7298703 (cit. on p. 21).

[224] J. L. Schönberger and J.-M. Frahm. "Structure-from-Motion Revisited". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, June 2016, pp. 4104–4113. ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.445 (cit. on pp. 20, 22, 122).

[225] J. L. Schönberger, E. Zheng, J.-M. Frahm, and M. Pollefeys. "Pixelwise View Selection for Unstructured Multi-View Stereo". In: *Computer Vision – ECCV 2016*. Ed. by B. Leibe, J. Matas, N. Sebe, and M. Welling. Vol. 9907. Cham: Springer International Publishing, 2016, pp. 501–518. ISBN: 978-3-319-46486-2 978-3-319-46487-9. DOI: 10.1007/978-3-319-46487-9_31 (cit. on pp. 20, 21).

[226] T. Schöps, V. Larsson, M. Pollefeys, and T. Sattler. "Why Having 10,000 Parameters in Your Camera Model Is Better Than Twelve". In: *arXiv:1912.02908 [cs]* (June 2020). arXiv: 1912.02908 [cs] (cit. on p. 43).

[227] H. Seok and J. Lim. "ROVO: Robust Omnidirectional Visual Odometry for Wide-baseline Wide-FOV Camera Systems". In: *arXiv:1902.11154 [cs]* (Mar. 2019). arXiv: 1902.11154 [cs] (cit. on p. 18).

[228] M. Servières, V. Renaudin, A. Dupuis, and N. Antigny. "Visual and Visual-Inertial SLAM: State of the Art, Classification, and Experimental Benchmarking". In: *Journal of Sensors* 2021 (Feb. 2021). Ed. by S. M. Potirakis, pp. 1–26. ISSN: 1687-7268, 1687-725X. DOI: 10.1155/2021/2054828 (cit. on p. 16).

[229] A. Shashua. "Algebraic Functions for Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17.8 (Aug. 1995), pp. 779–789. ISSN: 1939-3539. DOI: 10.1109/34.400567 (cit. on pp. 129, 131).

[230] A. Shashua and M. Werman. "Trilinearity of Three Perspective Views and Its Associated Tensor". In: *Proceedings of IEEE International Conference on Computer Vision*. Cambridge, MA, USA: IEEE Comput. Soc. Press, 1995, pp. 920–925. ISBN: 978-0-8186-7042-8. DOI: 10.1109/ICCV.1995.466837 (cit. on p. 131).

[231] A. Shashua. "Trilinearity in Visual Recognition by Alignment". In: *Computer Vision — ECCV '94*. Ed. by J.-O. Eklundh. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 1994, pp. 479–484. ISBN: 978-3-540-48398-4. DOI: 10.1007/3-540-57956-7_53 (cit. on p. 131).

[232] S. N. Sinha, D. Steedly, and R. Szeliski. "A Multi-stage Linear Approach to Structure from Motion". In: *Trends and Topics in Computer Vision*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos,

D. Tygar, M. Y. Vardi, G. Weikum, and K. N. Kutulakos. Vol. 6554. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 267–281. ISBN: 978-3-642-35739-8 978-3-642-35740-4. DOI: 10.1007/978-3-642-35740-4_21 (cit. on p. 22).

[233] R. Smith, M. Self, and P. Cheeseman. "Estimating Uncertain Spatial Relationships in Robotics". In: *Proceedings. 1987 IEEE International Conference on Robotics and Automation*. Vol. 4. Raleigh, NC, USA: Institute of Electrical and Electronics Engineers, 1987, pp. 850–850. DOI: 10.1109/ROBOT.1987.1087846 (cit. on p. 12).

[234] N. Snavely, S. M. Seitz, and R. Szeliski. "Photo Tourism: Exploring Photo Collections in 3D". In: ACM Press, 2006, p. 835. ISBN: 978-1-59593-364-5. DOI: 10.1145/1179352.1141964 (cit. on pp. 13, 19–21).

[235] N. Snavely, S. M. Seitz, and R. Szeliski. "Modeling the World from Internet Photo Collections". In: *International Journal of Computer Vision* 80.2 (Nov. 2008), pp. 189–210. ISSN: 0920-5691, 1573-1405. DOI: 10.1007/s11263-007-0107-3 (cit. on pp. 19, 20).

[236] N. Snavely, S. M. Seitz, and R. Szeliski. "Skeletal Graphs for Efficient Structure from Motion". In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*. Anchorage, AK, USA: IEEE, June 2008, pp. 1–8. ISBN: 978-1-4244-2242-5. DOI: 10.1109/CVPR.2008.4587678 (cit. on p. 21).

[237] R. Steffen, J.-M. Frahm, and W. Förstner. "Relative Bundle Adjustment Based on Trifocal Constraints". In: *Trends and Topics in Computer Vision*. Ed. by D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, and K. N. Kutulakos. Vol. 6554. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 282–295. ISBN: 978-3-642-35739-8 978-3-642-35740-4. DOI: 10.1007/978-3-642-35740-4_22 (cit. on p. 26).

[238] H. Stewenius, F. Schaffalitzky, and D. Nister. "How Hard Is 3-View Triangulation Really?" In: *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*. Beijing, China: IEEE, 2005, 686–693 Vol. 1. ISBN: 978-0-7695-2334-7. DOI: 10.1109/ICCV.2005.115 (cit. on p. 22).

[239] H. Stewénius, C. Engels, and D. Nistér. "Recent Developments on Direct Relative Orientation". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 60.4 (June 2006), pp. 284–294. ISSN: 09242716. DOI: 10.1016/j.isprsjprs.2006.03.005 (cit. on pp. 101, 102).

[240] H. Strasdat, J. Montiel, and A. J. Davison. "Visual SLAM: Why Filter?" In: *Image and Vision Computing* 30.2 (Feb. 2012), pp. 65–77. ISSN: 02628856. DOI: 10.1016/j.imavis.2012.02.009 (cit. on p. 16).

[241] P. Sturm, S. Ramalingam, and S. Lodha. "On Calibration, Structure from Motion and Multi-View Geometry for Generic Camera Models". In: *Imaging Beyond the Pinhole Camera*. Ed. by K. Daniilidis and R. Klette. Springer Netherlands, 2006, pp. 87–105. ISBN: 978-1-4020-4893-7. DOI: 10.1007/978-1-4020-4894-4_5 (cit. on p. 91).

[242] B. Suger, G. Diego Tipaldi, L. Spinello, and W. Burgard. "An Approach to Solving Large-Scale SLAM Problems with a Small Memory Footprint". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. May 2014, pp. 3632–3637. DOI: 10.1109/ICRA.2014.6907384 (cit. on p. 140).

[243] S. Sumikura, M. Shibuya, and K. Sakurada. "OpenVSLAM: A Versatile Visual SLAM Framework". In: *Proceedings of the 27th ACM International Conference on Multimedia* (Oct. 2019), pp. 2292–2295. DOI: 10.1145/3343031.3350539. arXiv: 1910.01122 (cit. on pp. 16, 17).

[244] N. Sunderhauf and P. Protzel. "Switchable Constraints for Robust Pose Graph SLAM". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vilamoura-Algarve, Portugal: IEEE, Oct. 2012, pp. 1879–1884. ISBN: 978-1-4673-1736-8 978-1-4673-1737-5 978-1-4673-1735-1. DOI: 10.1109/IROS.2012.6385590 (cit. on p. 144).

[245] N. Sunderhauf and P. Protzel. "Towards a Robust Back-End for Pose Graph SLAM". In: *2012 IEEE International Conference on Robotics and Automation*. St Paul, MN, USA: IEEE, May 2012, pp. 1254–1261. ISBN: 978-1-4673-1405-3 978-1-4673-1403-9 978-1-4673-1578-4 978-1-4673-1404-6. DOI: 10.1109/ICRA.2012.6224709 (cit. on pp. 13, 19, 144).

[246] N. Sunderhauf and P. Protzel. "Switchable Constraints vs. Max-Mixture Models vs. RRR - A Comparison of Three Approaches to Robust Pose Graph SLAM". In: *2013 IEEE International Conference on Robotics and Automation*. Karlsruhe, Germany: IEEE, May 2013, pp. 5198–5203. ISBN: 978-1-4673-5643-5 978-1-4673-5641-1. DOI: 10.1109/ICRA.2013.6631320 (cit. on p. 144).

[247] C. Sweeney, V. Fragoso, T. Höllerer, and M. Turk. "gDLS: A Scalable Solution to the Generalized Pose and Scale Problem". In: *Computer Vision – ECCV 2014*. Ed. by D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars. Vol. 8692. Cham:

Springer International Publishing, 2014, pp. 16–31. ISBN: 978-3-319-10592-5 978-3-319-10593-2. DOI: 10.1007/978-3-319-10593-2_2 (cit. on p. 20).

[248] C. Sweeney, V. Fragoso, T. Hollerer, and M. Turk. "Large Scale SfM with the Distributed Camera Model". In: *arXiv:1607.03949 [cs]* (Nov. 2016). arXiv: 1607.03949 [cs] (cit. on p. 24).

[249] C. Sweeney, L. Kneip, T. Hollerer, and M. Turk. "Computing Similarity Transformations from Only Image Correspondences". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA, USA: IEEE, June 2015, pp. 3305–3313. ISBN: 978-1-4673-6964-0. DOI: 10.1109/CVPR.2015.7298951 (cit. on p. 24).

[250] C. Sweeney, T. Sattler, T. Hollerer, M. Turk, and M. Pollefeys. "Optimizing the Viewing Graph for Structure-from-Motion". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. Santiago, Chile: IEEE, Dec. 2015, pp. 801–809. ISBN: 978-1-4673-8391-2. DOI: 10.1109/ICCV.2015.98 (cit. on p. 26).

[251] C. Sweeney, T. Hollerer, and M. Turk. "Theia: A Fast and Scalable Structure-from-Motion Library". In: *Proceedings of the 23rd ACM International Conference on Multimedia*. Brisbane Australia: ACM, Oct. 2015, pp. 693–696. ISBN: 978-1-4503-3459-4. DOI: 10.1145/2733373.2807405 (cit. on p. 20).

[252] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. Intelligent Robotics and Autonomous Agents. Cambridge, Mass: MIT Press, 2005. ISBN: 978-0-262-20162-9 (cit. on pp. 12, 13, 15, 21).

[253] S. Thrun and M. Montemerlo. "The Graph SLAM Algorithm with Applications to Large-Scale Mapping of Urban Structures". In: *The International Journal of Robotics Research* 25.5-6 (May 2006), pp. 403–429. ISSN: 0278-3649, 1741-3176. DOI: 10.1177/0278364906065387 (cit. on pp. 12, 13).

[254] Y. Tian, K. Khosoussi, D. M. Rosen, and J. P. How. "Distributed Certifiably Correct Pose-Graph Optimization". In: *arXiv:1911.03721 [cs, math]* (May 2021). arXiv: 1911.03721 [cs, math] (cit. on p. 144).

[255] R. Toldo, R. Gherardi, M. Farenzena, and A. Fusiello. "Hierarchical Structure-and-Motion Recovery from Uncalibrated Images". In: *Computer Vision and Image Understanding* 140 (Nov. 2015), pp. 127–143. ISSN: 10773142. DOI: 10.1016/j.cviu.2015.05.011. arXiv: 1506.00395 (cit. on p. 21).

[256] A. Torii and A. Imiya. *Computation of Epipolar Geometry and Trifocal Tensor from Spherical Images*. 2007 (cit. on pp. 96, 128, 129, 131).

[257] A. Torii and A. Imiya. "Two- and Three- View Geometry for Spherical Cameras". In: (Jan. 2005) (cit. on pp. 129, 131).

[258] J. Toth. "Kalibrierung omnidirektionaler Multi-Kamerasysteme". Bachelor Thesis. Freiberg: Technische Universität Bergakadmie Freiberg, June 2017 (cit. on pp. 58, 71, 170, 172).

[259] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. "Bundle Adjustment — A Modern Synthesis". In: *Vision Algorithms: Theory and Practice*. Ed. by G. Goos, J. Hartmanis, J. van Leeuwen, B. Triggs, A. Zisserman, and R. Szeliski. Vol. 1883. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 298–372. ISBN: 978-3-540-67973-8 978-3-540-44480-0. DOI: 10.1007/3-540-44480-7_21 (cit. on p. 23).

[260] R. Tsai. "A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using off-the-Shelf TV Cameras and Lenses". In: *IEEE Journal on Robotics and Automation* 3.4 (Aug. 1987), pp. 323–344. ISSN: 2374-8710. DOI: 10.1109/JRA.1987.1087109 (cit. on p. 54).

[261] R. Y. Tsai and T. S. Huang. "Uniqueness and Estimation of Three-Dimensional Motion Parameters of Rigid Objects with Curved Surfaces". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6.1 (Jan. 1984), pp. 13–27. ISSN: 0162-8828. DOI: 10.1109/TPAMI.1984.4767471 (cit. on p. 95).

[262] S. Umeyama. "Least-Squares Estimation of Transformation Parameters between Two Point Patterns". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13.4 (Apr. 1991), pp. 376–380. ISSN: 1939-3539. DOI: 10.1109/34.88573 (cit. on p. 147).

[263] S. Urban, J. Leitloff, and S. Hinz. "Improved Wide-Angle, Fisheye and Omnidirectional Camera Calibration". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 108 (Oct. 2015), pp. 72–79. ISSN: 09242716. DOI: 10.1016/j.isprsjprs.2015.06.005 (cit. on p. 55).

[264] V. Usenko, N. Demmel, and D. Cremers. "The Double Sphere Camera Model". In: *2018 International Conference on 3D Vision (3DV)*. Verona: IEEE, Sept. 2018, pp. 552–560. ISBN: 978-1-5386-8425-2. DOI: 10.1109/3DV.2018.00069 (cit. on pp. 35, 51).

[265] J. C. Virgolino Soares and M. A. Meggiolaro. "A Pose-Graph Optimization Tool for MATLAB". In: *Anais Do X Congresso Nacional de Engenharia Mecânica*. ABCM, 2018. DOI: 10.26678/ABCM.CONEM2018.CON18-0341 (cit. on p. 142).

[266] H. von Sanden. "Die Bestimmung der Kernpunkte in der Photogrammetrie". PhD thesis. Germany: Göttingen, Dec. 1908 (cit. on p. 94).

[267] X. Wang, R. Marcotte, G. Ferrer, and E. Olson. "ApriISAM: Real-Time Smoothing and Mapping". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. May 2018, pp. 2486–2493. DOI: 10.1109/ICRA.2018.8461072 (cit. on p. 142).

[268] Wei Wang and Hung Tat Tsui. "A SVD Decomposition of Essential Matrix with Eight Solutions for the Relative Positions of Two Perspective Cameras". In: *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*. Vol. 1. Barcelona, Spain: IEEE Comput. Soc, 2000, pp. 362–365. ISBN: 978-0-7695-0750-7. DOI: 10.1109/ICPR.2000.905353 (cit. on pp. 97, 104).

[269] J. Weng, T. Huang, and N. Ahuja. "Motion and Structure from Two Perspective Views: Algorithms, Error Analysis, and Error Estimation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11.5 (May 1989), pp. 451–476. ISSN: 01628828. DOI: 10.1109/34.24779 (cit. on pp. 95, 98, 106, 107).

[270] T. Werner and T. Pajdla. "Cheirality in Epipolar Geometry". In: *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*. Vol. 1. July 2001, 548–553 vol.1. DOI: 10.1109/ICCV.2001.937564 (cit. on p. 97).

[271] A. R. Widya, Y. Monno, K. Imahori, M. Okutomi, S. Suzuki, T. Gotoda, and K. Miki. "3D Reconstruction of Whole Stomach from Endoscope Video Using Structure-from-Motion". In: *arXiv:1905.12988 [cs, eess]* (May 2019). arXiv: 1905.12988 [cs, eess] (cit. on p. 2).

[272] D. Wilbers, C. Merfels, and C. Stachniss. "A Comparison of Particle Filter and Graph-Based Optimization for Localization with Landmarks in Automated Vehicles". In: *2019 Third IEEE International Conference on Robotic Computing (IRC)*. Naples, Italy: IEEE, Feb. 2019, pp. 220–225. ISBN: 978-1-5386-9245-5. DOI: 10.1109/IRC.2019.00040 (cit. on p. 13).

[273] K. Wilson and N. Snavely. "Robust Global Translations with 1DSfM". In: *Computer Vision – ECCV 2014*. Ed. by D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars. Vol. 8691. Cham: Springer International Publishing, 2014, pp. 61–75. ISBN: 978-3-319-10577-2 978-3-319-10578-9. DOI: 10.1007/978-3-319-10578-9_5 (cit. on p. 22).

[274] C. Wu. "Towards Linear-Time Incremental Structure from Motion". In: *2013 International Conference on 3D Vision - 3DV 2013*. June 2013, pp. 127–134. DOI: 10.1109/3DV.2013.25 (cit. on pp. 20, 23).

[275] C. Wu. "Towards Linear-Time Incremental Structure from Motion". In: *2013 International Conference on 3D Vision - 3DV 2013*. June 2013, pp. 127–134. DOI: 10.1109/3DV.2013.25 (cit. on p. 21).

[276] C. Wu, S. Agarwal, B. Curless, and S. M. Seitz. "Multicore Bundle Adjustment". In: *CVPR 2011*. June 2011, pp. 3057–3064. DOI: 10.1109/CVPR.2011.5995552 (cit. on pp. 20, 23).

[277] K. Yang, W. Fang, Y. Zhao, and N. Deng. "Iteratively Reweighted Midpoint Method for Fast Multiple View Triangulation". In: *IEEE Robotics and Automation Letters* 4.2 (Apr. 2019), pp. 708–715. ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2019.2893022 (cit. on pp. 22, 91).

[278] J. S. Yedidia, W. T. Freeman, and Y. Weiss. "Generalized Belief Propagation". In: *Proceedings of the 13th International Conference on Neural Information Processing Systems*. NIPS'00. Cambridge, MA, USA: MIT Press, Jan. 2000, pp. 668–674 (cit. on p. 13).

[279] X. Ying and Z. Hu. "Can We Consider Central Catadioptric Cameras and Fisheye Cameras within a Unified Imaging Model". In: *Computer Vision - ECCV 2004*. Ed. by T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, T. Pajdla, and J. Matas. Vol. 3021. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 442–455. ISBN: 978-3-540-21984-2 978-3-540-24670-1. DOI: 10.1007/978-3-540-24670-1_34 (cit. on p. 35).

[280] K. Yousif, A. Bab-Hadiashar, and R. Hoseinnezhad. "An Overview to Visual Odometry and Visual SLAM: Applications to Mobile Robotics". In: *Intelligent Industrial Systems* 1.4 (Dec. 2015), pp. 289–311. ISSN: 2199-854X. DOI: 10.1007/s40903-015-0032-7 (cit. on p. 18).

[281] F. Zhang and F. Liu. "Parallax-Tolerant Image Stitching". In: IEEE, June 2014, pp. 3262–3269. ISBN: 978-1-4799-5118-5. DOI: 10.1109/CVPR.2014.423 (cit. on p. 76).

[282] Z. Zhang. "A Flexible New Technique for Camera Calibration". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.11 (Nov. 2000), pp. 1330–1334. ISSN: 1939-3539. DOI: 10.1109/34.888718 (cit. on p. 54).

[283] Z. Zhang. "Flexible Camera Calibration by Viewing a Plane from Unknown Orientations". In: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Kerkyra, Greece: IEEE, 1999, 666–673 vol.1. ISBN: 978-0-7695-0164-2. DOI: 10.1109/ICCV.1999.791289 (cit. on p. 54).

[284] Z. Zhang and D. Scaramuzza. "A Tutorial on Quantitative Trajectory Evaluation for Visual(-Inertial) Odometry". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid: IEEE, Oct. 2018, pp. 7244–7251. ISBN: 978-1-5386-8094-0. DOI: 10.1109/IROS.2018.8593941 (cit. on pp. 147, 149).

[285] J. Zhao. "An Efficient Solution to Non-Minimal Case Essential Matrix Estimation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020), pp. 1–1. ISSN: 0162-8828, 2160-9292, 1939-3539. DOI: 10.1109/TPAMI.2020.3030161. arXiv: 1903.09067 (cit. on pp. 98, 104, 107, 119).

[286] L. Zhao, S. Huang, and G. Dissanayake. "Linear MonoSLAM: A Linear Approach to Large-Scale Monocular SLAM Problems". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. May 2014, pp. 1517–1523. DOI: 10.1109/ICRA.2014.6907053 (cit. on p. 24).

[287] L. Zhao, S. Huang, and G. Dissanayake. "Linear SFM: A Hierarchical Approach to Solving Structure-from-Motion Problems by Decoupling the Linear and Nonlinear Components". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 141 (July 2018), pp. 275–289. ISSN: 09242716. DOI: 10.1016/j.isprsjprs.2018.04.007 (cit. on p. 24).

[288] E. Zheng and C. Wu. "Structure from Motion Using Structure-Less Resection". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. Santiago, Chile: IEEE, Dec. 2015, pp. 2075–2083. ISBN: 978-1-4673-8391-2. DOI: 10.1109/ICCV.2015.240 (cit. on p. 14).

[289] D. Zhou, Y. Dai, and H. Li. "Ground Plane Based Absolute Scale Estimation for Monocular Visual Odometry". In: *arXiv:1903.00912 [cs]* (Mar. 2019). arXiv: 1903.00912 [cs] (cit. on p. 18).

[290] Q.-Y. Zhou, J. Park, and V. Koltun. "Open3D: A Modern Library for 3D Data Processing". In: *arXiv:1801.09847 [cs]* (Jan. 2018). arXiv: 1801.09847 [cs] (cit. on p. 163).

[291] S. Zhu, T. Shen, L. Zhou, R. Zhang, J. Wang, T. Fang, and L. Quan. "Parallel Structure from Motion from Local Increment to Global Averaging". In: *arXiv:1702.08601 [cs]* (June 2017). arXiv: 1702.08601 [cs] (cit. on p. 24).