Efficient Dense Registration, Segmentation, and Modeling Methods for RGB-D Environment Perception

Dissertation

zur

Erlangung des Doktorgrades (Dr. rer. nat.)

der

Mathematisch-Naturwissenschaftlichen Fakultät

der

Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von:

Jörg-Dieter Stückler

aus

Ettenheim

Bonn Januar, 2014

Angefertigt mit Genehmigung der
Mathematisch-Naturwissenschaftlichen Fakultät der Rheinischen
Friedrich-Wilhelms-Universität Bonn

1. Gutachter: Prof. Dr. Sven Behnke
2. Gutachter: Prof. Michael Beetz, PhD

Tag der Promotion: 26.09.2014
Erscheinungsjahr: 2014

Abstract

One perspective for artificial intelligence research is to build machines that perform tasks autonomously in our complex everyday environments. This setting poses challenges to the development of perception skills: A robot should be able to perceive its location and objects in its surrounding, while the objects and the robot itself could also be moving. Objects may not only be composed of rigid parts, but could be non-rigidly deformable or appear in a variety of similar shapes. Furthermore, it could be relevant to the task to observe object semantics. For a robot acting fluently and immediately, these perception challenges demand efficient methods.

This theses presents novel approaches to robot perception with RGB-D sensors. It develops efficient registration, segmentation, and modeling methods for scene and object perception. We propose multi-resolution surfel maps as a concise representation for RGB-D measurements. We develop probabilistic registration methods that handle rigid scenes, scenes with multiple rigid parts that move differently, and scenes that undergo non-rigid deformations. We use these methods to learn and perceive 3D models of scenes and objects in both static and dynamic environments.

For learning models of static scenes, we propose a real-time capable simultaneous localization and mapping approach. It aligns key views in RGB-D video using our rigid registration method and optimizes the pose graph of the key views. The acquired models are then perceived in live images through detection and tracking within a Bayesian filtering framework.

An assumption frequently made for environment mapping is that the observed scene remains static during the mapping process. Through rigid multi-body registration, we take advantage of releasing this assumption: Our registration method segments views into parts that move independently between the views and simultaneously estimates their motion. Within simultaneous motion segmentation, localization, and mapping, we separate scenes into objects by their motion. Our approach acquires 3D models of objects and concurrently infers hierarchical part relations between them using probabilistic reasoning. It can be

applied for interactive learning of objects and their part decomposition.

Endowing robots with manipulation skills for a large variety of objects is a tedious endeavor if the skill is programmed for every instance of an object class. Furthermore, slight deformations of an instance could not be handled by an inflexible program. Deformable registration is useful to perceive such shape variations, e.g., between specific instances of a tool. We develop an efficient deformable registration method and apply it for the transfer of robot manipulation skills between varying object instances.

On the object-class level, we segment images using random decision forest classifiers in real-time. The probabilistic labelings of individual images are fused in 3D semantic maps within a Bayesian framework. We combine our object-class segmentation method with simultaneous localization and mapping to achieve online semantic mapping in real-time.

The methods developed in this thesis are evaluated in experiments on publicly available benchmark datasets and novel own datasets. We publicly demonstrate several of our perception approaches within integrated robot systems in the mobile manipulation context.

Zusammenfassung

Wie können wir technische Systeme mit Fähigkeiten zur Umgebungswahrnehmung ausstatten, die es ihnen ermöglichen, intelligent zu handeln? Diese Fragestellung kommt in der Forschung zur Künstlichen Intelligenz in den unterschiedlichsten Kontexten auf. Beispielsweise wollen wir zukünftig immer weitere Bereiche in Fabriken automatisieren, die bisher ausschließlich menschlichen Arbeitern überlassen sind. Autonom fahrende Autos sind von einer kühnen Vision zu einem Entwicklungstrend in der Automobilbranche geworden. In den letzten Jahren haben wir auch einen großen Fortschritt in der Entwicklung von Roboterplattformen und -technologien gesehen, die uns einst in unseren Alltagsumgebungen unterstützen könnten. Aus diesen Entwicklungen ergeben sich stets neue Herausforderungen an die Umgebungswahrnehmung durch intelligente Systeme.

In dieser Arbeit beschäftigen wir uns mit Herausforderungen der visuellen Wahrnehmung in Alltagsumgebungen. Intelligente Roboter sollen sich selbst in ihrer Umgebung zurechtfinden, und Wissen über den Verbleib von Objekten erwerben können. Die Schwierigkeit dieser Aufgaben erhöht sich in dynamischen Umgebungen, in denen ein Roboter die Bewegung einzelner Teile differenzieren und auch wahrnehmen muss, wie sich diese Teile bewegen. Wenn ein Roboter sich selbst in dieser Umgebung bewegt, muss er auch seine eigene Bewegung von der Veränderung der Umgebung unterscheiden. Szenen können sich aber nicht nur durch die Bewegung starrer Teile verändern. Auch die Teile selbst können ihre Form in nicht-rigider Weise ändern.

Eine weitere Herausforderung stellt die semantische Interpretation von Szenengeometrie und -aussehen dar. Wir erwarten, dass intelligente Roboter auch selbständig neue Objekte entdecken können und die Zusammenhänge von Objekten begreifen. Die Bewegung von Objekten ist ein möglicher Hinweis, um Objekte ohne weiteres Vorwissen über die Szene zu vereinzeln und Zusammenhänge zu erkunden. Wenn wir eine Kategorisierung der Objekte vorgeben, sollen Roboter auch lernen, diese Kategorien in Bildern wiederzuerkennen.

Neben Genauigkeit und Zuverlässigkeit von Algorithmen zur Wahrnehmung, muss auch die Effizienz der Verfahren im Blick gehalten werden, da oft eine

flüssige und sofortige Handlung durch Roboter gewünscht ist. Dynamische Umgebungen verlangen oft ebenfalls Effizienz, wenn ein Algorithmus in Echtzeit den Veränderungen in der Szene folgen soll.

Seit einigen Jahren sind RGB-D Kamerasensoren kommerziell und kostengünstig erhältlich. Diese Entwicklung hatte einen starken Einfluß auf die Forschung im Bereich der Computer Vision. RGB-D Kameras liefern sowohl dichte Farb- als auch Tiefenmessungen in hoher Auflösung und Bildrate. Wir entwickeln unsere Methoden in dieser Arbeit für die visuelle Wahrnehmung mit dieser Art von Sensoren.

Eine typische Formulierung von Wahrnehmung ist es, einen Zustand oder eine Beschreibung zu finden, um Messungen mit Erwartungen in Einklang zu bringen. Für die geometrische Wahrnehmung von Szenen und Objekten entwickeln wir effiziente dichte Methoden zur Registrierung von RGB-D Messungen mit Modellen. Mit dem Begriff "dicht" beschreiben wir Ansätze, die alle verfügbaren Messungen in einem Bild verwenden, im Vergleich zu spärlichen Methoden, die das Bild beispielsweise zu einer Menge von interessanten Punkten in texturierten Bereichen reduzieren.

Diese Arbeit gliedert sich in zwei Teile. Im ersten Teil entwickeln wir effiziente Methoden zur Repräsentation und Registrierung von RGB-D Messungen. In Kapitel 2 stellen wir eine kompakte Repräsentation von RGB-D Messungen vor, die unseren effizienten Registrierungsmethoden zugrunde liegt. Sie fasst Messungen in einer 3D Volumenelement-Beschreibung in mehreren Auflösungen zusammen. Die Volumenelemente beinhalten Statistiken über die Punkte innerhalb der Volumen, die wir als Oberflächenelemente bezeichnen. Wir nennen unsere Repräsentation daher Multi-Resolutions-Oberflächenelement-Karten (engl. multi-resolution surfel maps, MRSMaps). Wir berücksichtigen in MRSMaps die typische Fehlercharakteristik von RGB-D Sensoren, die auf dem Prinzip der Projektion von texturiertem Licht beruhen. Bilder können effizient in MRSMaps aggregiert werden. Die Karten unterstützen auch die Fusion von Bildern aus mehreren Blickpunkten. Wir nutzen solche Karten für die Modell-Repräsentation von Szenen und Objekten.

Kapitel 3 führt eine Methode zur effizienten, robusten, und genauen Registrierung von MRSMaps vor, die Rigidheit der betrachteten Szene voraussetzt. Die Registrierung schätzt die Kamerabewegung zwischen den Bildern und gewinnt ihre Effizienz durch die Ausnutzung der kompakten multi-resolutionalen Darstellung der Karten. Während das Verfahren grobe bis feine Fehlregistrierungen korrigiert, wird Genauigkeit durch die Registrierung auf der feinsten gemeinsamen Auflösung zwischen den Karten erreicht. Die Verwendung von Farbe und lokalen Form- und Texturbeschreibungen erhöht die Robustheit des Verfahrens durch die Verbesserung der Assoziation von Oberflächenelementen zwischen den Karten. Die Registrierungsmethode erzielt hohe Bildverarbeitungsraten auf einer CPU. Wir demonstrieren hohe Effizienz, Genauigkeit und Robustheit unserer Methode im Vergleich zum bisherigen Stand der Forschung auf Vergleichsdaten-

sätzen.

In Kapitel 4 lösen wir uns von der Annahme, dass die betrachtete Szene zwischen Bildern statisch ist. Wir erlauben nun, dass sich rigide Teile der Szene bewegen dürfen, und erweitern unser rigides Registrierungsverfahren auf diesen Fall. Wir formulieren ein allgemeines Expectation-Maximization Verfahren zur dichten 3D Bewegungssegmentierung mit effizienten Approximationen durch Graph Cuts und variationaler Inferenz. Unser Ansatz segmentiert die Bildbereiche der einzelnen Teile, die sich unterschiedlich zwischen Bildern bewegen. Er findet die Anzahl der Segmente und schätzt deren Bewegung. Wir demonstrieren hohe Segmentierungsgenauigkeit und Genauigkeit in der Bewegungsschätzung unter Echtzeitbedingungen für die Verarbeitung.

Schließlich entwickeln wir in Kapitel 5 ein Verfahren für die Wahrnehmung von nicht-rigiden Deformationen zwischen zwei MRSMaps. Auch hier nutzen wir die multi-resolutionale Struktur in den Karten für ein effizientes Registrieren von grob zu fein. Wir schlagen Methoden vor, um aus den geschätzten Deformationen die lokale Bewegung zwischen den Bildern zu gewinnen. Wir evaluieren Genauigkeit und Effizienz des Verfahrens.

Der zweite Teil dieser Arbeit widmet sich der Verwendung unserer Kartenrepräsentation und Registrierungsmethoden für die Wahrnehmung von Szenen und Objekten. Kapitel 6 verwendet MRSMaps und unsere rigide Registrierungsmethode, um 3D Modelle von Szenen und Objekten zu lernen. Die Registrierung liefert die Kamerabewegung zwischen Schlüsselansichten auf Szene und Objekt. Diese Schlüsselansichten sind MRSMaps von ausgewählten Bildern aus der Kamerafahrt. Wir registrieren nicht nur zeitlich aufeinanderfolgende Schlüsselansichten, sondern stellen auch räumliche Beziehungen zwischen weiteren Paaren von Schlüsselansichten her. Die räumlichen Beziehungen werden in einem Simultanen Lokalisierungs- und Kartierungsverfahren (engl. simultaneous localization and mapping, SLAM) gegeneinander abgewogen, um die Blickposen der Schlüsselansichten in einem gemeinsamen Koordinatensystem zu schätzen. Von ihren Blickposen aus können die Schlüsselansichten dann in dichten Modellen übereinandergelegt werden. Wir entwickeln eine effiziente Methode, um neue räumliche Beziehungen zu entdecken, sodass die Kartierung in Echtzeit erfolgen kann. Weiterhin beschreiben wir ein Verfahren, um Objektmodelle im Kamerabild zu detektieren und initiale grobe Posenschätzungen herzustellen. Für das Verfolgen der Kamerapose bezüglich der Modelle, kombinieren wir die Genauigkeit unserer Registrierung mit der Robustheit von Partikelfiltern. Zu Beginn der Posenverfolgung, oder wenn das Objekt aufgrund von Verdeckungen oder extremen Bewegungen nicht weiter verfolgt werden konnte, initialisieren wir das Filter durch Objektdetektion. Das Verfahren verfolgt die Pose von Objekten in Echtzeit.

In Kapitel 7 wenden wir unsere erweiterten Registrierungsverfahren für die Wahrnehmung in nicht-rigiden Szenen und für die Übertragung von Objekthandhabungsfähigkeiten von Robotern an. Wir erweitern unseren rigiden Kartierungs-

ansatz aus Kapitel 6 auf dynamische Szenen, in denen sich rigide Teile bewegen. Die Methode extrahiert wiederum Schlüssenansichten aus RGB-D Video, die nun gegen weitere Ansichten bewegungssegmentiert werden. Die Bewegungssegmente werden zueinander in Bezug gesetzt, um Äquivalenz- und Teilebeziehungen von Objekten probabilistisch zu inferieren, denen die Segmente entsprechen. Unsere Registrierungsmethode liefert Bewegungschätzungen zwischen den Segmentansichten der Objekte, die wir als räumliche Beziehungen in einem SLAM Verfahren nutzen, um die Blickposen der Segmente zu schätzen. Aus diesen Blickposen wiederum können wir die Bewegungssegmente in dichten Objektmodellen vereinen.

Objekte einer Klasse teilen oft eine gemeinsame Topologie von funktionalen Elementen. Während Instanzen sich in Form unterscheiden können, entspricht die Korrespondenz von funktionalen Elementen oft auch einer Korrespondenz in den Formen der Objekte. Wir nutzen diese Eigenschaft aus, um die Handhabung eines Objektes durch einen Roboter auf neue Objektinstanzen derselben Klasse zu übertragen. Formkorrespondenzen werden durch unsere deformierbare Registrierung ermittelt. Wir beschreiben Handhabungsfähigkeiten durch Greifposen und Bewegungstrajektorien von Bezugssystemen im Objekt wie z. B. Werkzeugendeffektoren.

Abschließend in Teil II entwickeln wir einen Ansatz, der Kategorien von Objekten in RGB-D Bildern erkennt und segmentiert (Kapitel 8). Die Segmentierung basiert auf Ensemblen randomisierter Entscheidungsbäume, die Geometrie- und Texturmerkmale zur Klassifikation verwenden. Die Verfügbarkeit von dichter Tiefe ermöglicht es, die Merkmale gegen Skalenunterschiede im Bild zu normalisieren. Wir fusionieren Segmentierungen von Einzelbildern einer Szene aus mehreren Ansichten in einer semantischen Objektklassenkarte mit Hilfe unseres SLAM-Verfahrens.

Die vorgestellten Methoden werden auf öffentlich verfügbaren Vergleichsdatensätzen und eigenen Datensätzen evaluiert. Einige unserer Ansätze wurden auch in integrierten Robotersystemen für mobile Objekthantierungsaufgaben öffentlich demonstriert. Sie waren ein wichtiger Bestandteil für das Gewinnen der RoboCup-Roboterwettbewerbe in der RoboCup@Home Liga in den Jahren 2011, 2012 und 2013.

Acknowledgements

My gratitude goes to everyone at the Autonomous Intelligent Systems group at the University of Bonn for providing a great working environment. I address special thanks to my advisor Prof. Sven Behnke for his support and inspiring discussions. He created a motivating environment in which I could develop my research. I thank Prof. Michael Beetz for agreeing to review my thesis. The work of his group on 3D perception and intelligent mobile manipulation systems greatly inspired my research. I acknowledge all the hard work of the many students who contributed to our RoboCup competition entries. Deepest thanks belong to my love Eva who ceaselessly supported me during the intense time of the preparation of this thesis.

Für Eva und Enno

Contents

1. Introduction

How can we endow machines with the perception skills that enable them to act intelligently? Artificial intelligence research poses this question in many contexts such as the automation of the factories of the future, self-driving cars, and robots that assist in our homes. While in recent years, research has achieved tremendous progress in these areas, many challenges remain.

In this thesis, we consider challenges for visual perception in everyday environments. Intelligent robots need to perceive the whereabouts of themselves and objects in their surrounding. Difficulty increases in dynamic scenes: a robot should distinguish what parts in a scene are moving and how they change. This becomes even more challenging while a robot is moving. Then, it must differentiate its ego-motion from the motion of parts in the scene. Scene variations could not only be caused by moving rigid parts, but the parts themselves may vary in shape by non-rigid deformations.

A further challenge is the semantic interpretation of scene geometry and appearance. Intelligent robots should be able to discover novel objects and parse the semantic relation of objects. Without prior knowledge on the objects in a scene, motion can be used as a cue for singularizing objects and understanding their relations. Robots can also learn to recognize the category of objects in images.

Besides accuracy and robustness of perception algorithms, efficiency is another important dimension, as robots should act fluently and immediately. Frequently, dynamics also pose constraints on efficiency, as the algorithm has to keep track of changes in real-time.

The recent broad availability of RGB-D cameras had significant impact on the field of computer vision. These cameras provide dense color and depth images at high resolution and frame-rate. We present novel efficient approaches to visual perception with such sensors.

Perception can typically be phrased as finding a state or description that brings observations in alignment with expectations. For the geometric perception of scenes and object instances, we develop efficient dense registration meth-

ods that allow for aligning RGB-D measurements and models. The notion of dense describes approaches that utilize all available measurements in an image, in contrast to sparse approaches that, for instance, reduce an image to a set of interest points in textured image regions.

Underlying our efficient registration methods is a concise representation of RGB-D measurements. We represent RGB-D images densely in multi-resolution surfel maps(MRSMaps). The maps transform the images into a 3D volume element (voxel) representation that maintains statistics on the RGB-D measurements at multiple resolutions. We consider the error characteristics typical to textured-light projecting RGB-D cameras and propose an efficient aggregation technique for RGB-D images. The maps not only support the storage of a single image. They can also fuse images from multiple view points, such that they are suitable as multi-view models of scenes and objects.

We develop methods to register MRSMaps of

- rigid scenes,
- scenes with multiple rigid parts that move differently, and
- scenes with continuous shape deformations.

In static scenes, efficient rigid registration of RGB-D images recovers the camera motion between the images. The method is efficient through the concise representation in MRSMaps. It exploits the multi-resolution structure of the maps for correcting for coarse to fine misalignments, and achieves accuracy through utilizing the finest resolution common between the maps. Robustness of the registration is obtained by the use of color and local shape-texture descriptions for making associations. By registering an image towards a model, we find the pose of the camera relative to the model. Such models can represent rigid scenes or objects. Rigid registration also enables to learn the models of static scenes and objects. While the camera is moving, we estimate the motion of the camera by aligning the images in a common model frame through simultaneous localization and mapping (SLAM).

We also study the perception in dynamic scenes in which the moving parts are rigid. Motion is a fundamental grouping cue that we combine with geometry and texture hints for dense motion segmentation. We extend rigid registration towards rigid multi-body registration in order to find the moving parts between two images and estimates their motion. We formulate a general expectation-maximization (EM) framework for dense 3D motion segmentation with efficient approximations through graph cuts and variational inference. We utilize the method to discover the moving objects in RGB-D video and to build dense models. By observing the objects split and merge, we reason on part hierarchies, i.e., our approach acquires scene semantics in an unsupervised way.

For perceiving continuous deformations of objects, we develop an efficient deformable registration method. The method extends a state-of-the-art approach

to the efficient processing of RGB-D measurements by exploiting the multi-resolution structure in MRSMaps. We apply the method for object manipulation skill transfer. Objects of the same class often share a common topology of functional parts. While instances of the same class may differ in shape, in many cases, correspondences between the functional parts can be established by matching shape between the objects. This can be exploited to transfer manipulation skills between several objects of the same class, which would otherwise be a tedious endeavor, if the skill would need to be programmed separately for every single instance. Deformable registration recovers such shape variations.

To recognize objects by their category, we train random decision forest classifiers. The classifiers segments images efficiently into several object classes. The availability of depth allows for scale-invariant recognition by geometry and appearance. We make the observations of object-class semantics persistent in a semantic map of the environment, such that a robot memorizes the whereabouts of objects of specific categories.

1.1. Key Contributions

This thesis proposes novel approaches to efficient RGB-D environment perception. The approaches enable

- to acquire 3D models of scenes and objects,
- to perceive these models in live images,
- to observe moving rigid parts and shape variations in scenes,
- to parse the semantics of the environment from either motion cues or pretrained object-class knowledge, and to make this knowledge persistent in semantic models.

More specifically, this thesis makes the following contributions:

- We propose multi-resolution surfel maps(MRSMaps)—a concise representation of RGB-D measurements which is suitable for efficient registration and allows for aggregating multiple images within a single multi-view map (Ch. 2).
- Chapter 3 details an efficient, robust, and accurate registration method for MRSMaps that assumes rigidness of the viewed scene. The registration method achieves high frame rate on a CPU. We demonstrate state-of-the-art results in run-time and accuracy.
- In Chapter 4 we release the assumption on static scenes, and propose an efficient registration method for MRSMaps that segments scenes into

the rigid parts that move differently between two images. The approach concurrently estimates the rigid body motion of the parts.

- A run-time efficient deformable registration method for MRSMaps without the assumption on the rigidness of parts is presented in Chapter 5.
- Chapter 6 utilizes MRSMaps and our rigid registration method to learn 3D models of scenes and objects in a key-view based SLAM approach for which we demonstrate state-of-the-art results. We also propose means for detecting objects in RGB-D images, to estimate their 6-degree-of-freedom (DoF) pose, and to track them in real-time.
- Non-rigid registration enables novel approaches to semantic scene parsing from motion cues (Ch. 7). We segment and estimate the motion of rigid parts in a scene, and acquire models of these moving parts using SLAM techniques. By observing the parts split and merge, we find hierarchical relations between them. We also develop an approach that applies deformable registration for the transfer of robot skills between objects.
- In chapter 8 we propose an efficient object-class segmentation approach based on random decision forests(RFs) that is trained on specific object-classes. We make the segmentations of individual RGB-D images persistent in a multi-resolution semantic map using our SLAM approach. Uncertainty in the segmentation of individual images is fused in a 3D map using a Bayesian framework. This approach yields state-of-the-art results for RGB-D object-class segmentation.

A detailed discussion of our contributions in context with the state-of-the-art is made in the individual chapters.

1.2. Publications

Parts of this thesis have been published in journals and conference proceedings. The publications are provided in chronological order.

Journals:

- Jörg Stückler, Benedikt Waldvogel, Hannes Schulz, and Sven Behnke. *Dense Real-Time Mapping of Object-Class Semantics from RGB-D Video.* Accepted for publication in Journal of Real-Time Image Processing, to appear 2014. Chapter 8.
- Jörg Stückler and Sven Behnke. *Multi-Resolution Surfel Maps for Efficient Dense 3D Modeling and Tracking.* In Journal of Visual Communication and Image Representation, January 2014. Chapters 2, 3, and 6.

- Jörg Stückler, Dirk Holz, and Sven Behnke. *RoboCup@Home: Demonstrating Everyday Manipulation Skills in RoboCup@Home.* In IEEE Robotics & Automation Magazine, June 2012. Chapter 6.

Conferences:

- Jörg Stückler and Sven Behnke. *Efficient Deformable Registration of Multi-Resolution Surfel Maps for Object Manipulation Skill Transfer.* In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2014. Chapters 5 and 7.

- Jörg Stückler, David Droeschel, Kathrin Gräve, Dirk Holz, Michael Schreiber, Angeliki Topalidou-Kyniazopoulou, Max Schwarz, and Sven Behnke. *Increasing Flexibility of Mobile Manipulation and Intuitive Human-Robot Interaction in RoboCup@Home.* RoboCup 2013, Robot Soccer World Cup XVII, Springer, LNCS, 2014. Chapters 5, 6, and 7.

- Manus McElhone, Jörg Stückler, and Sven Behnke. *Joint Detection and Pose Tracking of Multi-Resolution Surfel Models in RGB-D.* In Proceedings of the 6th European Conference on Mobile Robots (ECMR), Barcelona, Spain, September 2013. Chapter 6.

- Jörg Stückler and Sven Behnke. *Efficient Dense 3D Rigid-Body Motion Segmentation in RGB-D Video.* In Proceedings of the British Machine Vision Conference (BMVC), Bristol, UK, September 2013. Chapter 4.

- Jörg Stückler and Sven Behnke. *Hierarchical Object Discovery and Dense Modelling From Motion Cues in RGB-D Video.* In Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI), Beijing, China, August 2013. Chapter 7.

- Jörg Stückler, Ishrat Badami, David Droeschel, Kathrin Gräve, Dirk Holz, Manus McElhone, Matthias Nieuwenhuisen, Michael Schreiber, Max Schwarz, and Sven Behnke. *NimbRo@Home: Winning Team of the RoboCup-@Home Competition 2012.* RoboCup 2012, Robot Soccer World Cup XVI, Springer, LNCS, 2013. Chapter 6.

- Jörg Stückler, Nenad Biresev, and Sven Behnke. *Semantic Mapping Using Object-Class Segmentation of RGB-D Images.* In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vilamoura, Portugal, October 2012. Chapter 8.

- Jörg Stückler and Sven Behnke. *Integrating Depth and Color Cues for Dense Multi-Resolution Scene Mapping Using RGB-D Cameras.* In Proceedings of the IEEE International Conference on Multisensor Fusion and Information Integration (MFI), Germany, September 2012. Chapter 6.

- Jörg Stückler and Sven Behnke. *Model Learning and Real-Time Tracking using Multi-Resolution Surfel Maps.* In Proceedings of the AAAI Conference on Artificial Intelligence (AAAI-12), Toronto, Canada, July 2012. Chapter 6.

- Jörg Stückler, David Droeschel, Kathrin Gräve, Dirk Holz, Jochen Kläß, Michael Schreiber, Ricarda Steffens, and Sven Behnke. *Towards Robust Mobility, Flexible Object Manipulation, and Intuitive Multimodal Interaction for Domestic Service Robots.* RoboCup 2011, Lecture Notes in Computer Science (LNCS), vol. 7416, 2012. Chapter 6.

- Jörg Stückler and Sven Behnke. *Robust Real-Time Registration of RGB-D Images using Multi-Resolution Surfel Representations.* In Proceedings of the German Conference on Robotics (ROBOTIK), Munich, Germany, May 2012. Chapters 2 and 3.

- Jörg Stückler and Sven Behnke. *Following Human Guidance to Cooperatively Carry a Large Object.* In Proceedings of the 11th IEEE-RAS International Conference on Humanoid Robots (Humanoids), Bled, Slovenia, October 2011. Chapter 6.

- Jörg Stückler and Sven Behnke. *Combining Depth and Color Cues for Scale- and Viewpoint-Invariant Object Segmentation and Recognition using Random Forests.* In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Taipei, Taiwan, October 2010. Chapter 8.

The following conference publications are closely related with the methods presented in this thesis and have been written during the time as a research assistant.

- Mark Schadler, Jörg Stückler, and Sven Behnke. *Multi-Resolution Surfel Mapping and Real-Time Pose Tracking using a Continuously Rotating 2D Laser Scanner.* In Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR), Linköping, Sweden, October 2013.

  This work is the outcome of a master thesis that I was supervising. It transfers the RGB-D image representation, rigid registration, and scene modeling methods that are presented in this thesis to mapping and localization for mobile robot navigation with 3D laser scanners. It was used as the mapping and localization component for our entry NimbRo Centauro to the DLR SpaceBot Cup 2013.

- Torsten Fiolka, Jörg Stückler, Dominik Klein, Dirk Schulz, and Sven Behnke. *Distinctive 3D Surface Entropy Features for Place Recognition.* In

Proceedings of the 6th European Conference on Mobile Robots (ECMR), Barcelona, Spain, September 2013.

- Torsten Fiolka, Jörg Stückler, Dominik Alexander Klein, Dirk Schulz, and Sven Behnke. *SURE: Surface Entropy for Distinctive 3D Features.* In Proceedings of Spatial Cognition 2012, Germany, September 2012.

  The preceding two publications are outcomes of a master thesis I was supervising. They present the SURE interest point detector and descriptor for RGB-D images and 3D point clouds, and its application for place recognition. The underlying representation are MRSMaps.

- German Martin Garcia, Dominik Alexander Klein, Jörg Stückler, Simone Frintrop, and Armin B. Cremers. *Adaptive Multi-cue 3D Tracking of Arbitrary Objects.* In Proceedings of DAGM-OAGM 2012, Graz, Austria, August 2012.

  This work is a publication of the results of a master thesis I was cosupervising. It tracks position and bounding box of objects in 3D using an adaptive shape and appearance model.

- Jochen Kläß, Jörg Stückler and Sven Behnke. *Efficient Mobile Robot Navigation using 3D Surfel Grid Maps.* In Proceedings of the German Conference on Robotics (ROBOTIK), Munich, Germany, May 2012.

  This publication reports on a Diploma thesis I was supervising. It uses a single resolution surfel grid for representing 3D laser scans of the environment. It tackles mapping, localization, and navigation with this representation.

- Bastian Oehler, Jörg Stückler, Jochen Welle, Dirk Schulz, and Sven Behnke. *Efficient Multi-Resolution Plane Segmentation of 3D Point Clouds.* In Proceedings of the International Conference on Intelligent Robotics and Applications (ICIRA), Aachen, Germany, December 2011.

  This work presents the outcome of a Diploma thesis I was supervising. Planes are extracted efficiently from depth images and 3D point clouds within a multi-resolution Hough voting framework. The underlying representation for the images and 3D point clouds are MRSMaps.

1.3. Open-Source Software Releases

We provide an open-source implementation of MRSMaps[1]. The current release includes our approaches to RGB-D image representation, registration, and scene and object modeling and tracking. The release of our software gives other researchers the opportunity to use and to build on top of our methods in their own research, to compare their results to our approach, and to validate our methods.

1.4. Collaborations

Parts of this thesis have been developed in collaboration with others. The joint object detection and tracking approach in a particle filter framework in Ch. 6 extends the master thesis of McElhone (2013) which I was supervising. I also supervised the master thesis of Biresev (2012) which applied my previous work on object-class segmentation using random forests (Stückler and Behnke, 2010) for semantic mapping. The semantic mapping approach has been extended towards online operation in Ch. 8. The approach operates also in real-time due to a GPU variant of the random forest classifier implemented by Waldvogel (2013) whose thesis was supervised by Hannes Schulz.

[1] http://code.google.com/p/mrsmap/

Part I.

RGB-D Representation and Registration Methods

2. RGB-D Image Representation in Multi-Resolution Surfel Maps

In this chapter, we develop a novel representation for RGB-D measurements. It is suited for single images as well as for aggregating several RGB-D images from different view-points. We denote this representation multi-resolution surfel map (MRSMap), since it maps RGB-D image content to surface elements(surfels) at multiple 3D resolutions.

We design MRSMaps as an image representation that respects sensor characteristics and provides the basis for efficient registration. We overlay voxel grids at multiple resolutions over the RGB-D measurements. The point set measured within a voxel is represented as surface element (surfel). When adding image content to a map, we limit the maximum resolution for surfels with distance to the sensor (see Fig. 2.1, left). If only one image is incorporated into a map, its multi-resolution structure is *local*, since with increasing distance from the sensor

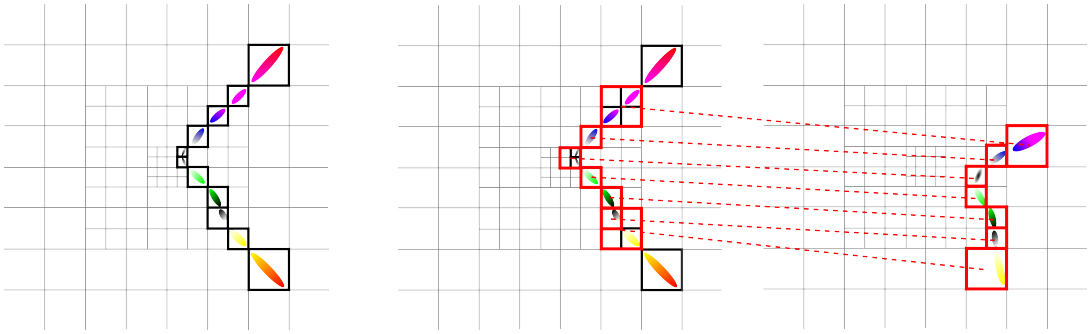

Figure 2.1.: Multi-resolution surfel maps represent RGB-D data as surfels at multiple resolutions (left). The maximum resolution is limited with distance to the sensor. We represent the data also at every lower resolution, such that surfels can be easily compared and matched at the finest resolution common between maps (right).

Figure 2.2.: Infrared textured light cameras provide RGB and depth images at good quality and high framerates. Left: Asus Xtion Pro Live. Center: RGB image. Right: Depth image (depth color coded).

origin, the maximum resolution decreases in which measurement statistics are aggregated.

By restricting the resolution with distance, our maps capture distance dependent degradation of measurement quality which is a typical property of RGB-D sensors. When using local multi-resolution, it is beneficial to represent RGB-D data at all resolutions concurrently – not only at the maximum resolution possible. In this way, maps that have been acquired from differing view-points can be matched at the finest resolution common between the maps (see Fig. 2.1, right).

We choose to compress the measured point sets into sample mean and covariance. This makes the computational effort for comparing and matching the content of a voxel constant and equal across resolutions. An appropriate choice of the distance-dependent limit spares unnecessary computations on high detail that corresponds to measurement noise, while it retains the fine-detailed scene structure available in the data.

2.1. RGB-D Sensor Characteristics

Measurement principles of current RGB-D sensors are mainly based on triangulation or time-of-flight. The Microsoft Kinect camera has had significant impact in computer vision and robotics due to its simple use, good quality depth, and low cost. Similar to textured-light stereo cameras for which a textured pattern in the visible light spectrum is projected into the scene, these RGB-D cameras project a speckle pattern in the infrared (IR) spectrum. It is measured with a dedicated IR image sensor (see Fig. 2.2, left), while color is observed with an additional RGB image sensor. The cameras provide VGA resolution (640×480) depth and RGB images at a framerate of 30 Hz. Some cameras such as the Asus Xtion Pro Live time-synchronize both image types in hardware, which is especially useful in moving scenes.

Depth is determined through correlation of the measured speckle pattern with a stored reference measurement which is recorded on a planar target during

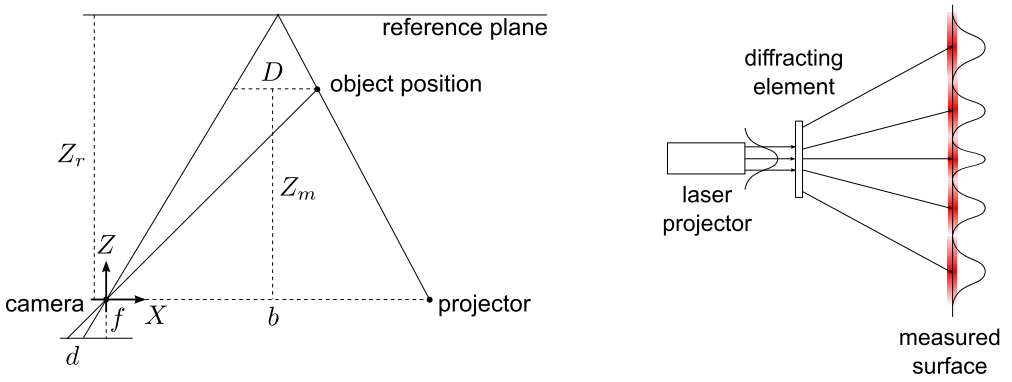

Figure 2.3.: Measurement principle of structured-light cameras. Left: Depth is estimated by measuring disparity of points in an IR projected speckle pattern towards a reference measurement. Right: Under a Gaussian beam profile of the laser, the intensity profile of the speckles flattens with distance from the optical axis.

factory calibration. Khoshelham and Elberink (2012) go into the details of the measurement principle, which we briefly restate in the following. An object is placed at a depth Z_m from the IR sensor (see Fig. 2.3). It is visible at a specific point in the projected speckle pattern. On the reference plane, depth Z_r has been measured for this point. The disparity d is the shift of the speckle point between its position on the reference plane and its new image position when measuring the object. We define D as the disparity of the speckle point on the plane through the object parallel to the reference plane. The similarity of triangles gives the relations

$$\frac{d}{f} = \frac{D}{Z_m} \tag{2.1}$$

and

$$\frac{D}{Z_r - Z_m} = \frac{b}{Z_m}, \tag{2.2}$$

where b is the baseline between IR camera and projector, and f is the focal length of the camera. From these relations the 3D coordinates of the object are determined by

$$\begin{aligned} Z_m &= \left(Z_r + \frac{1}{fb} d \right)^{-1} \\ X_m &= \frac{Z_m}{f} (x_m - x_c + \delta x) \\ Y_m &= \frac{Z_m}{f} (y_m - y_c + \delta y), \end{aligned} \tag{2.3}$$

where (x_m, y_m) and (X_m, Y_m, Z_m) are the measured image and 3D positions of the object, x_c and y_c are the optical center coordinates, and δ_x and δ_y correct for

lens distortion. Thus, measured depth is inversely related to disparity. Using the recovered 3D position of each pixel in the depth image, corresponding points in the RGB image can be found through projection. This process requires known extrinsics between RGB and IR camera and the intrinsic calibration of the RGB camera.

Khoshelham and Elberink (2012) identify three types of measurement errors that do not stem from imperfect calibration. Assuming Gaussian noise in disparity measurements, this noise can be propagated to the depth measurement using first-order error propagation:

$$\sigma_{Z_m}^2 = \left(\frac{\partial Z_m}{\partial d}\right) \sigma_d^2 \left(\frac{\partial Z_m}{\partial d}\right) = \frac{1}{(fb)^2} Z_m^4 \sigma_d^2, \tag{2.4}$$

hence, the standard deviation in depth is proportional to the squared depth to the sensor. Depth is also involved in the calculation of the X_m and Y_m coordinates in 3D of the object point. By propagating disparity uncertainty to X_m and Y_m,

$$\begin{aligned} \sigma_{X_m}^2 &= \left(\frac{\partial X_m}{\partial d}\right) \sigma_d^2 \left(\frac{\partial X_m}{\partial d}\right) = \frac{1}{f^4 b^2} \left(x_m - x_c + \delta x\right)^2 Z_m^4 \sigma_d^2, \\ \sigma_{Y_m}^2 &= \left(\frac{\partial Y_m}{\partial d}\right) \sigma_d^2 \left(\frac{\partial Y_m}{\partial d}\right) = \frac{1}{f^4 b^2} \left(y_m - y_c + \delta y\right)^2 Z_m^4 \sigma_d^2, \end{aligned} \tag{2.5}$$

we see that also these standard deviations depend on the squared depth to the sensor. Notably, they increase with distance from the optical center on the image plane. A further effect is that for a specific number of points per unit area, point density is inversely related to the squared depth to the sensor. Finally, since disparity is discretized into 10 bits for encoding its value before transmitting it via USB, the difference in depth between adjacent disparity discretizations,

$$\Delta_Z(d) = Z_m(d) - Z_m(d-1) = \frac{1}{fb} Z_m^2, \tag{2.6}$$

is also proportional to the squared depth.

Measurement noise, however, is also affected by the local quality of the speckle pattern, since it influences the quality of the disparity measurement. Assuming a Gaussian IR laser beam that illuminates a diffraction element to produce the speckle pattern, the intensity profile of the speckles flattens with distance from the beam's optical axis (Ohtsubo and Asakura, 1977). By this, disparity estimation through cross-correlation is less accurate with distance from the optical axis. The beam's optical axis approximately coincides with the image sensor's optical center at distances $d \gg b$, such that uncertainty in disparity can be expressed as a function of distance from the optical center, i.e.

$$\sigma_d := \sigma_d(x_m - x_c + \delta x, y_m - y_c + \delta y). \tag{2.7}$$

We conclude that these measurement properties should be incorporated into image representations to model the measured depth readings.

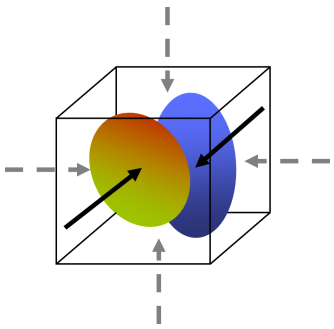

Figure 2.4.: Surfel view directions. We support up to six surfels for orthogonal view directions onto the voxel faces.

2.2. Multi-Resolution Surfel Maps

In MRSMaps, we represent the joint color and shape distribution of RGB-D measurements at multiple resolutions in 3D. We use octrees as a natural data structure for this purpose. The tree subdivides the represented 3D volume into cubic voxels at various resolutions, where resolution is defined as the inverse of the cube's side length. A node in the tree corresponds to a single voxel. Inner nodes branch to at least one of eight child nodes, dividing the voxel of the inner node into eight equally sized sub-voxels. The nodes at the same depth d in the tree share a common cube resolution $\rho(d) := 2^d \rho(0)$ which is a power of 2 of the cube resolution of the root node at depth 0.

In each node of the tree, i.e., inner nodes as well as leaf nodes, we store statistics on the joint spatial and color distribution of the points $\mathcal{P}$ within its volume. The distribution is approximated with sample mean μ and covariance Σ of the data, i.e., we model the data as normally distributed in a node's volume.

We denote the local description of voxel content as surfel s. It describes the local shape and color distribution within the voxel by the following attributes:

- mean $\mu_s \in \mathbb{R}^6$ and covariance $\Sigma_s \in \mathbb{R}^6$, where the first three coordinates μ_s^p model the 3D coordinates of the points within the voxel and the latter three dimensions $\mu_s^c = \left(\mu_s^L, \mu_s^\alpha, \mu_s^\beta\right)^T$ describe color,
- a surface normal $n_s \in \mathbb{R}^3$ pointing to the sensor origin and normalized to unit length,
- a local shape-texture descriptor h_s.

Since we build maps of scenes and objects from several perspectives, multiple distinct surfaces may be contained within a node's volume. We model this by maintaining multiple surfels in a node that are visible from different view directions (see Fig. 2.4). We use up to six orthogonal view directions $v \in \mathcal{V} :=$

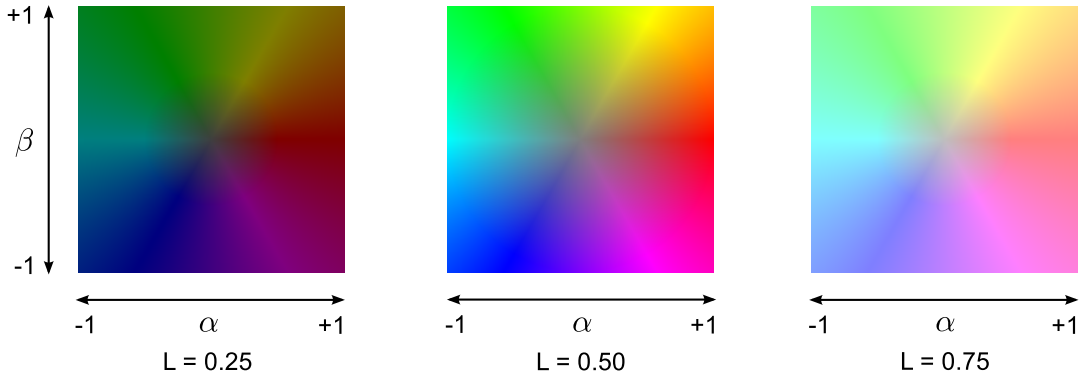

Figure 2.5.: $\alpha\beta$ chrominances for different luminance values.

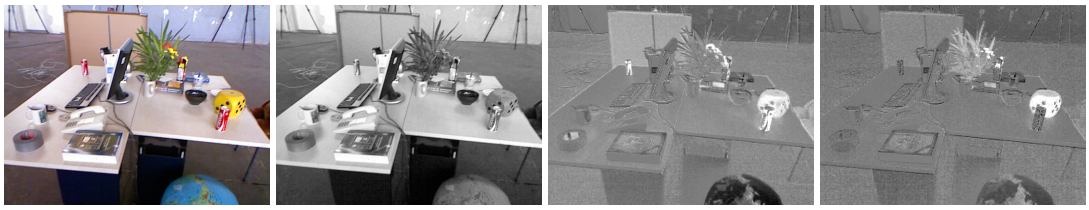

Figure 2.6.: L$\alpha\beta$ color space example. From left to right: Color image, L-, α-, β-channel.

$\{\pm e_x, \pm e_y, \pm e_z\}$ aligned with the basis vectors e_x, e_y, e_z of the map reference frame. When adding a new point p to the map, we determine the view direction onto the point $v_p = T_c^m p$ and associate it with the surfel belonging to the most similar view direction,

$$v' = \arg\max_{v \in \mathcal{V}} \left\{ v^T v_p \right\}. \tag{2.8}$$

The transform T_c^m maps p from camera to map frame.

By maintaining the joint distribution of points and color in a 6D Gaussian distribution, we also model the spatial distribution of color. In order to separate chrominance from luminance information and to represent chrominances in Cartesian space, we choose a variant of the HSL color space. We define the L$\alpha\beta$ color space through

$$\begin{aligned} L &:= \frac{1}{2}\left(\max\{R,G,B\} + \min\{R,G,B\}\right), \\ \alpha &:= R - \frac{1}{2}G - \frac{1}{2}B, \text{ and} \\ \beta &:= \frac{\sqrt{3}}{2}(G-B). \end{aligned} \tag{2.9}$$

The chrominances α and β represent hue and saturation of the color (Hanbury, 2008) and L its luminance (see Figs. 2.5 and 2.6).

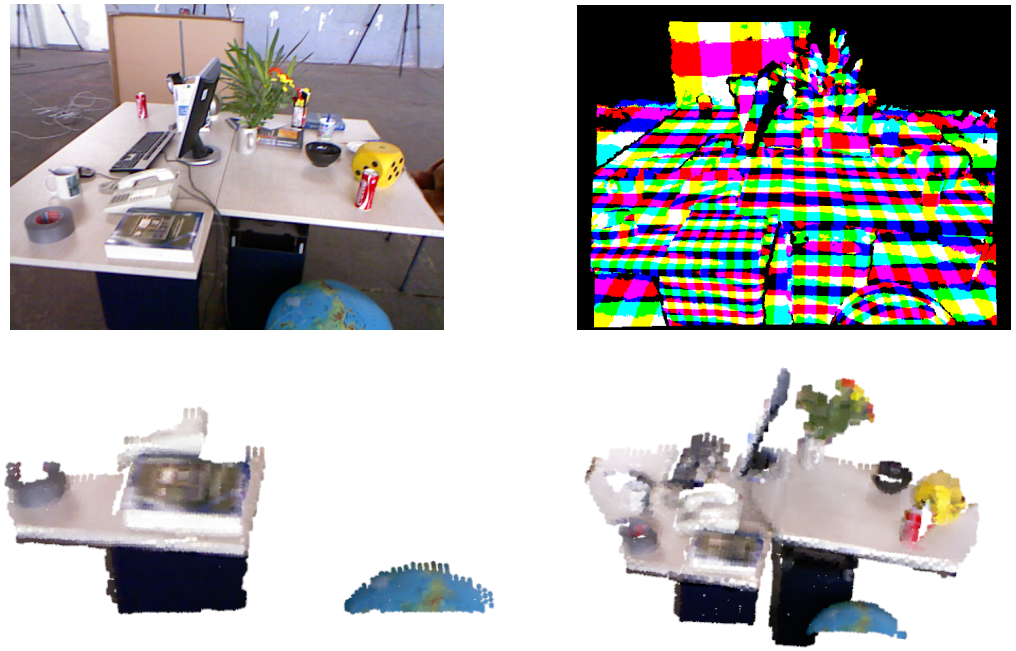

Figure 2.7.: Multi-resolution surfel map aggregation from an RGB-D image. Top left: RGB image of the scene. Top right: Maximum voxel resolution coding, color codes octant of the leaf in its parent's voxel (max. resolution $(0.0125\,\mathrm{m})^{-1}$). Bottom: 15 samples per color and shape surfel at $(0.025\,\mathrm{m})^{-1}$ (left) and at $(0.05\,\mathrm{m})^{-1}$ resolution (right).

Surface normals n are determined from the eigen decomposition of the point sample covariance in a local neighborhood at the surfel. We set the surface normal to the eigenvector that corresponds to the smallest eigenvalue, and direct the normal towards the view-point. Due to the discretization of the 3D volume into voxels, surfels may only receive points on a small surface patch compared to the voxel resolution. We thus smooth the normals by determining the normal from the covariance of the surfel and adjacent surfels in the voxel grid.

Neighboring voxels can efficiently be found using precalculated look-up tables (Zhou et al., 2011). We store the pointers to neighbors explicitly in each node to achieve better run-time efficiency than tracing the neighbors through the tree. The octree representation is still more memory-efficient than a multi-resolution grid, as it only allocates voxels that contain the 3D surface observed by the sensor.

2.2.1. Modeling Measurement Errors

We control the maximum resolution in the tree to consider the typical property of RGB-D sensors that measurement errors increase quadratically with depth

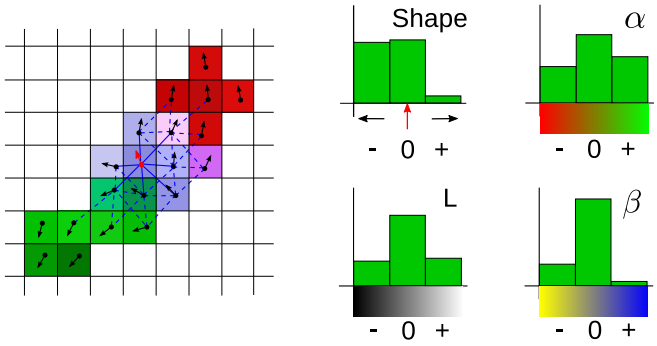

Figure 2.8.: 2D illustration of our local shape-texture descriptor. We determine a local description of shape, chrominance (α, β), and luminance (L) contrasts to improve the association of surfels. Each node is compared to its 26 neighbors. We smooth the descriptors between neighbors.

and with distance from the optical center on the image plane (see Sec. 2.1). We adapt the maximum resolution $\rho_{\max}(p)$ at a point p with the squared distance to the optical center,

$$\rho_{\max}(p) = \frac{1}{\lambda_\rho \|p\|_2^2}, \tag{2.10}$$

where λ_ρ is a factor that is governed by pixel as well as disparity resolution and noise and can be determined empirically. Fig. 2.7 shows the map representation of an RGB-D image in two example resolutions.

2.2.2. Shape-Texture Descriptor

We construct descriptors of shape and texture in the local neighborhood of each surfel (see Fig. 2.8). Similar to fast point feature histograms(FPFHs) (Rusu et al., 2009), we first build three-bin histograms h_s^{sh} of the three angular surfel-pair relations between the query surfel s and its up to 26 neighbors s' at the same resolution and view direction. The three angles are measured between the normals of both surfels $\measuredangle(n, n')$ and between each normal and the line $\Delta\mu := \mu - \mu'$ between the surfel means, i.e., $\measuredangle(n, \Delta\mu)$ and $\measuredangle(n', \Delta\mu)$. Each surfel-pair relation is weighted with the number of points in the neighboring node. We smooth the histograms to better cope with discretization effects by adding the histogram of neighboring surfels with a factor $\gamma = 0.1$ and normalize the histograms by the total number of points.

Similarly, we extract local histograms of luminance h_s^L and chrominance h_s^α, h_s^β contrasts. We bin luminance and chrominance differences between neighboring surfels into positive, negative, or insignificant. The shape and texture histograms are concatenated into a shape-texture descriptor h_s of the surfel. Fig. 2.9 shows

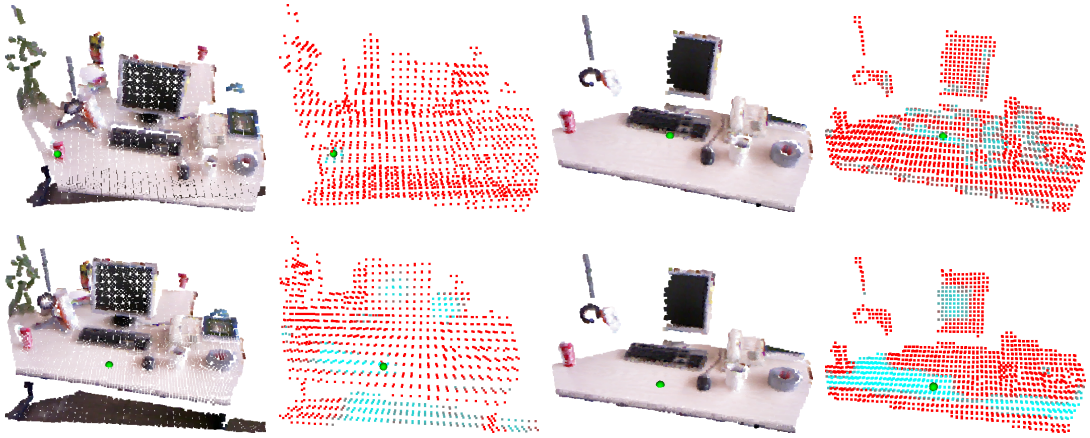

Figure 2.9.: Similarity in shape-texture descriptor for blob- (top left) and edge-like structures (top right) and in planar, textureless structures (bottom). The MRSMaps are shown as voxel centers at a single resolution each (left images). Feature similarity towards a reference point (green dot) is visualized by colored sufel means (right images, red: low, cyan: high similarity).

feature similarity on color blobs, edges, and planar structures determined using the Euclidean distance between the shape-texture descriptors.

2.2.3. Efficient RGB-D Image Aggregation

Instead of computing mean and covariance in the nodes with a two-pass algorithm, we use a one-pass update scheme with high numerical accuracy (Chan et al., 1979). It determines the sufficient statistics $\mathcal{S}(\mathcal{P}) := \sum_{p\in\mathcal{P}} p$ and $\mathcal{S}^2(\mathcal{P}) := \sum_{p\in\mathcal{P}} pp^T$ of the normal distribution from the statistics of two point sets $\mathcal{P}^A$ and $\mathcal{P}^B$ through

$$\begin{aligned} \mathcal{S}(\mathcal{P}^{A\cup B}) &\leftarrow \mathcal{S}(\mathcal{P}^A) + \mathcal{S}(\mathcal{P}^B), \\ \mathcal{S}^2(\mathcal{P}^{A\cup B}) &\leftarrow \mathcal{S}^2(\mathcal{P}^A) + \mathcal{S}^2(\mathcal{P}^B) + \frac{\delta\delta^T}{N_A N_B (N_A + N_B)}, \end{aligned} \tag{2.11}$$

where $N_{(.)} := |\mathcal{P}^{(\cdot)}|$ and $\delta := N_B\mathcal{S}(\mathcal{P}^A) - N_A\mathcal{S}(\mathcal{P}^B)$. From these, we obtain sample mean $\mu(\mathcal{P}) = \frac{1}{|\mathcal{P}|}\mathcal{S}(\mathcal{P})$ and covariance $\Sigma(\mathcal{P}) = \frac{1}{|\mathcal{P}|-1}\mathcal{S}^2(\mathcal{P}) - \mu\mu^T$.

Careful treatment of the numerical stability is required when utilizing one-pass schemes for calculating the sample covariance (Chan et al., 1979). We require a minimum sample size of $|\mathcal{P}| \geq 10$ to create surfels and stop incorporating new data points if $|\mathcal{P}| \geq 10,000$[1]. The discretization of disparity and color produced by the RGB-D sensor may cause degenerate sample covariances, which

[1]Using double precision (machine epsilon $2.2 \cdot 10^{-16}$) and assuming a minimum standard

we robustly detect by thresholding the determinant of the covariance at a small constant.

The use of an update scheme allows for an efficient incremental update of the map. In the simplest implementation, each point is added individually to the tree: Starting at the root node, the point's statistics is recursively added to the nodes that contain the point in their volume.

Adding each point individually is, however, not the most efficient way to generate the map. Instead, we exploit that by the projective nature of the camera, neighboring pixels in the image project to nearby points on the sampled 3D surface—up to occlusion effects. This means that neighbors in the image are likely to belong to the same octree nodes. In effect, the size of the octree is significantly reduced and the leaf nodes subsume local patches in the image (see top-right of Fig. 2.7). Through the distance-dependent resolution limit, patch-size does not decrease with distance to the sensor but even increases. We exploit these properties and scan the image to aggregate the sufficient statistics of contiguous image regions that belong to the same octree node. This measurement aggregation allows to construct the map with only several thousand insertions of node aggregates for a 640×480 image in contrast to 307,200 point insertions.

After the image content has been incorporated into the representation, we precompute mean, covariance, surface normals, and shape-texture features.

2.2.4. Handling of Image and Virtual Borders

Special care must be taken at the borders of the image and at virtual borders where background is occluded (see Fig. 2.10). Nodes that receive such border points only partially observe the underlying surface structure. When updated with these partial measurements, the true surfel distribution is distorted towards the visible points. In order to avoid this, we determine such nodes by scanning efficiently through the image, and neglect them.

Conversely, foreground depth edges describe contours of measured surfaces. We thus mark surfels as belonging to a contour if they receive foreground points at depth discontinuities (example contours illustrated in Fig. 2.10).

2.3. Experiments

MRSMaps are designed as concise representations of RGB-D images as well as of maps that aggregate many images from various view-points. In the subsequent experiments, we demonstrate run-time and memory requirements of MRSMaps.

deviation of 10^{-4} in $\mathcal{P}$, and reasonable map sizes (maximal radius smaller than 10^2 m), we obtain a theoretical bound for the relative accuracy of the covariance entries in the order of 10^{-6} at 10^4 samples (Chan and Lewis, 1979). More accurate but slower two-pass schemes could be used for extremely large map or sample sizes, or smaller noise.

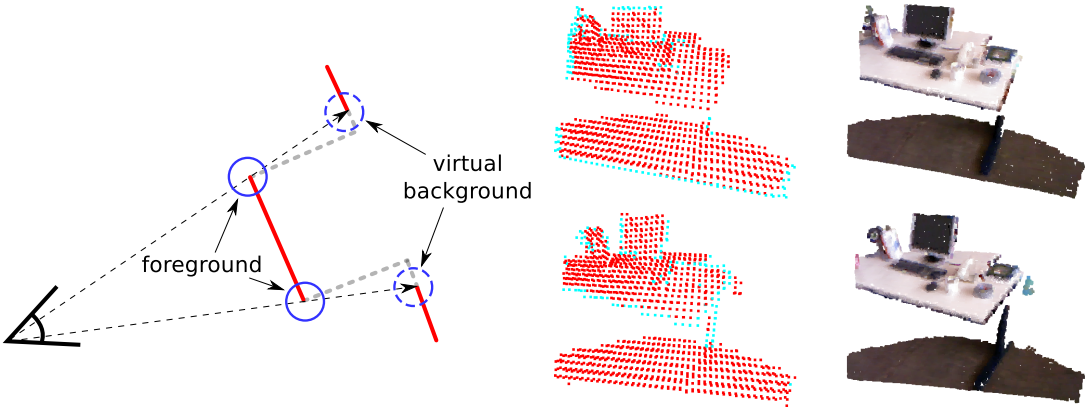

Figure 2.10.: Left: 2D illustration of occlusion types. We detect surfels that receive background and foreground points at depth discontinuities. The visibility of structure in the background changes with the view-point and is not consistent under view-point changes. Right, top: found virtual background border surfels (cyan). Right, bottom: foreground border surfels (cyan).

We utilize RGB-D sequences of the public RGB-D bechmark dataset (Sturm et al., 2012). The dataset contains RGB-D image sequences with ground truth information for the camera pose which has been measured using an external optical motion capture system. We use full 640×480 VGA resolution images and set the maximum resolution of our maps to $(0.0125\,\mathrm{m})^{-1}$ throughout the experiments which is a reasonable lower limit with respect to the minimum measurement range of the sensor (ca. 0.4 m) at a distance dependency of $\lambda_\rho = 0.02$. The experiments have been conducted on a consumer-grade PC with an Intel Core i7-4770K QuadCore CPU with a maximum clock speed of 3.50 GHz.

2.3.1. Single RGB-D Image Aggregation

The RGB-D benchmark dataset contains 47 sequences with a large variety in scene content. In some sequences, the camera is swept in-hand through office environments in various distances to surfaces. Other sequences observe mainly distant parts of a large open indoor environment. There are also sequences, in which the camera is attached on a mobile robot observing close-range and large indoor scenes from a low horizontal perspective near the ground. To obtain measurements in diverse scenes, we processed all the sequences contained in the RGB-D benchmark by constructing a MRSMap for each RGB-D image and measuring run-time and memory consumption.

Fig. 2.11 depicts the dependency of the MRSMap size in terms of nodes or voxels on the median depth in an image. Map size exhibits inverse quadratic relation to median depth which is indicated by a curve $n(z) := a\frac{1}{(z-b)^2} + c$ fitted

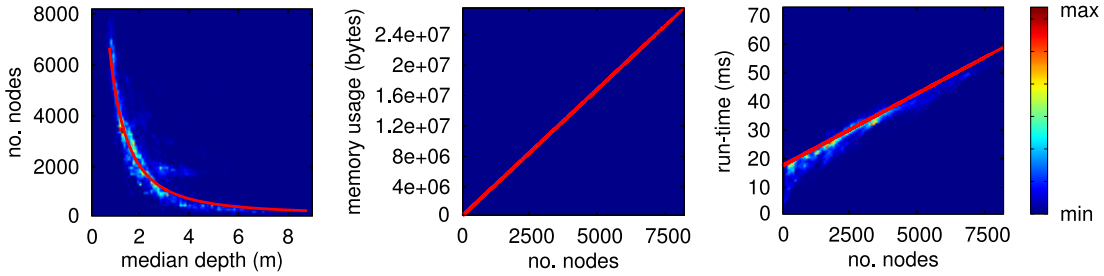

Figure 2.11.: Properties of MRSMap aggregation from single RGB-D images. We show histograms and function fits (red curves) over all sequences of the RGB-D benchmark dataset (Sturm et al., 2012). Left: number of nodes vs. median depth in image. Center: memory usage vs. number of nodes. Right: run-time vs. number of nodes.

to the local median of the points. The local median has been determined from points within a range of 0.1 m depth. A median of 2,368 and up to 8,189 nodes are instantiated, subsuming the 307,200 image pixels into 2 magnitudes less elements.

We can also see from Fig. 2.11, that memory consumption is linear in the number of nodes. Here, we fit a linear line to the acquired samples directly. We measured the memory claimed for tree structure, voxel properties, surfels, shape-texture descriptors, and node neighborhood pointers, which is 3,358 Bytes per node at double precision for six view directions. If only a single view direction is maintained in the nodes, we can safe 2,515 Bytes to reduce node size to 843 Bytes. A map with six view directions requires ca. 27.5 MB for 8,189 nodes. With a single view direction only about 6.9 MB are used, which is about 5.6 times larger than the 640×480 RGB-D image itself (ca. 1.2 MB) if it is stored with 2 Bytes for Bayer-pattern encoding of RGB and 2 Bytes for disparity at each pixel. We primarily design MRSMaps as a representation for image registration and for aggregating RGB-D measurements from multiple view-points. High memory efficiency is traded for run-time efficiency during registration for which map content such as surfels, shape-texture descriptors, and node neighborhood should be precalculated to gain significant speed ups.

The overall run-time to aggregate a MRSMap from an image scales approximately linearly with the number of nodes in the map (see Fig. 2.11). The timing includes to mark foreground and background borders, and to precompute surfels, node neighborhood, and shape-texture descriptors. The median overall run-time in all 47 sequences is 16.5 ms, while we measure 43.2 ms at maximum. Most of the individual processing steps such as tree construction and incorporation of sufficient statistics, determination of node neighborhood, evaluation of surfels, and calculation of the shape-texture descriptor also depend approximately lin-

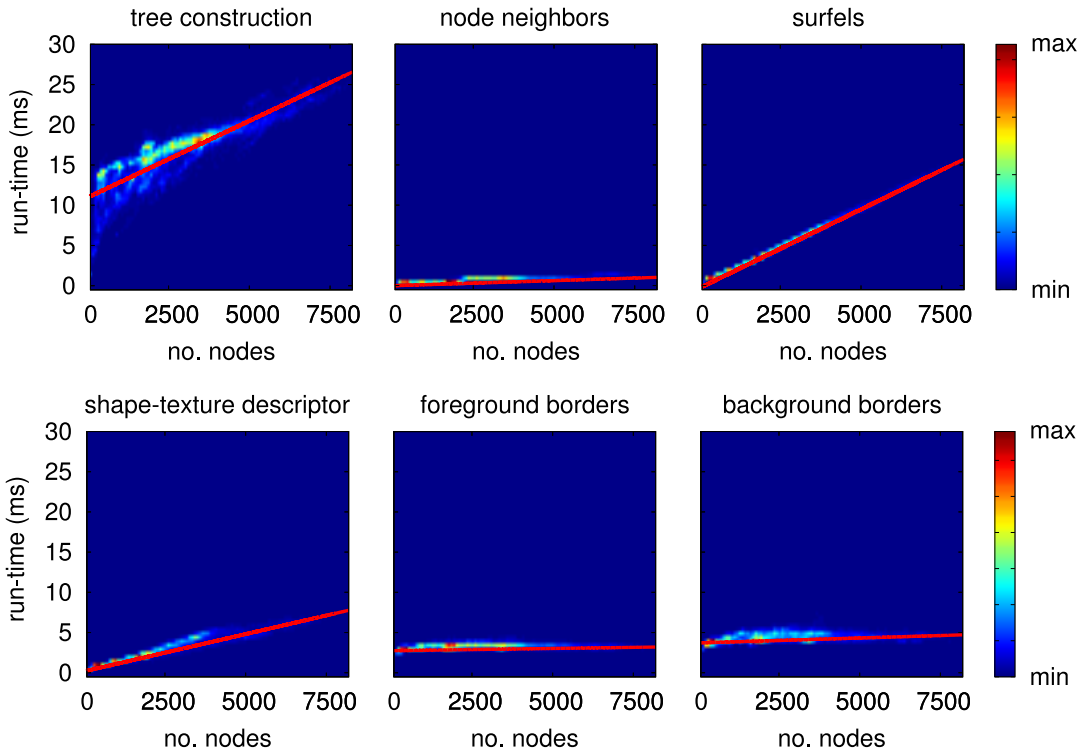

Figure 2.12.: Run-time of individual stages of MRSMap aggregation wrt. the number of nodes. We show histograms and linear function fits (red curves) over all sequences of the RGB-D benchmark dataset (Sturm et al., 2012). Top left: tree construction. Top center: node neighborhood precalculation. Top right: surfel evaluation (means, covariances, normals). Bottom left: shape-texture descriptor calculation. Bottom center: foreground border search. Bottom right: background (virtual) border search.

early on the number of nodes (see Fig. 2.12). Searching for fore- and background borders in the image naturally takes almost constant time with respect to the number of nodes.

2.3.2. Multi-View Map Aggregation

The results in Fig. 2.13 on three sequences of the RGB-D benchmark dataset demonstrate that MRSMaps efficiently store RGB-D sequences in multi-view maps. In the freiburg2_desk sequence, the camera is moved in-hand on a circle pointing inwards onto a cluttered table scene. We used the ground-truth pose available to integrate the RGB-D images into a single MRSMap. The increase in the number of nodes per iteration naturally depends on the degree of novelty of the viewed scene content. After the aggregation of 2,111 RGB-D images, the

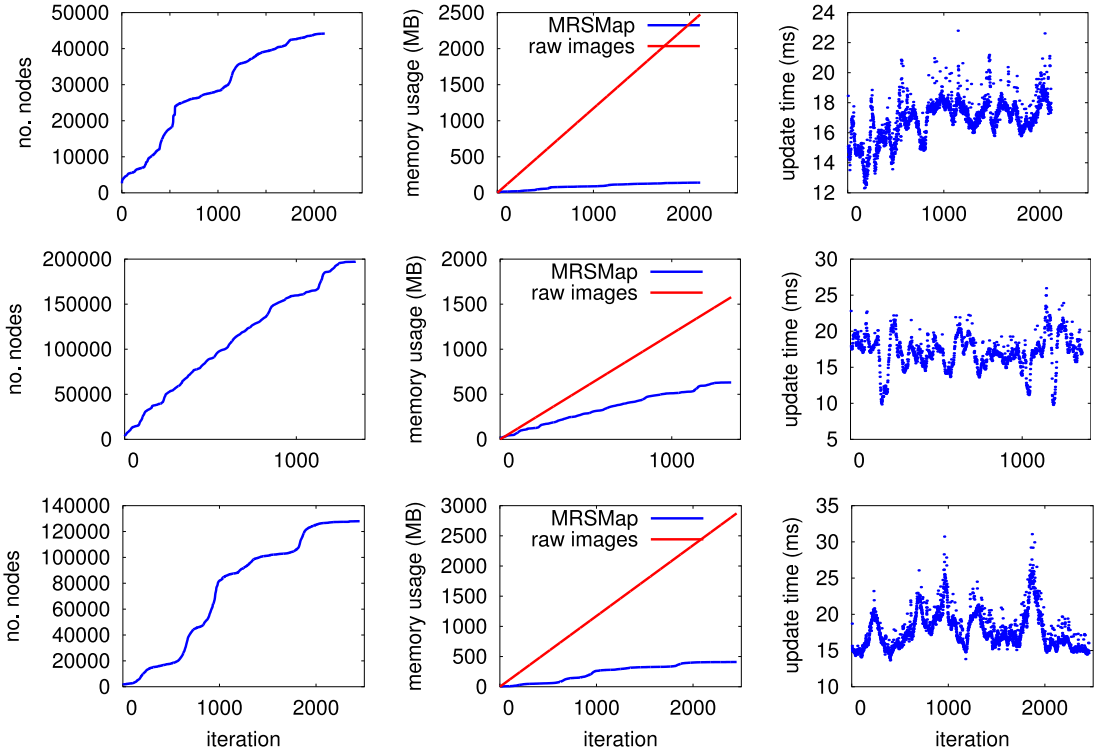

Figure 2.13.: Properties of MRSMap aggregation during incremental mapping in the freiburg2_desk (top), the freiburg1_room (center), and the freiburg3_long_office_household (bottom) sequence. Left: number of nodes. Center: memory usage. Right: run-time to update tree structure and sufficient statistics.

map contains 44,174 octree nodes and uses only 141.46 MB of memory compared to the 2,473.8 MB required to store the original 2,111 RGB-D images at 640×480 resolution. MRSMaps achieve a compression ratio of about 17.5 on this sequence. The unsteady evolution of the number of nodes is explained by alternating phases in which new scene content is observed and old parts are seen again. Remarkably, run-time for tree construction and incorporation of sufficient statistics only slowly increases with the number of nodes that are already contained in the map. It keeps below 22.8 ms throughout the sequence.

For the freiburg3_long_office_household sequence, our approach shows similar properties like for the freiburg2_desk sequence. The camera also moves in a circle around a table-top scene, mostly pointing inwards onto the tables. On this sequence, 2,451 images are processed with a total size of ca. 2,872.3 MB. The MRSMap contains 127,974 nodes in the end and utilizes 409.8 MB. The update time reaches 31.1 ms at maximum and varies around a median of 17.2 ms throughout the sequence. It indicates peaks in phases in which the number of nodes increases quickly.

Camera movement in the freiburg1_room sequence is qualitatively different to the previous two sequences. Here, the camera sweeps through an office and observes the scene content which is mostly placed near the walls from the inside of the room. During the most part of this sequence the evolution of the number of nodes is nearly linear since the camera does not rest on or reobserve previously mapped scene content. At the end of the sequence after about 900 iterations, when the camera moves to area seen at the beginning of the sequence, the increase in the number of nodes is less strong. The final map consists of 197,090 nodes, utilizing 631.2 MB of memory which corresponds to a compression ratio of approx. 2.5 towards the original images (1,577.3 MB).

2.4. Related Work

Storing raw 3D point clouds (Rusu et al., 2008) that have been acquired with 3D laser scanners or RGB-D cameras quickly becomes demanding in terms of memory consumption. Also, neighborhood look-ups are frequently part of algorithms that utilize the map representation, hence, the representation should support these look-ups efficiently. Elseberg et al. (2013) propose an octree implementation that can store point clouds of one billion points at 8 GB of RAM. Several compressive and more memory-efficient map representation have been studied. They typically are designed for visualization or autonomous robot navigation.

Vaskevicius et al. (2010) extract planar surface patches and demonstrate lower memory consumption than point- or voxel-based map representations. However, planarity is a strong assumption on the structure of the environment. In mobile robotics, occupancy grid mapping is a widely used technique for representing an environment in 2D (Thrun, 2002). This concept does not, however, directly transfer to modeling the environment in a 3D grid due to its high memory usage. Hornung et al. (2013) propose OctoMap, a framework that implements 3D occupancy mapping in octrees. Each leaf in the tree stores information about the cell being occupied, free, or unknown. The maximum resolution is not adapted to the measurement characteristics of the sensor, but every measurement is represented up to a common maximum resolution. To incorporate the modeling of unknown and free space, raycasting is performed for each measurement. Ryde and Hu (2010) represent occupied voxels at multiple resolutions in lists. This approach can be seen as an alternative implementation of octrees without making the tree structure explicit. In (Ryde and Corso, 2012) the authors adapt counting bloom filters, an efficient hashing function, to store the occupied voxels in hash maps. Our representation respects sensor error characteristics by choosing an adequate maximum resolution at each RGB-D image pixel. The measured surface in a voxel is modeled by its mean and covariance and hence provides a continuous surface representation as opposed to the discrete binary

occupancy information. Furthermore, we model six view-directions onto surfaces in each voxel. MRSMaps are specifically designed for efficient aggregation and registration.

For outdoor terrain modeling, elevation maps are frequently used (Herbert et al., 1989; Pfaff et al., 2007). A 2D grid is layed over the environment and the height of the surface is stored in each grid cell. Multi-level surface maps (Triebel et al., 2006) extend this 2.5 dimensional representation by maintaining multiple surface heights in each grid cell.

Krainin et al. (2011) model surfaces as a set of local RGB-D patches (also coined surfels). Signed distance functions (Curless and Levoy, 1996) are an alternative way to represent surfaces in voxel grids. They have been applied as map representation for RGB-D mapping in KinectFusion Newcombe et al. (2011a). Similar to our method, the 3D normal distribution transform (3D-NDT) represents 3D laser scans as Gaussian distributions in voxels of a 3D grid (Magnusson et al., 2007).

There are also extensions to model the RGB color of points and to utilize color for registration (Huhle et al., 2008). The authors propose to represent the spatial distibution of color within a voxel as a Gaussian mixture model. We propose a highly efficient image aggregation technique for point clouds that are organized in an image structure. It is specifically suited for RGB-D images. The 3D-NDT does not restrict the maximum resolution at a pixel to consider measurement characteristics. Furthermore, we rapidly extract local shape-texture descriptors from the map to improve the basin of convergence of our registration method. Contours and multiple view-points are also not included in the 3D-NDT.

2.5. Summary

In this chapter, we proposed MRSMaps, a 3D multi-resolution representation of RGB-D images and measurements from multiple view-points. The primary purpose of our method is to provide a representation suitable for rapid registration. Our maps incorporate typical measurement characteristics of RGB-D sensors. Since measurement noise, sampling density, and discretization effects depend quadratically on depth as well as the distance from the optical center on the image plane, we adapt the maximum resolution at each measurement to its squared distance from the sensor. In effect, the maps exhibit local multi-resolution structure.

We devised techniques to efficiently aggregate maps from images. We extract local rotation- and approximately illumination-invariant shape-texture descriptors that are intended to judge the correspondence of surfels between MRSMaps. In our evaluation, we demonstrated run-time efficiency and memory requirements of our implementation for maps from single images as well as for maps that incorporate images from multiple view-points. While a map of a single

RGB-D image requires more memory than the original image, the fusion of multiple views in a single MRSMap is more memory-efficient than storing the individual images. In MRSMaps, we trade off memory efficiency with run-time efficiency and representation accuracy for registration purposes by preserving a high degree of surface detail and precalculating features such as voxel neighborhood, surface normals, and shape-texture descriptors.

In future work, we will evaluate the use of hash tables instead of a pointer-based implementation of octrees to further increase efficiency. An implementation on GPU could also further improve run-time.

The applicability of MRSMaps is not restricted to RGB-D images. The measurement principle of rotating 2D laser scanners also produces point clouds that are organized image-like by stacking the individual 2D scan lines. In contrast to the dependency on the squared distance from the sensor in RGB-D images, the accuracy of 2D laser measurements is approximately linear in the distance. In recent work, we applied MRSMaps for 3D laser-based mapping and localization in rough terrain with mobile robots (Schadler et al., 2013).

3. Rigid Registration

Many perceptual abilities involve the alignment of measurements, e.g., to determine the ego-motion of a camera, to track objects, or to observe changes in a scene. We begin our investigation of RGB-D registration methods by devising an efficient registration algorithm that estimates motion between MRSMaps under the assumption that the observed scene remains rigid. In experiments, we demonstrate that our approach performs accurately, robustly, and is computationally efficient to run on CPU.

3.1. Background

Our registration approaches are formulated as optimization problems of non-linear functions. In the following, we will introduce basic concepts and notation.

3.1.1. Non-Linear Function Optimization

Let $f : \mathbb{R}^N \to \mathbb{R}$ be a real-valued, twice Lipschitz continuously differentiable function $f(x)$ in N real variables. We aim at finding a minimizer $\widehat{x}$ at the minimum of this function,

$$\widehat{x} = \arg\min_x f(x). \tag{3.1}$$

At $\widehat{x}$, f attains its minimum. We also call f the objective function.

If f is non-linear and non-convex, several local minima may exist. We therefore define the concept of a local minimizer $\widehat{x}$ to yield a minimum of f in a local neighborhood, i.e.,

$$\exists \epsilon \in \mathbb{R} : \forall x \in \left\{ x' \in \mathbb{R}^N \mid \left\| x' - \widehat{x} \right\| < \epsilon \right\} : f(x) \geq f(\widehat{x}). \tag{3.2}$$

We say that f attains a local minimum at $\widehat{x}$.

Since f is twice continuously differentiable, we can find first- and second-order derivatives,

$$\nabla f := \left(\frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_N} \right),$$
$$\nabla^2 f := \begin{pmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_N \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_N \partial x_N} \end{pmatrix}, \tag{3.3}$$

where we denote ∇f as the gradient or Jacobian and $\nabla^2 f$ as the Hessian of f. The Hessian is symmetric, i.e., $\left(\nabla^2 f\right)_{ij} = \left(\nabla^2 f\right)_{ji}$.

Taylor-expansion yields first- (linear) and second-order (quadratic) approximations to f,

$$f(x) \approx \widetilde{f}(x) = f(x') + \nabla f(x')\,(x - x'), \tag{3.4}$$

$$f(x) \approx \widetilde{f}(x) = f(x') + \nabla f(x')\,(x - x') + \frac{1}{2}(x - x')^T \nabla^2 f(x')\,(x - x'), \tag{3.5}$$

which we will utilize in iterative minimization schemes.

Global convergence of an optimization method means that it will find a minimum from any state. Local convergence restricts the basin of convergence to a local neighborhood around the minimum $\widehat{x}$, i.e.

$$\exists \epsilon > 0, \epsilon \in \mathbb{R} : \exists \delta \in (0,1) \subset \mathbb{R} : \forall i \in \mathbb{N}^+ : \\ \|\widehat{x} - x_0\| \leq \epsilon \Rightarrow (f(x_i) - f(\widehat{x})) \leq \delta\,(f(x_{i-1}) - f(\widehat{x}))\,. \tag{3.6}$$

3.1.1.1. Gradient Descent Methods

Gradient descent methods utilize the linear approximation to f in Eq. (3.4) to determine several steps towards a local minimum. The simplest gradient descent approach is the method of steepest descent (Kelley, 1999),

$$x_i = x_{i-1} - \lambda \nabla f(x_{i-1}) \tag{3.7}$$

in which we take steps $-\lambda \nabla f(x_{i-1})$ towards a minimum.

Due to the linear approximation of f, the choice of λ determines if x_i actually decreases f towards $f(x_{i-1})$. An important part of steepest descent algorithms is hence to control λ. Typically, λ is gradually adjusted in a line-search in which the result of the steepest descent iteration is tested to get closer to a local minimum in f along the line defined by the linear approximation. If λ is set such that f attains a local minimum along the line, the line-search is called exact.

The steepest descent algorithm with exact line-search provides linear convergence rate, i.e., there is a constant $\delta \in (0,1) \subset \mathbb{R}$ such that for all i

$$(f(x_i) - f(\widehat{x})) \leq \delta\,(f(x_{i-1}) - f(\widehat{x}))\,. \tag{3.8}$$

It can be shown (e.g., Nesterov, 2004), that the convergence constant δ depends on the minimum and maximum eigenvalues of the Hessian $\nabla f(\widehat{x})$ at the local minimum $\widehat{x}$ of f. Steepest descent typically becomes slow close to the minimum and can exhibit slow convergence along valleys of the objective function. However, it has global convergence guarantees.

Several gradient descent methods exist that improve convergence speed. For general non-linear functions, the conjugate gradient descent method may yield quadratic convergence rate, i.e., for all i,

$$(f(x_i) - f(\widehat{x})) \leq \delta \left(f(x_{i-1}) - f(\widehat{x})\right)^2, \tag{3.9}$$

and linear convergence in the worst case (Hager and Zhang, 2006).

3.1.1.2. Newton's Method

If we use the local quadratic approximation in Eq. (3.5), we can set the derivative $\nabla \widetilde{f}(x) = 0$ to zero to obtain

$$\nabla f(x') + \nabla^2 f(x')\,(x - x') = 0. \tag{3.10}$$

This yields the second-order update scheme

$$x_i = x_{i-1} - \lambda \left(\nabla^2 f(x_{i-1})\right)^{-1} \nabla f(x_{i-1}), \tag{3.11}$$

which is known as Newton's method. Again, the step-length λ has to be chosen appropriately. For convergence towards a local minimum at $\widehat{x}$, also the sufficient condition $x \left(\nabla^2 f(x_i)\right) x > 0$ must hold that the Hessian in each iteration is positive definite.

Newton's method has quadratic convergence rate. However, convergence can only be guaranteed locally, i.e., for sufficiently small deviations from the minimum for which the second-order approximation does not yield a maximum or saddle point.

3.1.2. Non-Linear Least Squares Optimization

In non-linear least squares optimization we can write the objective function in the form

$$f(x) = \frac{1}{2} e^T(x) W e(x), \tag{3.12}$$

where $e(x) := h(x) - z \in \mathbb{R}^M$ is a vector of M residuals between $h(x)$ and target variables $z \in \mathbb{R}^M$, e and h are non-linear Lipschitz continuously differentiable functions, and $W \in \mathbb{R}^{N \times N}$ is a weighting matrix.

Jacobian and Hessian of f are

$$\nabla f(x) = \nabla e^T(x) W e(x), \tag{3.13}$$

$$\nabla^2 f(x) = \nabla e^T(x) W \nabla e(x) + \nabla^2 e^T(x) W e(x). \tag{3.14}$$

The following iterative optimization methods utilize this special form to obtain efficient algorithms.

3.1.2.1. The Gauss-Newton Method

The Gauss-Newton method exploits the local linearization

$$f(x) \approx \widetilde{f}(x) = \frac{1}{2}\widetilde{e}^T(x) W \widetilde{e}(x), \tag{3.15}$$

of the residuals e around x'

$$e(x) \approx \widetilde{e}(x) = e(x') + \nabla e(x')\,(x - x') \tag{3.16}$$

to yield

$$\begin{aligned}\widetilde{f}(x) = &\frac{1}{2} e^T(x') W e(x') + \frac{1}{2} e^T(x') W \nabla e(x')\,(x - x') \\ &+ \frac{1}{2}(x - x')^T \nabla e^T(x') W e(x') + \frac{1}{2}(x - x')^T \nabla e^T(x') W \nabla e(x')\,(x - x').\end{aligned} \tag{3.17}$$

Taking the derivative with respect to x we obtain

$$\nabla \widetilde{f}(x) = \nabla e^T(x') W e(x') + \nabla e^T(x') W \nabla e(x')\,(x - x'). \tag{3.18}$$

Setting $\nabla \widetilde{f}(x) = 0$ we finally arrive at

$$x = x' - \left(\nabla e^T(x') W \nabla e(x')\right)^{-1} \nabla e^T(x') W e(x'). \tag{3.19}$$

The Gauss-Newton method iteratively updates the estimate through

$$x_i = x_{i-1} - \left(\nabla e^T(x_{i-1}) W \nabla e(x_{i-1})\right)^{-1} \nabla e^T(x_{i-1}) W e(x_{i-1}). \tag{3.20}$$

Due to the linear approximation in Eq. (3.15), taking the full update may diverge. Damped Gauss-Newton methods only use a fraction of the update,

$$x_i = x_{i-1} - \lambda \left(\nabla e^T(x_{i-1}) W \nabla e(x_{i-1})\right)^{-1} \nabla e^T(x_{i-1}) W e(x_{i-1}), \tag{3.21}$$

for which line-search methods are suitable to adjust λ.

The Gauss-Newton method can be shown to demonstrate quadratic convergence rate. Convergence can be proven locally (Kelley, 1999).

3.1.2.2. The Levenberg-Marquardt Method

The Levenberg-Marquardt (LM) method can be seen as a hybrid of the steepest descent and the damped Gauss-Newton method. For convergence of the damped Gauss-Newton method, the matrix $\nabla e^T(x_{i-1})W\nabla e(x_{i-1})$ must have full column rank, be uniformly bounded, and be well-conditioned (Kelley, 1999). The LM method provides a method with better global convergence properties that utilizes efficient damped Gauss-Newton steps.

By formulating

$$x_i = x_{i-1} - \left(\nabla e^T(x_{i-1})W\nabla e(x_{i-1}) + \lambda_i I\right)^{-1} \nabla e^T(x_{i-1})We(x_{i-1}) \tag{3.22}$$

parameter λ_i adjusts between steepest descent and Gauss-Newton iterations. Intuitively, if λ_i is large, the method will take a small step into mainly the steepest descent direction

$$x_i \approx x_{i-1} - \frac{1}{\lambda_i}\nabla e^T(x_{i-1})We(x_{i-1}). \tag{3.23}$$

For vanishing λ_i, we have Gauss-Newton steps

$$x_i \approx x_{i-1} - \left(\nabla e^T(x_{i-1})W\nabla e(x_{i-1})\right)^{-1} \nabla e^T(x_{i-1})We(x_{i-1}). \tag{3.24}$$

Initially,

$$\lambda_1 = \tau \max\left\{\left(\nabla e^T(x_0)W\nabla e(x_0)\right)_{ii}\right\} \tag{3.25}$$

is set to a trade-off between steepest descent and Gauss-Newton (Madsen et al., 2004) with free parameter τ.

In each iteration i, the new estimate x_i is kept, if the gain ratio

$$\rho := \frac{f(x_{i-1}) - f(x_i)}{\widetilde{f}(x_{i-1}) - \widetilde{f}(x_i)} \tag{3.26}$$

is larger than zero and indicates that $\widetilde{f}(x_i)$ well approximates $f(x_i)$ (Madsen et al., 2004). If so, we decrease $\lambda_{i+1} = \lambda_i \max\left\{\frac{1}{3}, 1-(2\rho-1)^3\right\}$ to perform steps closer to Gauss-Newton. Otherwise, we set back $x_i = x_{i-1}$ and update $\lambda_{i+1} = \lambda_i\nu_i$ and $\nu_{i+1} = 2\nu_i$ to obtain steps closer to the steepest descent direction. In the initial iteration, or if the LM-step is accepted, we reset the tracking parameter to $\nu = 2$.

3.1.2.3. Multiple Objectives

The non-linear least squares optimization can be extended to optimize for multiple objectives in a weighted sum

$$f(x) = \frac{1}{2}\sum_{j=1}^{M} \mu_j e_j^T(x)W_j e_j(x), \tag{3.27}$$

with weights μ_j and residuals $e_j(x) := h_j(x) - z_j$. Note that here the residuals $e_j(x)$ may differ in dimensionality and in functions h_j. We observe that the residuals and weight matrices of the individual objectives can be stacked into a single combined residual and block-diagonal weighting matrix

$$f(x) = \frac{1}{2} e^T(x) W e(x), \tag{3.28}$$

with

$$e(x) := \begin{pmatrix} \sqrt{\mu_1}\, e_1(x) \\ \vdots \\ \sqrt{\mu_M}\, e_M(x) \end{pmatrix} \tag{3.29}$$

and

$$W := \operatorname{diag}(W_1, \ldots, W_M). \tag{3.30}$$

3.2. Efficient Rigid Registration of Multi-Resolution Surfel Maps

We utilize MRSMaps as a compact multi-resolution image representation for our rigid registration approach. We have seen in Ch. 2 that MRSMaps can be constructed efficiently which is an important prerequisite for fast registration. In this section, we devise efficient means to align two MRSMaps that further exploit the multi-resolution structure of the maps. Our approach recovers the 6-DoF rigid-body motion $x \in \mathbb{SE}(3)$ between scene and model map in the Special Euclidean Group $\mathbb{SE}(3)$.

3.2.1. Multi-Resolution Surfel Association

Starting from an initial coarse guess of the alignment, surfels are associated between both maps from coarse to fine, always using the finest common resolution possible. Since associations between surfels are not known beforehand, they need to be estimated in the registration process. Registration hence performs two alternating steps: *surfel association* and *surfel alignment* which both need to be addressed efficiently.

Our aim is to find for each surfel $s_s \in m_s$ in the scene map m_s a corresponding surfel $s_m \in m_m$ in the model map m_m. We start iterating through the surfels $s_s \in m_s$ from fine to coarse resolutions, and consider surfels in all view directions. In order to choose the finest resolution possible between both maps, we skip surfels in nodes for which a surfel association exists on finer resolutions already. This way, we save redundant matches on coarse resolutions.

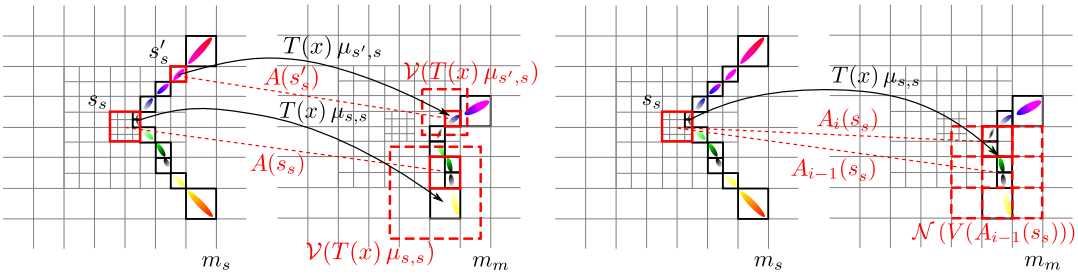

Figure 3.1.: Multi-resolution surfel association between scene $s_s \in m_s$ and model surfels $s_m \in m_m$. Left: Associations $A(s)$ are found for scene surfels s_s, s'_s by finding the best match in a cubic query volume $\mathcal{V}(T(x)\,\mu)$ centered at the scene surfel mean position in the model map frame under the current pose estimate x. Right: If an association is known from the previous iteration, we search for best matches among the direct neighbors $\mathcal{N}(V(A_{i-1}(s_s)))$ in the voxel grid.

If a surfel s_s is considered for association, we query a set of potential matches $\widehat{A}(s_s) = \{s_m \in m_m \mid \text{close}(s_m, s_s, x)\}$ and establish the best match,

$$A(s_s) = \underset{s_m \in \widehat{\mathcal{A}}(s_s)}{\arg\min} \{\text{assoc-dist}(s_s, s_m, x)\}, \tag{3.31}$$

where $\text{close}(s_s, s_m, x)$ indicates neighborhood of surfels, and $\text{assoc-dist}(s_s, s_m, x)$ is a distance measure. The latter incorporates the Euclidean distance between the surfel means and the similarity in shape-texture descriptor,

$$\text{assoc-dist}(s_s, s_m, x) = \left\| \mu^p_{s,m} - T(x)\,\mu^p_{s,s} \right\|_2 \cdot \|h_{s,s} - h_{s,m}\|_2 . \tag{3.32}$$

We denote the set of surfel associations by

$$\mathcal{A} = \{(s_s, s_m) \in m_s \times m_m \mid s_m = A(s_s)\}. \tag{3.33}$$

We use two ways to define the association neighborhood of a scene surfel s_s. Fig. 3.1 illustrates these association strategies. Let $v(s)$ denote the view direction of a surfel s. Initially, if the surfel has not been associated in a previous iteration yet, we find all surfels s_m in the model map, whose voxels $V(s_m)$ are within a cubic volume $\mathcal{V}(T(x)\,\mu_{s,s})$. For each found surfel, we check if its mean is within a specific spatial and color distance, if it is similar in shape-texture descriptor, and if the view directions are compatible,

$$\text{close}(s_s, s_m, x) \Leftrightarrow V(s_m) \in \mathcal{V}(T(x)\,\mu_{s,s}) \;\wedge\; \text{compatible}(s_s, s_m, x), \tag{3.34}$$

where

$$\begin{aligned}\text{compatible}(s_s, s_m, x) \Leftrightarrow\; & \text{pos-compatible}(s_s, s_m, x) \;\wedge\; \text{col-compatible}(s_s, s_m) \\ & \wedge\; \text{descr-compatible}(s_s, s_m) \;\wedge\; \text{viewdir-compatible}(s_s, s_m, x),\end{aligned} \tag{3.35}$$

$$\text{pos-compatible}(s_s, s_m, x) \Leftrightarrow \left\| \mu^p_{s,m} - T(x)\,\mu^p_{s,s} \right\|_2 \leq \chi_p, \tag{3.36}$$

$$\begin{aligned}\text{col-compatible}(s_s, s_m) \Leftrightarrow \left|\mu^L_{s,s} - \mu^L_{s,m}\right| \leq \chi_L \\ \wedge\ \left|\mu^\alpha_{s,s} - \mu^\alpha_{s,m}\right| \leq \chi_\alpha\ \wedge\ \left|\mu^\beta_{s,s} - \mu^\beta_{s,m}\right| \leq \chi_\beta.\end{aligned} \tag{3.37}$$

$$\text{descr-compatible}(s_s, s_m) \Leftrightarrow \|h_{s,s} - h_{s,m}\|_2 \leq \chi_h, \tag{3.38}$$

$$\text{viewdir-compatible}(s_s, s_m, x) \Leftrightarrow (T(x)\,v(s_s))^T\, v(s_m) \geq \chi_v, \tag{3.39}$$

The cubic query volume is axis-aligned with the model map reference frame. Its extents are multiples of the resolution $\rho(s_s)$ of the query surfel s_s. Such cubic volume queries are efficiently implemented in octree data structures.

If there is an association from previous iterations, we exploit it to find new associations more efficiently. We assume that the pose change determined in the previous iteration $i-1$ moves the scene surfel in the model map frame only by a small amount. We then try to associate the scene surfel s_s with the previously associated surfel $A_{i-1}(s_s)$ or its direct neighbors in the voxel grid, i.e.,

$$\text{close}(s_s, s_m, x) \Leftrightarrow V(s_m) \in \mathcal{N}(V(A_{i-1}(s_s)))\ \wedge\ \text{compatible}(s_s, s_m, x), \tag{3.40}$$

where $\mathcal{N}(V(A_{i-1}(s_s)))$ is the set of surfels contained in the voxel $V(A_{i-1}(s_s))$ and its direct neighbors. Since we precalculate the 26-neighborhood of each node, this look-up amounts to only constant time. Note that in the case that no compatible association can be found under the current pose estimate, i.e., $A_i(s_s) = \emptyset$, a volume query is used in the next iteration and a representative surfel could be associated on a coarser resolution instead.

We process resolutions from fine to coarse. For every resolution, all surfels are associated independently which allows the load within a resolution to be distributed over multiple CPU cores.

3.2.2. Pose Estimation

3.2.2.1. Observation Model

Our goal is to register a scene map m_s towards a model map m_m. We formulate our problem as finding the most likely pose x that maximizes the likelihood $p(m_s \mid x, m_m)$ of observing the scene in the model map. We choose to represent poses $x = (q,t)^T$ by translations $t \in \mathbb{R}^3$ and by unit quaternions $q \in \mathbb{H}$ for a compact representation of the rotational part without singularities.

We determine the observation likelihood between scene and model map as

$$p(m_s \mid x, m_m) = \prod_{(s_s, s_m) \in \mathcal{A}} p(s_s \mid x, s_m), \tag{3.41}$$

where $\mathcal{A}$ is the set of surfel associations between the maps. The observation likelihood of a surfel is defined in terms of deviation of the surfel mean from the mean of its association under the current pose estimate,

$$\begin{aligned} p(s_s \mid x, s_m) &= \mathcal{N}\left(d(s_s, s_m, x); 0, \Sigma(s_s, s_m, x)\right), \\ d(s_s, s_m, x) &:= \mu_{s,m} - T(x)\,\mu_{s,s}, \\ \Sigma(s_s, s_m, x) &:= \Sigma_{s,m} + R(q)\Sigma_{s,s}R(q)^T, \end{aligned} \tag{3.42}$$

where $R(q)$ is a rotation matrix that corresponds to q. We marginalize the surfel distributions for the spatial dimensions, since color would be meaningless far from the measured surface.

Due to the difference in view poses between images, the scene content may be discretized differently between the maps. We compensate for inaccuracies due to discretization effects through trilinear interpolation: Let $V(A(s_s))$ be the voxel that contains the associated surfel for the scene surfel s_s, and let $\mu'_{s,s} := T(x)\,\mu_{s,s}$. If $V(A(s_s))$ equals $V(\mu'_{s,s})$, i.e., the scene surfel s_s is associated with a model surfel s_m within the voxel at the projected position of the scene surfel, we determine a new mean $\overline{\mu}_{s,m}$ and covariance $\overline{\Sigma}_{s,m}$ from the surfels

$$\mathcal{S}_{\text{adj}}\left(\mu'_{s,s}\right) := \left\{ s_m \in m_m \mid V(s_m) \in \mathcal{N}_{\text{adj}}\left(\mu'_{s,s}\right) \right\} \tag{3.43}$$

in the eight adjacent voxels $\mathcal{N}_{\text{adj}}\left(\mu'_{s,s}\right)$ to $\mu'_{s,s}$:

$$\begin{aligned} \overline{\mu}_{s,m} &:= \frac{1}{\sum_s w(V(s), \mu'_{s,s})} \sum_{s \in \mathcal{S}_{\text{adj}}(\mu'_{s,s})} w(V(s), \mu'_{s,s})\,\mu_s \\ \overline{\Sigma}_{s,m} &:= \frac{1}{\sum_s w(V(s), \mu'_{s,s})^2} \sum_{s \in \mathcal{S}_{\text{adj}}(\mu'_{s,s})} w(V(s), \mu'_{s,s})^2\,\Sigma_s. \end{aligned} \tag{3.44}$$

The interpolation weight

$$w(V(s), \mu'_{s,s}) := \delta_x(V(s), \mu'_{s,s}) \cdot \delta_y(V(s), \mu'_{s,s}) \cdot \delta_z(V(s), \mu'_{s,s}) \tag{3.45}$$

is determined from the inverse displacements

$$\begin{aligned} \delta_x(V(s), \mu'_{s,s}) &= e_x^T(\rho(s)^{-1} - c(V(s)) + \mu'_{s,s}), \\ \delta_y(V(s), \mu'_{s,s}) &= e_y^T(\rho(s)^{-1} - c(V(s)) + \mu'_{s,s}), \\ \delta_z(V(s), \mu'_{s,s}) &= e_z^T(\rho(s)^{-1} - c(V(s)) + \mu'_{s,s}) \end{aligned} \tag{3.46}$$

along the base vectors e_x, e_y, e_z of the map frame, where $\rho(s)$ is the resolution of the surfel and $c(V(s))$ is the center position of voxel $V(s)$.

3.2.2.2. Pose Optimization

We optimize the logarithm of the observation likelihood (Eq. (3.41))

$$L(x) = \sum_{(s_s,s_m)\in\mathcal{A}} \ln(|\Sigma(s_s,s_m,x)|) + d^T(s_s,s_m,x)\Sigma^{-1}(s_s,s_m,x)d(s_s,s_m,x) \tag{3.47}$$

for the pose x in two stages: We apply fast approximate LM optimization to initialize fine registration using Newton's method.

Levenberg-Marquardt Optimization As detailed in Sec. 3.1.2.2, the LM method is suitable for weighted non-linear least squares problems of the form

$$\arg\max_x e^T(x)We(x), \tag{3.48}$$

where $e(x) = y - f(x)$ is a vector of residuals and W is a weighting matrix.

Let $\mathcal{A} = \left\{(s_s^1, s_m^1), \ldots, (s_s^{|\mathcal{A}|}, s_m^{|\mathcal{A}|})\right\}$ be the set of associated surfels. We stack the residuals for the associated surfels

$$e(x) = \begin{pmatrix} d(s_s^1, s_m^1, x) \\ \cdots \\ d(s_s^{|\mathcal{A}|}, s_m^{|\mathcal{A}|}, x) \end{pmatrix} \tag{3.49}$$

and neglect the effect of the pose on the covariance to obtain a constant block-diagonal weighting matrix

$$W(x) = \operatorname{diag}\left(w\left(s_s^1, s_m^1\right)\Sigma^{-1}\left(s_s^1, s_m^1, x\right), \ldots, w\left(s_s^{|\mathcal{A}|}, s_m^{|\mathcal{A}|}\right)\Sigma^{-1}\left(s_s^{|\mathcal{A}|}, s_m^{|\mathcal{A}|}, x\right)\right). \tag{3.50}$$

We weight each surfel association additionally with the similarity $w(s_s, s_m) := \chi_h - \|h(s_s) - h(s_m)\|_2$ of the shape-texture descriptors.

LM optimization now performs damped Gauss-Newton steps

$$x_{i+1} = x_i \oplus \left(J(x_i)^T W(x_i) J(x_i) + \lambda I\right)^{-1} J(x_i)^T W(x_i) e(x_i), \tag{3.51}$$

where

$$J(x) := \begin{pmatrix} J(s_s^1, s_m^1, x) \\ \cdots \\ J(s_s^{|\mathcal{A}|}, s_m^{|\mathcal{A}|}, x) \end{pmatrix} \tag{3.52}$$

is the Jacobian stacked from individual Jacobians $J(s_s, s_m, x) := \nabla T(x)\mu_{s,s}$ per surfel association, and λ is a damping parameter that is adapted by the LM method (see Sec. 3.1). The operator $x \oplus x'$ concatenates poses in $\mathbb{SE}(3)$ such that $T(x \oplus x') = T(x')T(x)$.

Due to the block-diagonal structure of $W(x)$, this update decomposes into simple sums over terms per association, i.e.,

$$J(x)^T W(x) J(x) = \sum_{(s_s, s_m) \in \mathcal{A}} J(s_s, s_m, x)^T W(s_s, s_m, x) J(s_s, s_m, x) \tag{3.53}$$

and

$$J(x)^T W(x) e(x) = \sum_{(s_s, s_m) \in \mathcal{A}} J(s_s, s_m, x)^T W(s_s, s_m, x) d(s_s, s_m, x). \tag{3.54}$$

During the LM optimization, we do not use trilinear interpolation of surfels. We stop iterating the LM method, if the pose still not changes after surfel association, or a maximum number of iterations is reached.

Optimization using Newton's Method We fine-tune the registration by applying Newton's method (see Sec.3.1.1.2) directly on the observation log-likelihood (Eq. (3.47))

$$x_{i+1} = x_i \oplus \left(- \left(\nabla^2 L(x_i) \right)^{-1} \left(\nabla L(x_i) \right) \right) \tag{3.55}$$

with trilinear interpolation and neglecting shape-texture descriptors. While Newton's method requires second-order derivatives, they can be efficiently calculated analytically due to the simple form of the observation log-likelihood.

Our combined approach typically converges within 10-20 iterations of LM and 5 iterations of Newton's method to a precise estimate. We parallelize the evaluation of the inner sum terms in LM and the first- and second-order derivatives in Newton's method which yields a significant speed-up on multi-core CPUs.

The normalization constraint on the quaternion part of the pose x requires special considerations for the optimization. We incorporate the normalization by only optimizing for a compact 3-dimensional quaternion representation that consists of the coefficients of the imaginary parts. The real part of the quaternion can be recovered from the normalization constraint and its initial sign before the optimization. This approach alone would only be valid for angular misalignments below 90° as of the zero-crossing of the real part. To allow for arbitrary angular misalignments, we compose the current pose estimate x from a constant part x' and a subsequent pose change Δx, i.e., $T(x) = T(\Delta x) T(x')$. In each iteration of LM or Newton's method, we set $x' = x$ and optimize for Δx instead.

3.2.2.3. Pose Uncertainty

Censi (2007) proposes a closed-form approximation to the covariance of the iterative closest points (ICP) pose estimate

$$\Sigma_x \approx \left(\frac{\partial^2 L}{\partial x^2} \right)^{-1} \left(\frac{\partial^2 L}{\partial z \partial x} \right) \Sigma_z \left(\frac{\partial^2 L}{\partial z \partial x} \right)^T \left(\frac{\partial^2 L}{\partial x^2} \right)^{-1}, \tag{3.56}$$

where $L(x, z)$ is the objective function, x is the pose estimate and z are the measurements used for the registration. It is an application of a general solution to the covariance of non-linear optimization problems. In this sense, the approach is also applicable to our setting.

Here, the objective function L is the observation log-likelihood in Eq. (3.47). The measurements z are stacked from surfels

$$\mathcal{S}_{\mathcal{A}} := \{s_s \in m_s \mid A(s_s) \neq \emptyset\} \cup \{s_m \in m_m \mid \exists s_s \in m_s : s_m = A(s_s)\} \tag{3.57}$$

that appear in associations. By letting $\mathcal{S}_{\mathcal{A}} = \left\{s_1, \ldots, s_{|\mathcal{S}|}\right\}$ we write

$$z := \left(\mu_{s,1}^T, \ldots, \mu_{s,|\mathcal{S}|}^T\right)^T. \tag{3.58}$$

The measurement covariance

$$\Sigma_z := \operatorname{diag}\left(\Sigma_{s,1}, \ldots, \Sigma_{s,|\mathcal{S}|}\right) \tag{3.59}$$

is a block-diagonal matrix composed from the surfel covariances.

Again we observe that due to the block-diagonal structure of Σ_z and the summation over surfel associations in the observation log-likelihood L, the calculation of the pose covariance decomposes into individual terms per surfel association

$$\Sigma_x \approx \sum_{(s_s, s_m) \in \mathcal{A}} \left(\frac{\partial^2 l}{\partial x^2}\right)^{-1} \left(\sum_{s \in \{s_s, s_m\}} \left(\frac{\partial^2 l}{\partial s \partial x}\right) \Sigma_s \left(\frac{\partial^2 l}{\partial s \partial x}\right)^T\right) \left(\frac{\partial^2 l}{\partial x^2}\right)^{-1}, \tag{3.60}$$

where $l(s_s, s_m, x) := \ln p(s_s, s_m, x)$. These terms can hence be calculated individually per surfel association which we exploit for parallel computation on multi-core CPUs.

The uncertainty of the pose estimate can also be obtained for results of the LM optimization. We may simply replace the Hessian $\frac{\partial^2 L}{\partial x^2}$ with its approximation $J^T(x)W(x)J(x)$ made by the LM method:

$$\Sigma_x \approx \left(J^T(x)W(x)J(x)\right)^{-1} \left(\frac{\partial^2 L}{\partial z \partial x}\right) \Sigma_z \left(\frac{\partial^2 L}{\partial z \partial x}\right)^T \left(J^T(x)W(x)J(x)\right)^{-1}. \tag{3.61}$$

The covariance of the pose estimate captures uncertainty along unobservable dimensions, for instance, if the maps view a planar surface.

3.2.2.4. Regularization

If prior knowledge about the camera motion is known, we can utilize it to further increase the robustness of the registration. The observation likelihood in Eq. (3.41) is augmented with a prior on the pose estimate,

$$p(x \mid m_s, m_m) = \eta p(m_s \mid x, m_m)\, p(x). \tag{3.62}$$

We model this prior $p(x) = \mathcal{N}(x; \mu_{x,0}, \Sigma_{x,0})$ normal distributed about a mean pose $\mu_{x,0}$ with covariance $\Sigma_{x,0}$. We then optimize the logarithm of Eq. (3.62),

$$L^{rgl}(x) = L(x) + \ln\left(\mathcal{N}(x; \mu_{x,0}, \Sigma_{x,0})\right). \tag{3.63}$$

In order to incorporate both objectives, we augment our pose optimization approach (Sec. 3.2.2.2) in two ways. First, we employ the multi-objective non-linear least squares framework in Sec. 3.1.2.3 to implement the pose prior in the LM optimization. We neglect the normalization of the normal distribution and add the residual $e_{prior}(x) := x - \mu_{x,0}$ with weighting matrix $W_{prior} := \Sigma_{x,0}^{-1}$. Fine tuning of the registration then performs Newton's method directly on Eq. (3.62). Here, the prior contributes additional terms to first- and second-order derivatives of the objective function.

A covariance estimate of the pose uncertainty can also be determined in the regularized case. From Eq. (3.62) we observe that each factor contributes a normal distributed estimate for the pose. Hence, we determine the regularized pose estimate $x \sim \mathcal{N}\left(\mu_x^{rgl}, \Sigma_x^{rgl}\right)$ as normal distributed with the mean resulting from the regularized registration, and the covariance

$$\Sigma_x^{rgl} \approx \left(\Sigma_x^{-1} + \Sigma_{x,0}^{-1}\right)^{-1}, \tag{3.64}$$

where Σ_x is the covariance estimate for the mean pose propagated from the measurements (see Sec. 3.2.2.3).

3.3. Experiments

We evaluate our rigid registration approach on the publicly available RGB-D benchmark dataset by Sturm et al. (2012) (see Sec. 2.3). As in Sec. 2.3, we process full resolution 640×480 images and set the maximum resolution of our maps to 0.0125 m at a distance dependency of $\lambda_\rho = 0.02$. For the assessment of run-time efficiency of our implementation, we carry out the experiments on an Intel Core i7-4770K QuadCore CPU at a maximum clock speed of 3.50 GHz.

We compare our method to other dense registration approaches, i.e., warp (Steinbruecker et al., 2011), generalized iterative closest points (GICP) (Segal et al., 2009), and 3D-NDT (Stoyanov et al., 2012). For our experiments, we used a reimplementation of warp contained in the OpenCV library with default settings but 14 iterations in the coarsest resolution and a maximum depth difference of 0.5 m. The maximum distance for matches in GICP has been chosen as $d_{\max} = 0.2\,m$. 3D-NDT performs NDT to NDT registration and has been configured to use the four scales 0.05 m, 0.1 m, 0.2 m, and 0.4 m. Higher resolutions were not possible due to memory limitations. With fovis (Huang et al., 2011), we also include a state-of-the-art sparse method that is based on matching interest points for comparison.

3.3.1. Evaluation Measure

The availability of ground-truth trajectories for the RGB-D benchmark sequences allows for quantifying the accuracy of registration estimates. The relative pose error (RPE) metric measures the pose difference of the relative ground truth motion and the registration estimate between two frames t and $t+\Delta$,

$$E_{\Delta,t} := \left(T_t^{-1} T_{t+\Delta}\right)^{-1} \left(\widehat{T}_t^{-1} \widehat{T}_{t+\Delta}\right) \tag{3.65}$$

with ground truth poses T_t, $T_{t+\Delta}$ and registration estimate poses $\widehat{T}_t$, $\widehat{T}_{t+\Delta}$ (Sturm et al., 2012). Time-sequential frame-to-frame registration can be evaluated by considering time differences of $\Delta = 1$. Since rotational errors also influence translational errors, we evaluate accuracy by translational error.

3.3.2. Accuracy

Tables 3.1 and 3.2 list median and maximum error of each method. We classify the sequences into static/dynamic (s/d) scenes, near/far (n/f) measurements, and continuous/discontinuous (c/g) recordings. Remarkably, our method achieves best performance in median as well as maximum error in most of the sequences. In the 24 static scenes with close-range measurements and continuous recordings, our approach is most accurate in 16 sequences (67%). GICP and 3D-NDT are outperformed by warp, fovis, and our approach which have been designed for RGB-D image registration.

From the tables and Fig. 3.2, we see that our method performs very accurately in the median and yields low maximum errors in static scenes with close-range measurements that have been recorded without frame gaps (s,n,c). Overall, our method performs clearly best in most of these sequences. In close-range, continuous, and dynamic scenes (Fig. 3.3), the accuracy of all approaches degrades with the degree of image coverage by dynamic objects. With only few portions of the image containing dynamic objects, our approach performs very accurately and is well competitive to the other approaches. In the freiburg3_walking sequences, larger parts of the image are dynamic which seems to affect our approach more strongly than 3D-NDT and fovis. In chapters 4 and 5 we will present non-rigid registration methods for MRSMaps that better cope with dynamics in the scene.

In sequences with mostly distant measurements such as freiburg2_large_no_loop, geometric features are barely measurable due to sensor noise and discretization of disparity. Fovis as a sparse image registration approach has an advantage on these sequences which indicates that point feature-based registration would complement our approach well. Further difficult scenes contain only little geometric structure but fine-grained texture such as the freiburg1_floor or the freiburg3_nostructure_texture sequences. In two out of three cases, warp and fovis yield higher median accuracy. Also this could be addressed in our registration method by including sparse interest points.

Table 3.1.: Comparison of median relative pose error (RPE) in mm for incremental registration on the sequences of the RGB-D benchmark dataset (Sturm et al., 2012).

sequence	prop.	MRSMap	warp (OpenCV)	GICP	3D-NDT	fovis
fr1 360	s,n,c	**5.0**	5.9	18.8	7.9	7.1
fr1 desk	s,n,c	**4.6**	5.8	10.2	7.9	6.3
fr1 desk2	s,n,c	**4.4**	6.2	10.4	8.2	6.6
fr1 floor	s,n,g	5.8	**2.1**	5.0	5.8	2.6
fr1 plant	s,n,c	**3.5**	4.2	16.1	7.3	4.6
fr1 room	s,n,c	**3.5**	4.6	10.2	6.1	5.4
fr1 rpy	s,n,c	**3.0**	5.1	10.4	6.8	5.4
fr1 teddy	s,n,c	**4.5**	6.1	21.3	8.9	7.1
fr1 xyz	s,n,c	**2.4**	4.1	3.9	5.2	4.6
fr2 360 hemisphere	s,f,c	27.3	40.6	18.2	53.8	**10.4**
fr2 360 kidnap	s,f,g	24.9	30.0	14.6	46.0	**11.8**
fr2 coke	d,n,c	**2.9**	3.3	5.9	12.1	3.6
fr2 desk	s,n,c	2.2	**2.1**	6.0	4.4	2.5
fr2 desk with person	d,n,g	2.3	2.1	5.1	4.1	**2.0**
fr2 dishes	s,n,c	2.6	2.1	5.7	7.7	**2.0**
fr2 flowerbouquet	d,n,c	2.8	2.3	4.8	13.8	**1.8**
fr2 flowerbouquet bg	d,n,c	4.8	3.5	5.8	5.0	**3.2**
fr2 large no loop	s,f,c	25.0	20.5	21.3	32.6	**11.0**
fr2 large with loop	s,f,c	25.8	94.7	22.5	48.5	**12.1**
fr2 metallic sphere	d,n,c	3.3	**3.0**	7.1	10.7	3.6
fr2 metallic sphere 2	d,n,c	5.5	**3.3**	9.4	9.6	6.2
fr2 pioneer 360	s,f,c	19.8	**12.7**	18.7	24.3	16.0
fr2 pioneer slam	s,f,g	14.5	10.0	16.6	17.3	**9.0**
fr2 pioneer slam 2	s,f,g	11.4	**6.4**	16.1	16.5	7.7
fr2 pioneer slam 3	g,f,c	6.9	5.7	10.3	10.1	**4.3**
fr2 rpy	s,n,c	1.7	1.7	**1.3**	4.1	1.7
fr2 xyz	s,n,c	**1.6**	2.0	1.7	4.0	1.9
fr3 cabinet	s,n,c	**4.7**	5.1	8.4	9.1	7.5
fr3 large cabinet	s,n,c	**8.3**	13.9	15.5	13.1	10.4
fr3 long office househ.	s,n,c	**2.7**	3.2	7.8	4.2	3.7
fr3 nostruct notext far	s,n,c	9.3	40.4	**8.6**	13.8	11.3
fr3 nostruct notext near	s,n,c	15.3	28.2	12.5	17.1	**11.2**
fr3 nostruct text far	s,n,c	18.1	19.2	**10.9**	18.6	20.8
fr3 nostruct text near	s,n,c	11.6	**7.0**	8.9	10.6	7.3
fr3 sitting halfsphere	d,n,c	**3.0**	4.6	11.4	5.5	5.3
fr3 sitting rpy	d,n,c	5.2	5.9	13.2	**4.0**	5.5
fr3 sitting static	d,n,c	4.2	2.7	2.8	**2.4**	3.0
fr3 sitting xyz	d,n,c	5.8	5.1	7.4	**3.6**	5.0
fr3 struct notxt far	s,n,c	**2.2**	8.6	4.5	2.9	9.1
fr3 struct notxt near	s,n,c	**2.1**	8.6	2.9	2.4	9.3
fr3 struct txt far	s,n,c	5.5	8.1	7.1	**5.4**	8.8
fr3 struct txt near	s,n,c	**3.2**	5.9	5.6	5.5	6.5
fr3 teddy	s,n,c	**2.7**	3.5	11.5	11.3	3.4
fr3 walking halfsphere	d,n,c	13.4	11.6	15.8	**7.9**	10.2
fr3 walking rpy	d,n,c	18.5	15.6	20.7	**10.8**	11.3
fr3 walking static	d,n,c	13.6	5.7	5.0	6.2	**4.7**
fr3 walking xyz	d,n,c	22.1	14.1	12.6	11.5	**10.5**
no. of best perf. (s,n,c)		**18 (16)**	7 (2)	3 (3)	6 (1)	13 (2)
total median		**4.1**	4.2	7.5	7.7	4.5

Table 3.2.: Comparison of maximum relative pose error (RPE) in mm for incremental registration on the sequences of the RGB-D benchmark dataset (Sturm et al., 2012).

sequence	prop.	MRSMap	warp (OpenCV)	GICP	3D-NDT	fovis
fr1 360	s,n,c	**41.8**	75.4	88.3	148.1	43.1
fr1 desk	s,n,c	**25.9**	131.8	54.9	26.6	34.2
fr1 desk2	s,n,c	**22.7**	147.4	261.2	46.3	49.9
fr1 floor	s,n,g	413.1	3167	193.6	**128.5**	412.4
fr1 plant	s,n,c	**28.1**	300.8	831.4	62.2	61.6
fr1 room	s,n,c	**45.0**	167.8	212.6	51.2	55.1
fr1 rpy	s,n,c	**28.9**	41.8	636.6	41.9	38.7
fr1 teddy	s,n,c	**79.4**	381.1	356.6	126.6	82.4
fr1 xyz	s,n,c	**9.6**	18.1	42.0	35.9	25.8
fr2 360 hemisphere	s,f,c	6296	5.6e5	1516	3505	**537.7**
fr2 360 kidnap	s,f,g	3124	1.0e5	**103.9**	1567	554.3
fr2 coke	d,n,c	**53.9**	78.2	2571	1792	236.2
fr2 desk	s,n,c	17.0	**14.1**	64.4	31.7	15.5
fr2 desk with person	d,n,g	70.2	307.6	179.4	77.0	**67.3**
fr2 dishes	s,n,c	**65.0**	2.8e4	2738	254.4	257.3
fr2 flowerbouquet	d,n,c	**66.1**	1.3e4	2411	254.0	79.2
fr2 flowerbouquet bg	d,n,c	26.9	2591	30.3	35.6	**20.0**
fr2 large no loop	s,f,c	485.7	7.6e5	1289	1995	**173.0**
fr2 large with loop	s,f,c	2177	5.1e5	1662	2176	**220.6**
fr2 metallic sphere	d,n,c	**50.0**	8.6e4	1891	146.1	166.7
fr2 metallic sphere 2	d,n,c	**69.4**	8.0e5	1062	285.9	407.6
fr2 pioneer 360	s,f,c	**173.1**	3.2e4	1369	628.0	235.4
fr2 pioneer slam	s,f,g	569.2	2.0e5	1785	1279	**566.2**
fr2 pioneer slam 2	s,f,g	908.6	1.6e5	1627	946.8	**902.7**
fr2 pioneer slam 3	s,f,g	**153.6**	1.8e5	1946	621.8	445.0
fr2 rpy	s,n,c	30.0	189.5	28.7	56.2	**11.0**
fr2 xyz	s,n,c	27.3	**8.8**	26.8	18.2	9.9
fr3 cabinet	s,n,c	**23.1**	291	58.0	111.8	77.7
fr3 large cabinet	s,n,c	120.4	4060	85.2	**70.7**	220.3
fr3 long office househ.	s,n,c	**16.1**	34.0	209.1	40.3	35.6
fr3 nostruct notext far	s,n,c	**49.4**	6.0e4	66.1	77.8	108.4
fr3 nostruct notext near	s,n,c	**57.8**	3.2e4	182.1	144.6	79.3
fr3 nostruct text far	s,n,c	**57.7**	1230	58.6	74.5	101.5
fr3 nostruct text near	s,n,c	60.8	100.5	67.1	90.9	**41.6**
fr3 sitting halfsphere	d,n,c	33.6	**32.3**	69.6	60.6	69.2
fr3 sitting rpy	d,n,c	105.3	1.7e4	413.2	165.7	**100.6**
fr3 sitting static	d,n,c	29.5	13.8	44.7	**9.1**	46.2
fr3 sitting xyz	d,n,c	48.4	26.8	51.2	**21.4**	34.8
fr3 struct notxt far	s,n,c	**17.1**	2579	23.2	19.2	62.4
fr3 struct notxt near	s,n,c	13.2	1108	**12.2**	44.5	86.9
fr3 struct txt far	s,n,c	23.0	39.0	23.3	**21.4**	45.2
fr3 struct txt near	s,n,c	**14.5**	34.8	34.2	82.8	38.2
fr3 teddy	s,n,c	323.0	5.2e4	310.6	714.5	**73.5**
fr3 walking halfsphere	d,n,c	203.6	109.0	84.7	84.8	**74.6**
fr3 walking rpy	d,n,c	1443	2.9e4	334.5	251.0	**172.3**
fr3 walking static	d,n,c	128.8	97.3	39.7	**32.2**	61.3
fr3 walking xyz	d,n,c	150.3	114.4	102.0	102.3	**64.3**
no. of best perf. (s,n,c)		**22 (16)**	3 (2)	2 (1)	6 (2)	14 (3)

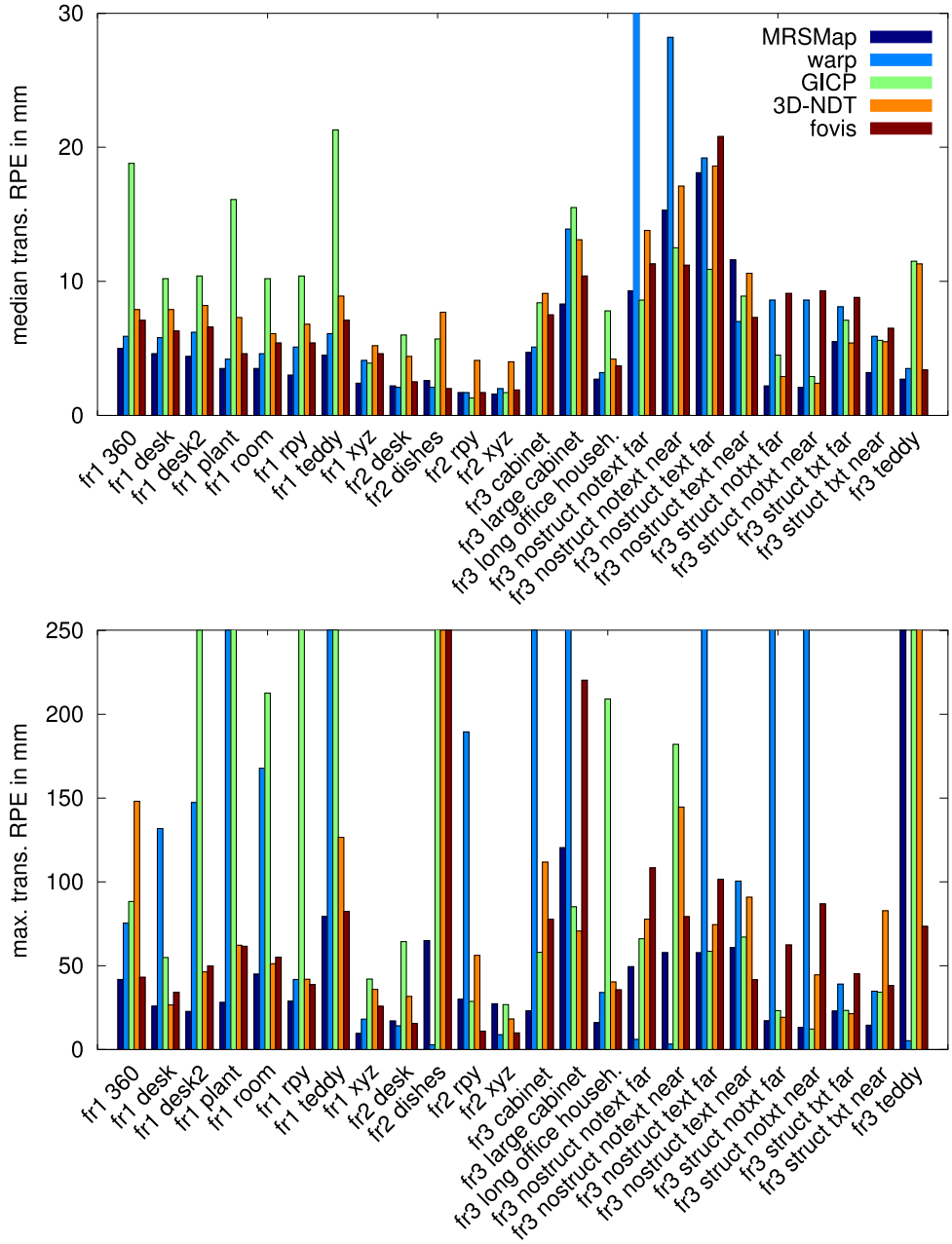

Figure 3.2.: Median (top) and maximum (bottom) translational RPE of the registration estimate on static sequences of the RGB-D bechmark dataset (close-range measurements, no frame gaps).

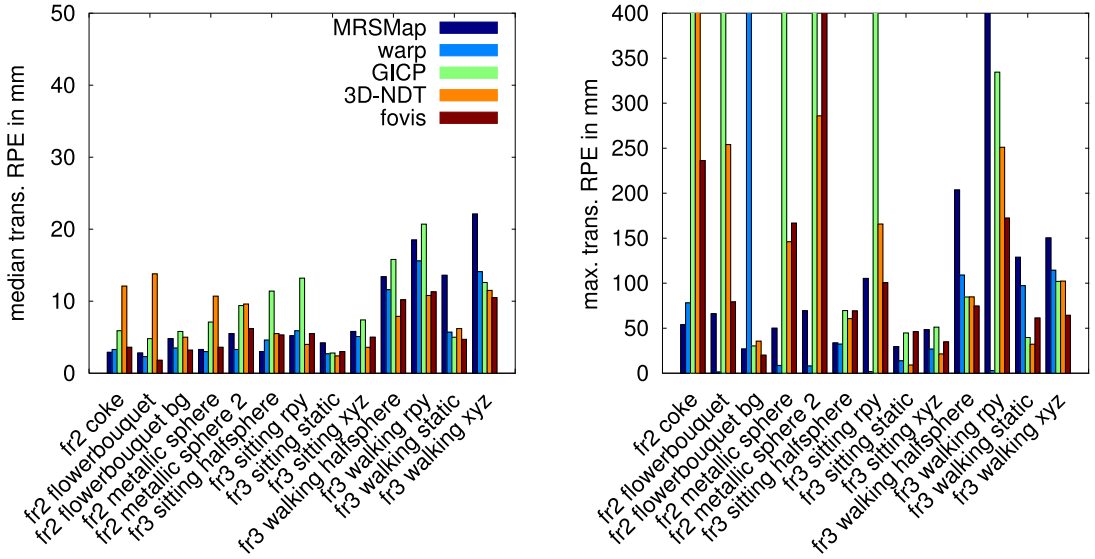

Figure 3.3.: Median (left) and maximum (right) translational RPE of the registration estimate on sequences of the RGB-D bechmark dataset with dynamic objects (close-range measurements, no frame gaps).

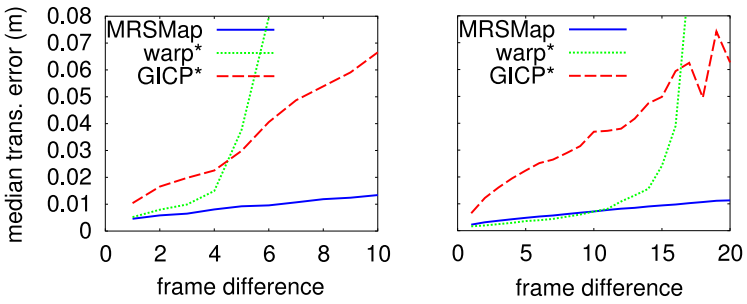

Figure 3.4.: Median translational error of the registration estimate for different frame skips on the freiburg1_desk (left) and freiburg2_desk (right) sequences (*results from (Steinbruecker et al., 2011)).

3.3.3. Robustness

In Fig. 3.4, we evaluate the robustness of our approach for skipping frames on the freiburg1_desk and freiburg2_desk sequences[1]. Our approach achieves similar accuracy than warp for small displacements, but retains the robustness of ICP methods for larger displacements when warp fails. This property is important for real-time operation, if frames need to be dropped eventually. On both sequences, the distribution of translational errors made by our method is narrow at small errors for small frame-gaps (see Fig. 3.5). With increasing

[1]Results for warp and GICP taken from (Steinbruecker et al., 2011).

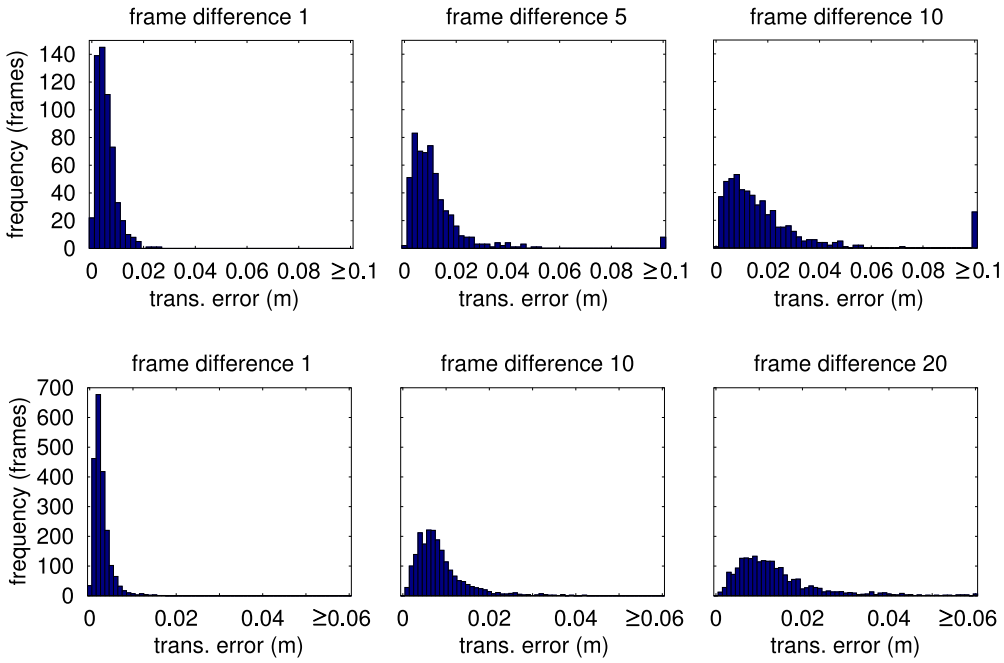

Figure 3.5.: Histograms of translational errors of the registration estimate for different frame skips on the freiburg1_desk (top) and freiburg2_desk (bottom) sequences.

frame gap, the histograms flatten towards larger translational errors. It can be seen that outliers beyond 0.1 m and 0.06 m, respectively, are rare. Note that freiburg1_desk sequence contains fast camera motion with strong motion blur, while the camera moves slowly in freiburg2_desk.

Figs. 3.6 and 3.7 give further insights into the robustness of our method with respect to translational and rotational camera motion between frames. We chose both sequences to contain cluttered close-range scenery in the majority of frames, such that the effect of varying distances on the results is small. Our approach can handle translational and rotational motion of 0.2 m and 0.2 rad well in most cases, even on the freiburg1_desk sequence. The use of both shape-texture descriptors as well as color increases the basin of convergence of our registration method.

3.3.4. Run-Time

Our approach achieves ca. 23 Hz in average on the sequences (see Table 3.3). It is much more efficient than GICP or 3D-NDT, and demonstrates slightly faster run-time than warp. Fovis requires lower run-time on the sequences. This is a natural result, since it reduces the image to a set of interest points and registers those instead of using all available measurements in the image.

Table 3.3.: Comparison of average (std. dev.) runtime in milliseconds for incremental registration on the RGB-D benchmark sequences.

sequence	MRSMap	warp (OpenCV)	GICP	3D-NDT	fovis
fr1 360	47.1 (15.9)	50.0 (10.7)	4601 (1919)	532.2 (306.0)	**9.5 (2.4)**
fr1 desk	66.1 (9.9)	69.9 (12.7)	2809 (1338)	315.2 (169.2)	**8.5 (1.9)**
fr1 desk2	65.8 (13.5)	63.1 (11.4)	3677 (2146)	341.6 (143.7)	**8.4 (1.8)**
fr1 floor	77.7 (10.3)	57.7 (13.0)	2197 (1020)	231.8 (79.6)	**11.6 (2.5)**
fr1 plant	49.9 (8.6)	60.4 (8.8)	3645 (2947)	626.9 (247.0)	**8.5 (1.8)**
fr1 room	58.1 (13.7)	63.9 (11.2)	3667 (2136)	403.0 (196.9)	**8.9 (1.7)**
fr1 rpy	63.0 (12.9)	63.6 (8.2)	4066 (4484)	455.2 (333.3)	**8.5 (1.7)**
fr1 teddy	44.8 (11.2)	63.7 (10.2)	4359 (3168)	665.5 (209.7)	**8.9 (2.3)**
fr1 xyz	68.2 (6.8)	71.3 (7.8)	2425 (766.4)	351.0 (99.6)	**8.8 (1.7)**
fr2 360 hemisphere	23.8 (5.9)	15.9 (8.9)	2858 (4602)	1223 (1062)	**6.9 (3.2)**
fr2 360 kidnap	21.9 (5.8)	20.0 (7.5)	2292 (1428)	1065 (898.9)	**6.5 (2.5)**
fr2 coke	47.8 (11.4)	31.6 (10.0)	2211 (4650)	653.8 (386.7)	**6.2 (1.5)**
fr2 desk	49.6 (6.9)	63.2 (7.2)	2266 (623.6)	500.2 (135.0)	**8.3 (1.1)**
fr2 desk with person	42.8 (7.5)	57.9 (5.2)	1988 (734)	528.5 (160.8)	**7.7 (0.9)**
fr2 dishes	52.7 (11.0)	34.4 (7.4)	2003 (4629)	519.6 (366.5)	**8.2 (2.5)**
fr2 flowerbouquet	50.2 (6.2)	36.0 (5.6)	2107 (3676)	706.8 (350.3)	**6.0 (0.9)**
fr2 flowerbouquet bg	48.5 (6.0)	37.9 (2.0)	1817 (655.5)	431.1 (107.9)	**6.4 (1.0)**
fr2 large no loop	26.7 (7.6)	36.8 (12.3)	3644 (5955)	942.2 (583.5)	**7.1 (2.2)**
fr2 large with loop	25.5 (7.7)	31.6 (17.2)	3559 (6745)	963.4 (628.3)	**6.5 (2.2)**
fr2 metallic sphere	49.4 (14.4)	31.3 (3.3)	2770 (5857)	699.7 (462.8)	**6.2 (1.7)**
fr2 metallic sphere 2	42.3 (12.1)	29.0 (6.7)	2129 (2817)	572.7 (308.7)	**5.9 (1.6)**
fr2 pioneer 360	36.2 (7.5)	25.4 (3.6)	4240 (7103)	834.6 (744.4)	**5.5 (2.1)**
fr2 pioneer slam	40.2 (8.6)	31.0 (8.0)	4598 (7189)	718.4 (563.8)	**6.5 (2.4)**
fr2 pioneer slam 2	40.0 (8.3)	31.8 (6.1)	4122 (5956)	843.3 (972.6)	**7.9 (2.6)**
fr2 pioneer slam 3	38.7 (7.4)	31.6 (7.8)	3008 (5252)	649.4 (400.1)	**5.8 (2.3)**
fr2 rpy	44.7 (9.5)	57.6 (9.3)	1424 (464.7)	489.5 (275.7)	**7.4 (1.3)**
fr2 xyz	52.1 (14.3)	63.8 (5.4)	1365 (345.0)	451.9 (127.1)	**8.2 (1.2)**
fr3 cabinet	54.0 (7.7)	32.5 (1.8)	2667 (1183)	511.7 (289.6)	**6.7 (2.0)**
fr3 large cabinet	29.6 (7.0)	31.5 (2.7)	2402 (733.5)	705.8 (324.9)	**5.6 (1.2)**
fr3 long office househ.	54.0 (13.8)	66.5 (8.7)	2620 (1502)	472.7 (199.1)	**9.7 (1.3)**
fr3 nostruct notxt far	32.2 (2.6)	26.3 (4.3)	2269 (843.6)	383.4 (133.9)	**8.3 (2.7)**
fr3 nostruct notxt near	55.7 (5.2)	24.6 (6.0)	2346 (1355)	287.3 (135.7)	**9.2 (3.9)**
fr3 nostruct txt far	29.7 (3.6)	60.2 (18.5)	2100 (943.7)	638.2 (228.8)	**10.0 (3.8)**
fr3 nostruct txt near	56.0 (6.3)	67.3 (11.6)	1870 (699.7)	218.8 (78.2)	**10.6 (2.3)**
fr3 sitting halfsphere	37.2 (7.4)	61.5 (15.6)	2290 (805.5)	476.6 (159.1)	**6.2 (1.8)**
fr3 sitting rpy	31.8 (8.8)	57.5 (17.7)	2633 (1939)	493.3 (215.3)	**6.4 (2.0)**
fr3 sitting static	31.2 (4.9)	74.5 (4.2)	1652 (424.6)	479.4 (141.5)	**6.9 (0.9)**
fr3 sitting xyz	29.4 (7.2)	63.4 (8.0)	1885 (424.6)	489.3 (152.6)	**5.8 (1.2)**
fr3 struct notxt far	38.9 (3.5)	30.1 (1.6)	2112 (502.5)	433.7 (129.3)	**4.2 (0.5)**
fr3 struct notxt near	73.1 (7.0)	29.8 (1.2)	1923 (319.4)	216.7 (82.6)	**4.2 (0.5)**
fr3 struct txt far	36.9 (4.0)	85.2 (8.4)	2390 (608.6)	472.6 (153.9)	**12.2 (1.8)**
fr3 struct txt near	59.6 (6.4)	62.1 (10.4)	2189 (558.2)	358.6 (164.7)	**10.7 (1.8)**
fr3 teddy	44.9 (11.1)	39.0 (7.8)	3172 (5292)	614.7 (244.3)	**6.9 (1.7)**
fr3 walking halfsphere	34.7 (10.2)	59.6 (9.6)	2538 (821.5)	485.1 (153.2)	**6.8 (2.2)**
fr3 walking rpy	33.4 (10.3)	55.2 (14.7)	3037 (3941)	542.6 (224.3)	**6.6 (2.3)**
fr3 walking static	28.5 (7.2)	62.9 (7.7)	1623 (404.8)	557.1 (171.7)	**5.6 (1.3)**
fr3 walking xyz	29.8 (9.5)	56.8 (12.5)	2075 (570.6)	509.3 (201.4)	**5.3 (1.1)**
average (std.dev.)	43.6 (15.4)	45.7 (19.0)	2632 (3829)	607.5 (478)	**7.4 (2.5)**

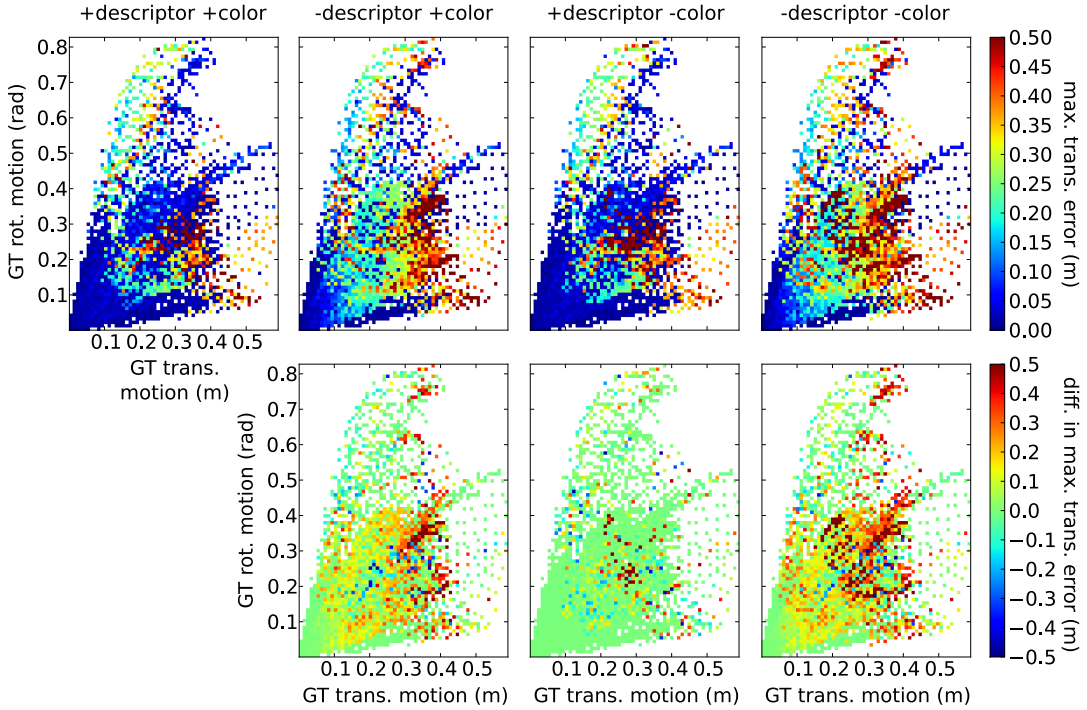

Figure 3.6.: Maximum translational error of the registration estimate in relation to ground truth translation and rotation on the freiburg1_desk sequence.

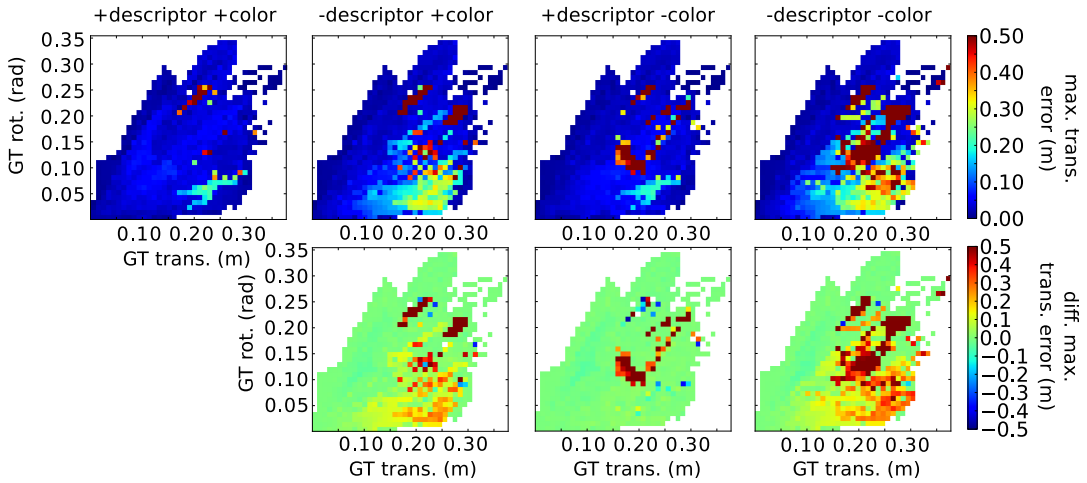

Figure 3.7.: Maximum translational error of the registration estimate in relation to ground truth translation and rotation on the freiburg2_desk sequence.

3.4. Related Work

Estimating camera pose between two images from a monocular color or grayscale camera is in general an ill-posed problem without knowledge about the 3D location of observed features. The seminal works by Nister et al. (2004) and Davison et al. (2007) proposed two competing approaches for estimating the motion of a monocular camera from a sequence of images. Davison et al. (2007) filter camera pose and 3D position of interest points in an extended Kalman filter (EKF) framework. Nister et al. (2004) determine the relative poses of the camera in three frames from interest point correspondences using the 5-point-algorithm (Nister, 2004), random sample consensus (RANSAC), and refinement through bundle adjustment. The idea of local bundle adjustment has also been used in methods based on sliding windows (Mouragnon et al., 2006) and approaches that sparsify the camera trajectory in key frames (Klein and Murray, 2007).

The 3D geometry of interest points can be directly estimated from stereo camera images to find relative camera poses in real-time. Nister et al. (2004) proposed to estimate stereo camera motion using a 3-point algorithm and bundle adjustment refinement. Howard (2008) enforce consistent rigid arrangement of the interest point matches to further improve robustness of sliding window bundle adjustment. Textured-light projecting RGB-D cameras share common principles with stereo camera. Fovis (Huang et al., 2011) applies concepts from stereo image processing to RGB-D cameras. The approach initializes interest point matching and bundle adjustment with a coarse rotation estimate that is obtained through image correlation.

In robotics and computer graphics, depth images are frequently registered by variants of the ICP (Besl and McKay, 1992) algorithm. For instance, May et al. (2009) match time-of-flight depth images using ICP. Such methods operate on raw points directly and typically require subsampling to a managable image size to achieve high frame rate. GICP (Segal et al., 2009) unifies the ICP formulation for various error metrics such as point-to-point, point-to-plane, and plane-to-plane. Magnusson et al. (2007) propose a registration method for the 3D-NDT. The 3D grid discretization allows for efficient nearest neighbor look-ups. Scans are registered by minimizing the matching likelihood of scene points to the 3D-NDT model. In Color-NDT (Huhle et al., 2008), they propose to enrich 3D-NDT with Gaussian mixture distributions of color in each cell, and propose a registration method for this representation. More recently, Stoyanov et al. (2012) extended the 3D-NDT to register a 3D-NDT of a scene point cloud to a model 3D-NDT. To the best of our knowledge, none of the above ICP methods is reported to support real-time capable scan-matching of RGB-D images.

Our approach bears similarities to 3D-NDT matching. However, we propose novel methods to increase robustness and to enable high frame-rate operation on RGB-D images: Our approach exploits measurement principles of RGB-D

sensors to aggregate maps at high frame-rate. To register such views efficiently, we propose a multi-resolution strategy to data association. This strategy is supported by the use of color and shape-texture descriptors to judge the compatibility between surfels. While 3D-NDT also supports a multi-resolution representation of 3D scans, their registration optimizes from coarse to fine resolutions, only considering a single resolution at a time. Our highly efficient implementation registers 640×480 RGB-D images at a frame rate of about 23 Hz on a CPU.

In recent years, affordable depth cameras such as time-of-flight or structured-light cameras (e.g. Microsoft Kinect, Asus Xtion) have become available. Paired with the developments in computer vision on real-time dense depth estimation from monocular image sequences, exploiting dense depth for robotic perception is now a viable option. The premise is to increase the robustness of image registration over the use of sparse interest points detected in textured image regions. It should be most beneficial in textureless environments that have geometric structure. Efficient means have to be developed, however, to take full advantage of the high frame rates and high-resolution images provided by such sensors. Steinbruecker et al. (2011) proposed a method for dense real-time registration of RGB-D images. They model the perspective warp between images through view-pose changes and optimize for the best pose that explains the difference in intensity. In our approach, we construct 3D representations of the images and optimize for the relative pose between them. Note that our registration method is more general, since our representation supports data fusion from multiple view points. Hence, we also employ it for the registration of images to maps that aggregate multiple views, e.g., for tracking multi-view object models.

Endres et al. (2012) match RGB interest points between frames, align them using the depth measured at the interest points, and refine the registration estimate with ICP. Our registration method incorporates shape and texture seamlessly and is also applicable to textureless shapes. In KinectFusion (Newcombe et al., 2011a), depth images are aligned with a map that represents surface by a signed distance function in a 3D voxel grid. The map is updated in each frame with the aligned depth image. The registration method is based on ICP and projects the current map into the depth image for data association. To achieve real-time performance, the approach is implemented on GPU.

3.5. Summary

In this chapter, we introduced an efficient registration method for MRSMaps in which we assume rigidness of the observed scene. Our method gains efficiency from the concise representation of MRSMaps. The precalculation of surfel means, covariance, normals, shape-texture descriptors, and voxel neighbors in the map acquisition stage supports efficient registration.

Registration is performed in a dual iterative refinement procedure. Given

the latest pose estimate, we associate surfels between the maps on the finest common resolution by efficient volume queries in the octree representation. If an association for a surfel is available from previous iterations, we bootstrap the association by only searching among the direct neighbors in the voxel grid. Surfel associations are spared, if representative surfels on finer resolutions already have a matching.

For aligning the maps from surfel associations, we optimize the observation likelihood of one map in the other. Each surfel association contributes a normal-distributed factor to the optimization objective. The logarithm of this objective is optimized using a combination of the LM method for coarse registration and Newton's method for fine alignment.

In experiments, we demonstrate superior performance of our approach to state-of-the-art methods for visual odometry in terms of accuracy and robustness on an RGB-D benchmark dataset. Especially on static scenes with close-range measurements and continuous recordings, our method outperforms the other approaches in accuracy on most sequences. The run-time of our algorithm is in average ca. 23 Hz and is competitive to the other dense registration methods.

In future work we will consider the implementation of our registration method on GPU. While we designed our method to register maps that include images from multiple view-points, our registration approach could further be tailored for visual odometry purposes by associating surfels through back-projection into the images. MRSMaps primarily model the distribution of depth measurements. To also incorporate fine-grained texture information into the registration, sparse interest points could be included into our map and registration algorithms.

4. Rigid Multi-Body Registration

The rigid registration approach presented in the previous chapter assumes that the complete scene moves with a single rigid-body motion. In this section, we propose an image registration approach that releases this assumption: We register RGB-D measurements of rigid image parts that move differently between two images. We do not require the correct segmentation of the image to be known a-priori, but estimate the segmentation into rigid parts concurrently with their motion (see Fig. 4.1).

Several approaches to motion segmentation in monocular or stereo video have been investigated. Sparse interest points have frequently been used to segment the image into sets of interest points with common 3D rigid body motion (Gruber and Weiss, 2004; Schindler and Suter, 2006; Rothganger et al., 2007; Ross et al., 2010; Agrawal et al., 2005). Most recent methods for dense 3D motion segmentation are still far from real-time performance (Sekkati and Mitiche, 2006; Zhang et al., 2011; Wang et al., 2012; Roussos et al., 2012).

We develop an expectation-maximization framework that recovers motion segments, estimates their 3D rigid-body motion, and also finds the number of segments in the scene efficiently. Our formulation makes no difference between background and foreground objects and, hence, copes with camera motion and multiple moving objects in the scene. We exploit dense depth information from

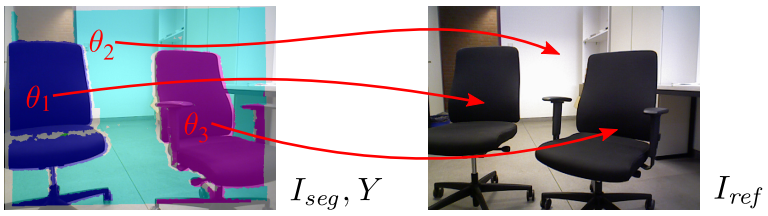

Figure 4.1.: The objective of our rigid multi-body registration algorithm is to estimate a segmentation Y of an image I_{seg} into segments that undergo rigid body motions θ_k towards a reference image I_{ref}.

RGB-D cameras and utilize our highly efficient image representation and rigid registration techniques within a rapid segmentation method. By representing RGB-D images in MRSMaps instead of using the raw images, our algorithm operates on significantly less image sites which also facilitates efficient dense inference of the segmentation.

4.1. Background

We formulate rigid multi-body registration within the expectation-maximization (EM) framework. EM concurrently optimizes for parameters as well as recovers latent, i.e., not directly observed variables, in a probabilistic model. In our multi-body registration approach, we assign labels to image pixels for different moving objects. This labeling problem is represented as a probabilistic graphical model, for which we consider variational and graph-based approximations for efficient inference. In the following, we introduce basic concepts and notation of EM and probabilistic graphical models.

4.1.1. Expectation-Maximization

Let $p(x \mid \theta)$ be a probability distribution over the random variables $x \in X$, parametrized by θ. If this distribution can be directly evaluated for observed $x \in X$, we may use an optimization method from Sec. 3.1.1 to determine a maximum likelihood (ML) solution for the parameters θ,

$$\theta_{ML} := \arg\max_{\theta} p(x \mid \theta). \tag{4.1}$$

EM comes into play, if the distribution $p(x \mid \theta)$ misses unobserved, latent variables Y, and only a simple closed form for $p(x, y \mid \theta)$ exists. If it is difficult to optimize the marginal

$$p(x \mid \theta) = \sum_{y \in Y} p(x, y \mid \theta) \tag{4.2}$$

for the parameters θ directly, EM provides an iterative approach that splits the optimization into two simpler steps.

We begin the derivation of EM by applying the logarithm to the probability distribution in Eq. (4.2),

$$\ln p(x \mid \theta) = \ln \sum_{y \in Y} p(x, y \mid \theta). \tag{4.3}$$

EM now constructs a tractable lower bound to Eq. (4.3), which is maximized instead. If this bound gets closer to the actual objective in each iteration, EM converges to a local maximum of Eq. (4.3).

Since the logarithm is a concave function, we can apply Jensen's inequality $f\left(\sum_i a_i x_i\right) \geq \sum_i f\left(a_i x_i\right)$ with $a_i \geq 0$ and $\sum_i a_i = 1$ to obtain

$$L(q,\theta) := \sum_{y \in Y} q(y) \ln \frac{p(x,y \mid \theta)}{q(y)} \leq \ln \sum_{y \in Y} q(y) \frac{p(x,y \mid \theta)}{q(y)}, \tag{4.4}$$

with a function $q(y)$ that satisfies $q(y) \geq 0$ and $\sum_{y \in Y} q(y) = 1$.

We closely follow Bishop (2006) to find the optimal choice of $q(y)$ for the lower bound. We show that

$$\ln p(x \mid \theta) = L(q,\theta) + \mathrm{KL}(q(y) \parallel p(y \mid x,\theta)), \tag{4.5}$$

in which

$$\mathrm{KL}(q(y) \parallel p(y \mid x,\theta)) := -\sum_{y \in Y} q(y) \ln \frac{p(y \mid x,\theta)}{q(y)} \tag{4.6}$$

is the Kullback Leibler divergence (KL-divergence) between $q(y)$ and $p(y \mid x,\theta)$. Using Bayes' rule we decompose

$$\ln p(x,y \mid \theta) = \ln p(y \mid x,\theta) + \ln p(x \mid \theta), \tag{4.7}$$

and substitute into $L(q,\theta)$ to obtain

$$L(q,\theta) = \sum_{y \in Y} q(y) \left(\ln \frac{p(y \mid x,\theta)}{q(y)} + \ln p(x \mid \theta) \right). \tag{4.8}$$

Since $\sum_{y \in Y} q(y) = 1$, we have

$$L(q,\theta) = \ln p(x \mid \theta) - \mathrm{KL}(q(y) \parallel p(y \mid x,\theta)), \tag{4.9}$$

so $L(q,\theta)$ is maximized, if the KL-divergence term vanishes, which happens when $q(y) = p(y \mid x,\theta)$.

This result is utilized in a dual iterative algorithm (Bishop, 2006). Assuming that we have a current estimate of the paramters $\overline{\theta}$ from a previous iteration, we maximize the lower bound $L(q,\overline{\theta})$ for the functional $q(y)$. As we have seen, this amounts to the minimization of the KL-divergence, which vanishes for $q(y) = p(y \mid x,\overline{\theta})$. Hence, we need to determine the distribution $p(y \mid x,\overline{\theta})$ given the current parameter estimate, which we call the expectation step (E-step).

In the maximization step (M-step), we hold the distribution $\overline{q}(y)$ from the previous E-step fixed, and maximize $L(\overline{q},\theta)$ for a new parameter estimate $\widehat{\theta}$,

$$\widehat{\theta} = \arg\max_{\theta} \sum_{y \in Y} \overline{q}(y) \ln p(x,y \mid \theta) - \overline{q}(y) \ln \overline{q}(y). \tag{4.10}$$

It is apparent that only the first term depends on θ. This term can be interpreted as the expectation of $\ln p(x,y \mid \theta)$ conditioned on $p(y \mid x,\overline{\theta})$, hence the name expectation-maximization.

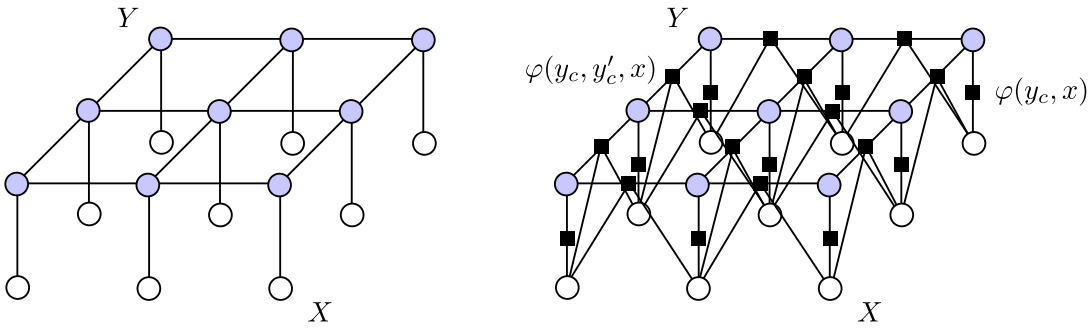

Figure 4.2.: Undirected graphical model of Markov random fields (MRFs) and factor graph of conditional random fields (CRFs) with unary and pairwise potentials.

The EM procedure can be proven to converge to a local maximum of the objective in Eq. (4.2) (Dempster et al., 1977). The M-step increases the lower bound and, hence, the log-likelihood in Eq. (4.2). Since we keep $\overline{q}(y)$ fixed from the previous E-step, $q(y)$ does not equal $p(y \mid x, \widehat{\theta})$ such that the KL-divergence term increases (but adding to the objective). The subsequent E-step will then improve the lower bound by cancelling the KL-divergence.

4.1.2. Probabilistic Graphical Models for Image Labeling Tasks

Random fields represent observations and spatial layout in images in a probabilistic graphical model. Markov random fields(MRFs) (Geman and Geman, 1984) and conditional random fields(CRFs) (Lafferty et al., 2001) are variants of undirected graphical models that are frequently used in the formulation of image labeling problems.

4.1.2.1. Undirected Graphical Models

Undirected graphical models represent Markov properties of a probability distribution $p(X)$ over a set of random variables $X = \{X_1, \ldots, X_N\}$ in an undirected graph $G = (\mathcal{V}, \mathcal{E})$. The nodes $\mathcal{V}$ of the graph correspond to the random variables in X. Edge connectivity models conditional independency relations: Two sets X_A and X_B of random variables are conditionally independent given a set X_C, if there is no path between the nodes for the variables in X_A to nodes for X_B other than through nodes for X_C. This leads to the factorization of $p(x)$ into local potentials $\varphi(x_c) \geq 0$ over maximal cliques $c \in \mathcal{C}$ of random variables X_c in the graph,

$$p(x) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \varphi(x_c) \tag{4.11}$$

where x denotes an instantiation of the random variable X, X_c is the set of random variables contained in clique c, and $Z := \sum_{x \in X} p(x)$ is the partition function. This general kind of graphical model is also referred to as MRF or Markov network (MN).

Undirected graphical models can alternatively be viewed in terms of energy functions $E(x)$ where

$$E(x) = \sum_{c \in \mathcal{C}} -\ln \varphi(x_c), \tag{4.12}$$

such that

$$p(x) = \frac{1}{Z} \exp(-E(x)). \tag{4.13}$$

Factor graphs (Kschischang et al., 2001) are intermediate representations of graphical models that allow inference algorithms such as belief propagation (BP) to be formulated in a concise way. The probability distribution of the graphical model is written as a product of factors $f_c(x_c)$

$$p(x) = \prod_c f_c(x_c) \tag{4.14}$$

over sets of variables $X_c := X(f_c)$. For undirected graphical models, the factors correspond to the clique potentials. The partition function is subsumed in a factor over the empty set of random variables. Factor graphs $G_F = (\mathcal{V}_F, \mathcal{E}_F)$ then represent the graphical model $G = (\mathcal{V}, \mathcal{E})$ by nodes $\mathcal{V}_F := \mathcal{V} \cup \mathcal{V}_f$ for the random variables ($\mathcal{V}$) and the factors ($\mathcal{V}_f$). Edges $\mathcal{E}_F$ connect random variables with the factor nodes they are involved in. We denote the neighbors of random variables x_c and factors f_c by $\mathcal{N}_F(x_c)$ and $\mathcal{N}_F(f_c)$, respectively.

4.1.2.2. Image Modeling in Markov Random Fields

MRF models of images define the conditional probability distribution of the latent variables Y given the observations X in a generative way (Geman and Geman, 1984)

$$p(y \mid x) = \frac{1}{Z(x)} p(x, y) = \frac{1}{Z(x)} p(x \mid y) p(y), \tag{4.15}$$

for which the partition function is $Z(x) := \sum_{y \in Y} p(x, y)$ (see Fig. 4.2).

For typical MRFs, we assume stochastic independence between the observations x_i at the image sites such that we specify the observation likelihood $p(x_i \mid y_i)$ at the individual site. The prior $p(y)$ models the stochastic relationships between spatial neighbors. Frequently, the latent variable y_i depends on its four direct neighbors $\mathcal{N}(y_i)$ in the image grid. In the terminology of undirected graphical models, the observation likelihood generates cliques $(x_i, y_i) \in \mathcal{C}_U$ of size 2 between the observations x_i and latent variables y_i. Each pair y_c, y'_c of neighboring latent variables in the grid defines a clique $(y_c, y'_c) \in \mathcal{C}_P$. The resulting

undirected graphical model has a maximal clique size of 2, and its distribution is

$$p(y \mid x) = \frac{1}{Z(x)} \prod_{(x_c, y_c) \in \mathcal{C}_U} p(x_c \mid y_c) \prod_{(y_c, y_c') \in \mathcal{C}_P} \varphi(y_c, y_c'). \tag{4.16}$$

We denote $p(x_c \mid y_c)$ as a unary potential, since it only depends on one latent variable, and $\varphi(y_c, y_c')$ are pairwise potentials. For the sake of notation simplicity, we refer by x_c, y_c, and y_c' to instantiations of random variables as well as the random variables' nodes in the graph.

4.1.2.3. Conditional Random Fields

CRFs do not explain the image in a generative model for $p(y \mid x)$. Instead this distribution is directly modeled as an undirected graphical model on y conditioned on the observations x (Lafferty et al., 2001):

$$p(y \mid x) = \frac{1}{Z(x)} \prod_{y_c \in \mathcal{C}} \varphi(y_c, x), \tag{4.17}$$

where $\mathcal{C}$ is the set of cliques in the graphical model and $\varphi(y_c, x)$ are local potentials on the variables y_c within a clique conditioned on the observations x (see Fig. 4.2). If we restrict the model to direct pairwise dependencies between neighboring image sites, we have

$$p(y \mid x) = \frac{1}{Z(x)} \prod_{y_c \in \mathcal{C}_U} \varphi(y_c, x) \prod_{(y_c, y_c') \in \mathcal{C}_P} \varphi(y_c, y_c', x). \tag{4.18}$$

The important difference to MRFs is that every potential may depend on all the observations.

4.1.2.4. Inference using Loopy Belief Propagation

4.1.2.5. Undirected Graphical Models

Exact inference is in general computationally intractable in graphs with loops such as the MRF and CRF image models. Various approximate inference methods exist like simulated annealing, loopy belief propagation (LBP), Monte Carlo Markov chain, or variational methods (Bishop, 2006). BP passes local messages in factor graphs to either find the marginals $p(x_i)$ of random variables $X_i \in X$ or the maximum-likelihood assignment $x_{ML} = \arg\max_{x \in X} p(x)$.

Sum-Product Algorithm: The sum-product algorithm determines the marginals of random variables (Bishop, 2006)

$$p(x_i) = \prod_{f \in \mathcal{N}_F(x_i)} \mu_{f \to x_i}(x_i) \tag{4.19}$$

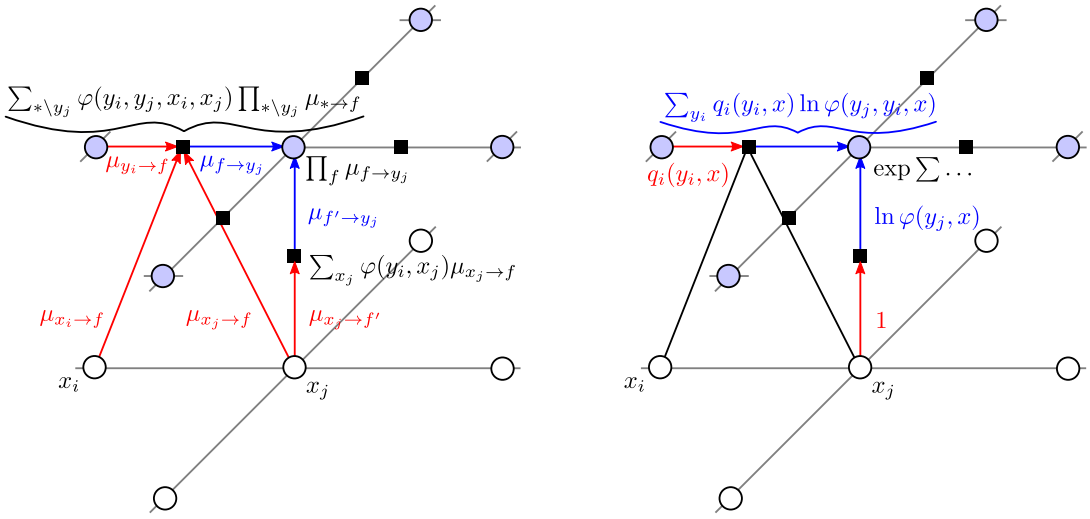

Figure 4.3.: Left: Local update scheme in a CRF for sum-product loopy belief propagation. Right: Local update scheme in a CRF for its variational mean-field approximation.

from local messages $\mu_{f \to x_i}(x_i)$ which are passed from neighboring factor nodes. The message from a factor f to a variable node x_i

$$\mu_{f \to x_i}(x_i) = \sum_{x' = (x'_1, \ldots, x'_M) \in X(f) \setminus x_i} f(x_i, x') \prod_{m \in \{1, \ldots, M\}} \mu_{x'_m \to f}(x'_m) \tag{4.20}$$

is obtained from messages to f from neighboring variable nodes except x_i. Messages from random variables to factor nodes

$$\mu_{x_i \to f}(x_i) = \prod_{f' \in \mathcal{N}_F(x_i) \setminus f} \mu_{f' \to x_i}(x_i) \tag{4.21}$$

conversely involve messages passed from neighboring factor nodes except the target factor node itself. Initially, we set $\mu_{x_i \to f}(x_i) = 1$ and $\mu_{f \to x_i}(x_i) = f(x_i)$. Evidence at a subset of random variables can be incorporated by clamping the messages from the observed random variables to the observed distribution. Fig. 4.3 illustrates the local update scheme of LBP.

Max-Sum Algorithm: The max-sum algorithm determines the ML assignment of the random variables (Bishop, 2006)

$$x_{ML} = \arg\max_x p(x). \tag{4.22}$$

By transforming the probability distribution $p(x)$ into the log-domain, we obtain an efficient algorithm that replaces the sums in Eqs. (4.19), (4.20), and (4.21)

by the max-operator and the products by sums:

$$x_{ML,i} = \arg\max_{x_i} \sum_{f \in \mathcal{N}_F(x_i)} \mu_{f \to x_i}(x_i), \tag{4.23}$$

$$\mu_{f \to x_i}(x_i) = \max_{x'_i = (x'_1, \ldots, x'_M) \in X(f) \setminus x_i} \ln f(x_i, x') + \sum_{m \in \{1,\ldots,M\}} \mu_{x'_m \to f}(x'_m), \tag{4.24}$$

$$\mu_{x_i \to f}(x_i) = \sum_{f' \in \mathcal{N}_F(x_i) \setminus f} \mu_{f' \to x_i}(f'). \tag{4.25}$$

Care needs to be taken, if multiple assignments to local x_i would maximize Eq. (4.23). In this case, back-tracking needs to be performed to recover a correct maximum-likelihood assignment (Bishop, 2006).

Sum-product and max-sum BP are exact inference algorithm for tree-structured graphs. A single sweep through the tree suffices to compute the marginals. In graphs with loops, BP only yields approximate algorithms and needs to be iterated. Initialization and scheduling of the message passing have a strong influence on the convergence and the quality of the local optimum that is found by the algorithms. Since messages are exchanged only between direct neighbors in the graph, it may take many iterations until contextual information is distributed to distant nodes. This can be especially problematic in graphical models of images with only pairwise spatial neighborhoods, leading to slow convergence or poor local optima.

4.1.2.6. Inference using Variational Mean-Field Approximations

In variational approximate inference, we seek to replace a probability distribution $p(x)$ with a functional $q(x)$ with a specific form such that a good fit to the original distribution can be obtained but inference is much simpler.

Variational mean-field approximations which are also referred to as factorized approximations represent the distribution

$$p(x) \approx q(x) = \prod_i q_i(x_i) \tag{4.26}$$

by factors over subsets of the random variables X. Our goal is then to minimize the KL-divergence between the distributions, i.e.,

$$\widehat{q}(x) = \arg\min_{q(x)} \ \mathrm{KL}(q(x) \parallel p(x)) \tag{4.27}$$

with

$$\mathrm{KL}(q(x) \parallel p(x)) := \sum_{x \in X} q(x) \ln \frac{q(x)}{p(x)}. \tag{4.28}$$

For inference using the mean-field approximation, we derive a rule to update the individual factors $q_j(x_j)$ such that the KL-divergence in Eq. (4.28) is minimized with respect to this factor. We first plug the factorized distribution into the KL-divergence to obtain

$$\mathrm{KL}(q(x) \parallel p(x)) = \sum_{x \in X} \prod_i q_i(x_i) \ln \frac{\prod_i q_i(x_i)}{p(x)}. \tag{4.29}$$

By isolating the factor $q_j(x_j)$ we have

$$\mathrm{KL}(q(x) \parallel p(x)) = \sum_{x \in X} q_j(x_j) \prod_{i \neq j} q_i(x_i) \ln \frac{q_j(x_j) \prod_{i \neq j} q_i(x_i)}{p(x)}, \tag{4.30}$$

allowing to write

$$\mathrm{KL}(q(x) \parallel p(x)) = \sum_{x \in X} q_j(x_j) \prod_{i \neq j} q_i(x_i) \left(\ln q_j(x_j) + \sum_{i \neq j} \ln q_i(x_i) - \ln p(x) \right) \tag{4.31}$$

such that

$$\begin{aligned} \mathrm{KL}(q(x) \parallel p(x)) &= const \\ &+ \sum_{x_j \in X_j} q_j(x_j) \ln q_j(x_j) - \sum_{x_j \in X_j} q_j(x_j) \sum_{x' \in X \setminus X_j} \prod_{i \neq j} q_i(x_i) \ln p(x_j, x') \end{aligned} \tag{4.32}$$

up to terms constant in $q_j(x_j)$. We abbreviate the second term through

$$\ln \overline{p}(x_j) := const + \sum_{x' \in X \setminus X_j} \prod_{i \neq j} q_i(x_i) \ln p(x_j, x'). \tag{4.33}$$

With the definition in Eq. (4.33) we see that Eq. (4.32) is the KL-divergence $\mathrm{KL}(q_i(x_i) \parallel \overline{p}(x_j))$ which is minimized if $\ln q_i(x_i) = \ln \overline{p}(x_j)$. The constant term in Eq. (4.33) needs not to be calculated explicity, since the probabilities $\sum_{x_j} p(x_j) = 1$ must sum to one which can be established through normalization. In typical inference algorithms, the local factors $q_i(x_i)$ are iteratively updated using Eq. (4.33) and renormalization. Inference results in approximations of the marginals of X_i.

In MRF and CRF models, we approximate the distributions

$$p(y \mid x) \approx q(y, x) = \prod_j q_j(y_j, x) \tag{4.34}$$

by local factors $q_j(y_j, x)$ for each image site, in which the observations x are constants. Combining the MRF model in Eq. (4.16) with the solution for the

factorized approximation in Eq. (4.33) yields

$$\begin{aligned}
&\ln q_j(y_j,x) = const + \sum_{y'\in Y\setminus Y_j} \prod_{i\neq j} q_i(y_i,x) \ln p(y_j,y' \mid x) = const \\
&+ \sum_{y'} \prod_{i\neq j} q_i(y_i,x) \left(\ln p(y_j \mid x) + \sum_{(y_j,\overline{y})\in\mathcal{C}_P(y_j)} \ln p(y_j,\overline{y}) + \sum_{(y,\overline{y})\in\mathcal{C}_P\setminus\mathcal{C}_P(y_j)} \ln p(y,\overline{y}) \right) \\
&= const + \ln p(y_j \mid x) + \sum_{(y_j,y_i)\in\mathcal{C}_P(y_j)} \sum_{y_i\in Y_i} q_i(y_i,x)\ \ln p(y_j,y_i)
\end{aligned} \tag{4.35}$$

where $\mathcal{C}_P(y_j)$ is the set of cliques that involve y_j. In this derivation we exploit the normalization of the local factors and that terms independent of y_j can be subsumed in a constant. For the CRF model, analogous derivation yields

$$\ln q_j(y_j,x) = const + \ln \varphi(y_j,x) + \sum_{(y_j,y_i)\in\mathcal{C}_P(y_j)} \sum_{y_i\in Y_i} q_i(y_i,x)\ \ln \varphi(y_j,y_i,x). \tag{4.36}$$

This local update scheme is illustrated in Fig. 4.3.

Note the exponential updates in contrast to the LBP updates (Sec. 4.1.2.4). Mean-field approximations provide us with an alternative method for approximate inference in MRFs and CRFs for image processing. Saito et al. (2012) recently demonstrated that variational mean-field inference yields similar accuracy like LBP but faster convergence in image labeling tasks.

4.1.2.7. Inference using Graph Cuts

Graph cuts are efficient algorithms that can be utilized for finding maximum-likelihood assignments of random variables (Boykov et al., 2001). While the inference methods introduced in previous sections only update random variables with local information, graph cuts make global decisions that potentially involve large sets of random variables. In certain cases, graph cuts are exact. Otherwise they often yield approximate algorithms with good lower bounds on the quality of the found local maxima.

Binary Labeling Problems: We initially formulate graph cuts for binary labeling problems, i.e., each random variable takes on one of two labels $\{0,1\}$. We restrict our investigation to grid-like graphical models on the random variables with pairwise potentials. The energy function of the graphical model in this case is

$$E(y) = \sum_{y_c\in\mathcal{C}_U} E_U(y_c) + \sum_{(y_c,y_c')\in\mathcal{C}_P} E_P(y_c,y_c'), \tag{4.37}$$

where we define $E_U(y_c) := -\ln \varphi(y_c)$ and $E_P(y_c,y_c') := -\ln \varphi(y_c,y_c')$.

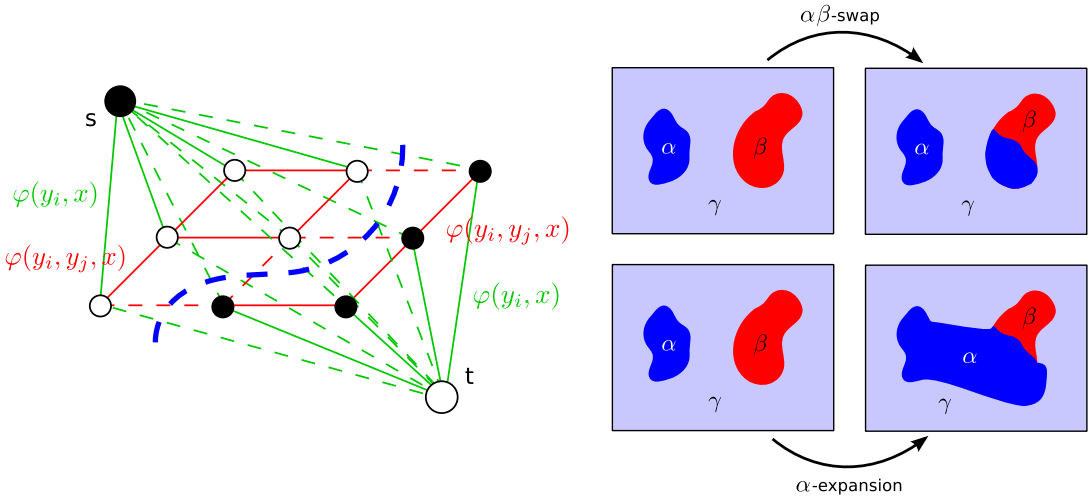

Figure 4.4.: Left: s-t-cut on a binary CRF with unary and pairwise potentials. The cost of the cut is the sum of the log potentials of the cut edges (dashed lines). Right: Possible moves for $\alpha\beta$-swap and α-expansion.

Graph cuts represent this energy function in an s/t-graph $G_{s/t} = \left(\mathcal{V}_{s/t}, \mathcal{E}_{s/t}\right)$ (Boykov and Veksler, 2006) (see Fig. 4.4). The nodes $\mathcal{V}_{s/t} := \mathcal{V} \cup \{s,t\}$ contain the image sites $\mathcal{V}$ and a source s and a sink t. The directed edges $\mathcal{E}_{s/t}$ connect nodes for image sites y_c, y'_c that appear in pairwise potentials $(y_c, y'_c) \in \mathcal{C}_P$. In addition, each image site node is connected from the source and to the sink. Each edge $(v, v') \in \mathcal{E}_{s/t}$ is assigned a weight: Edges for pairwise potentials have weight $w(y_c, y'_c) := -\ln \varphi(y_c, y'_c)$, while weights of edges of nodes with source or sink are set to the unary potentials, i.e., $w(s, y_c) := -\ln \varphi(y_c = 1)$ and $w(y_c, t) := -\ln \varphi(y_c = 0)$.

We denote a partitioning $\mathcal{S} \cup \mathcal{T}, \mathcal{S} \cap \mathcal{T} = \emptyset$ of the nodes in $G_{s/t}$ such that $s \in \mathcal{S}$ and $t \in \mathcal{T}$ an s/t-cut. Each s/t-cut can be assigned a cost that is determined by the weights of the edges that it "cuts", i.e., edges $(v, v') \in \mathcal{E}_{s/t}$ such that v and v' are not within the same set $\mathcal{S}$ or $\mathcal{T}$. A binary labeling directly corresponds to an s/t-cut by assigning $\forall y_i \in \mathcal{S} : y_i = 0$ and $\forall y_i \in \mathcal{T} : y_i = 1$. It can be shown (Boykov et al., 2001) that a minimum energy labeling of the image sites can be found by determining an s/t-cut with minimum cost, referred to as min-cut. Finding a min-cut is equivalent to the max-flow problem of determining a flow from source to sink with maximum edge weights. Several polynomial time algorithms exist for the min-cut max-flow problem (e.g., see (Cook et al., 1998)). For the graph cut optimization to be exact, i.e., to be guaranteed to find a global optimum of the energy function, the pairwise potentials need to be regular (Kolmogorov and Zabih, 2004), i.e.,

$$E_P(0,0) + E_P(1,1) \leq E_P(0,1) + E_P(1,0). \tag{4.38}$$

Multi-Label Problems: Boykov et al. (2001) proposed the α-expansion and the α-β-swap algorithms to find approximate solutions for multi-label problems in polynomial time which are in general NP-hard. In multi-label problems, each image site $i \in \mathcal{I}$ is assigned a label y_i from a set $\mathcal{L} = \{l_1, \ldots, l_M\}$ of M labels. We consider the same type of energy functions as in Eq. (4.37) with unary and pairwise potentials.

The α-expansion algorithm performs expansion moves which allow to replace the label at any site with a specific label α. In the α-β-swap algorithm, swap moves only consider sites that are labeled α or β. The labeling of these sites can be exchanged arbitrarily. Note that multiple sites can be reassigned within one move in both schemes. Both algorithms then iterate randomly through labels α (expansion) or pairs of labels α and β (swap) and determine a move that best reduces the energy in Eq. (4.37). Boykov et al. (2001) showed that optimal expansion and swap moves can be found by transforming the optimization problem to binary labeling problems and finding an s/t-min-cut.

For the optimality of the moves, however, the energy function is required to satisfy certain conditions. α-expansion finds a local optimum, if the pairwise potentials define a metric, i.e., for all α, β, γ

$$E_P(\alpha,\beta) \geq 0 \qquad \text{(non-negativity)} \tag{4.39}$$

$$E_P(\alpha,\beta) = 0 \Leftrightarrow \alpha = \beta \qquad \text{(coincidence)} \tag{4.40}$$

$$E_P(\alpha,\beta) = E_P(\beta,\alpha) \qquad \text{(symmetry)} \tag{4.41}$$

$$E_P(\alpha,\beta) \leq E_P(\alpha,\gamma) + E_P(\gamma,\beta) \qquad \text{(triangle inequality)} \tag{4.42}$$

Swap moves converge to a local optimum, if the pairwise potentials are semi-metrics, i.e., at least Eqs. (4.39), (4.40), and (4.41) hold. These conditions are equivalent to the submodularity of the modified energies $E'(y)$ (Ramalingam et al., 2008) that are optimized by the α-expansion and the α-β-swap algorithms. Note, that while the individual moves are optimal, graph cuts only find approximate solutions, i.e., local optima, for multi-label problems. Due to the global optimization used for the moves, graph cuts often find better local optima than algorithms based on local update schemes such as the max-sum-algorithm (Sec. 4.1.2.4). For α-expansion the found locally optimal energy can be bound to a constant factor from the global optimum (Boykov et al., 2001).

Label Costs: The pairwise potentials act as a smoothness regularizer that prefer coherent segments. Delong et al. (2012) introduced the concept of label costs into the graph cut optimization framework to also trade-off model complexity by incurring costs for using a label. The energy function in Eq. (4.37) is augmented with additional label-cost terms,

$$E(y) = \sum_{y_c \in \mathcal{C}_U} E_U(y_c) + \sum_{(y_c, y_c') \in \mathcal{C}_P} E_P(y_c, y_c') + \sum_{L \subseteq \mathcal{L}} E_L(L), \tag{4.43}$$

where

$$E_L(L) := \begin{cases} h_L & \text{if } \exists y_i : y_i \in L, \\ 0 & \text{otherwise.} \end{cases} \tag{4.44}$$

Delong et al. (2012) propose modifications of the α-expansion and the α-β-swap algorithms to consider label costs. In the case of α-expansion with label costs, optimality bounds worsen with strength of the label cost terms, and run-time efficiency empirically decreases by about 40% to 60% compared to standard α-expansion. Both algorithms still provide very efficient algorithms that converge to good solutions for several problems in practice. Remarkably, label costs can be related with the Akaike information criterion (AIC) (Akaike, 1974) and the Bayesian information criterion (BIC) (Schwarz, 1978) which are principled Bayesian methods to trade-off model complexity.

Finally, we note that efficient graph cuts for higher-order or non-submodular potentials are an active research topic (see e.g.,(Ramalingam et al., 2008; Kolmogorov and Rother, 2007; Fix et al., 2011)).

4.2. Efficient Rigid Multi-Body Registration of RGB-D Images

Our approach to rigid multi-body registration segments moving rigid parts between two RGB-D images, i.e., it determines the number of rigid parts, their 3D rigid-body motion, and the image regions that map the parts. We assume that an image $I = (x_i, \ldots, x_N)$ is partitioned into a set of discrete sites i with observations x_i such as pixels or map elements in a 3D representation. Let $Y = Y_1 \times \ldots \times Y_N$ be the labeling domain of the image sites. The concrete labeling $y_i \in Y_i = \mathcal{L} := \{O, 1, \ldots, M\}$ denotes the membership of site i in one of the distinct motion segments $\mathcal{M} = \{m_k\}_{k=1}^{M}$ or in the set of outliers O. With $y \in Y$, we denote a concrete labeling of the whole image. All sites within a segment move with a common six degree-of-freedom (6-DoF) rigid-body motion $\theta_k \in \mathbb{SE}(3)$ between the segmented image I_{seg} and a reference image I_{ref}.

4.2.1. An Expectation-Maximization Framework for Dense 3D Motion Segmentation of Rigid Parts

We explain the segmented image by the rigid-body motion of segments towards the reference image, i. e., we seek rigid-body motions $\Theta = \{\theta_k\}_{k=1}^{M}$ that maximize the observation likelihood of the segmented image in the reference image:

$$\arg\max_{\Theta} p(I_{seg} \mid \Theta, I_{ref}). \tag{4.45}$$

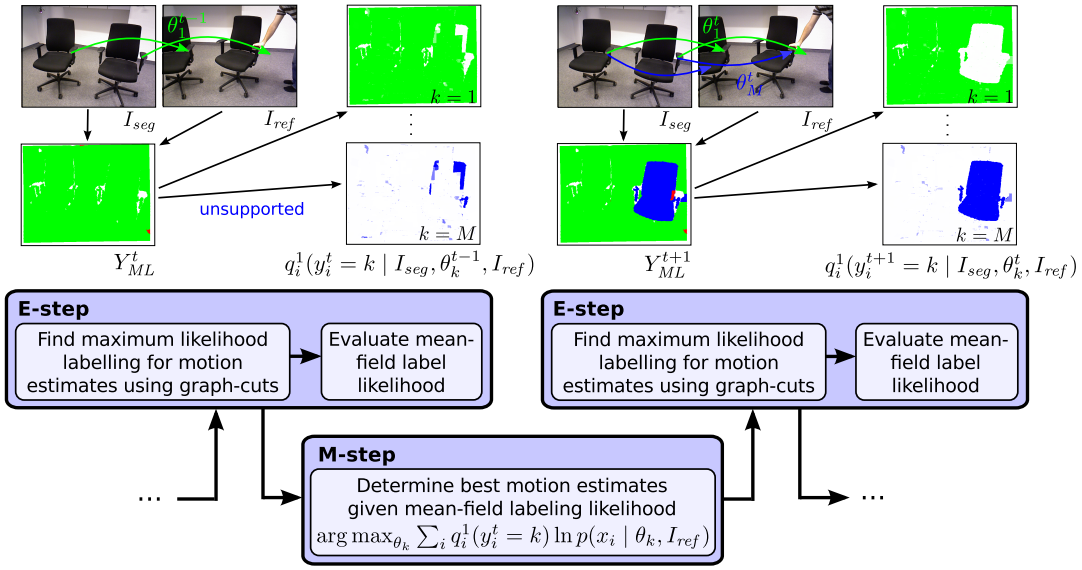

Figure 4.5.: We segment motion in an RGB-D image I_{seg} towards a reference image I_{ref} in an efficient expectation-maximization framework. In the E-step, we evaluate the likelihood of image site labels y_i under the latest motion estimates θ_k. Efficient graph cuts yield a maximum likelihood labeling y_{ML} given the motion estimates, which is then used to approximate the label likelihoods. In the M-step, new motion estimates for each segment are found through image registration which takes the soft assignment of sites to labels into account.

In our formulation, the labeling of the image sites is a latent variable that we estimate jointly with the rigid-body motions of the segments using EM (see Sec. 4.1.1),

$$\arg\max_{\Theta} \sum_{y \in Y} p(y \mid I_{seg}, \overline{\Theta}, I_{ref}) \ \ln p(I_{seg}, y \mid \Theta, I_{ref}). \tag{4.46}$$

where $\overline{\Theta}$ is the latest motion estimate of the segments from the previous iteration of the EM algorithm, and $p(y \mid I_{seg}, \overline{\Theta}, I_{ref})$ is the posterior distribution of the image labeling. Our EM approach is illustrated in Fig. 4.5. We further factorize

$$p(I_{seg}, y \mid \Theta, I_{ref}) = p(I_{seg} \mid y, \Theta, I_{ref}) p(y \mid \Theta, I_{ref}). \tag{4.47}$$

If we assume a uniform prior $p(y \mid \Theta, I_{ref})$ over labelings without knowing the image content, we can formulate our EM-objective as

$$\arg\max_{\Theta} \sum_{y \in Y} p(y \mid I_{seg}, \overline{\Theta}, I_{ref}) \ \ln p(I_{seg} \mid y, \Theta, I_{ref}). \tag{4.48}$$

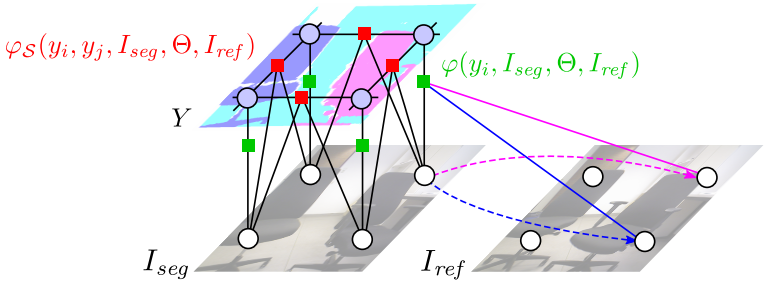

Figure 4.6.: We model the likelihood of an image labeling in a CRF with unary and pairwise potentials. The unary potentials measure the likelihood of observation between segmented and reference image under the motion estimate of a label. The pairwise potentials penalize differing labelings between image sites with low contrast and curvature.

The EM algorithm alternates the following two steps in several iterations until convergence, or until a maximum number of iterations is reached:

E-step: Determine the posterior distribution of the image labeling given the latest motion estimates $\overline{\Theta}$ to form the conditional expectation in (4.46).

M-step: Find new motion estimates Θ by maximizing the conditional expectation (4.46), given the posterior distribution of the image labeling.

4.2.2. Image Labeling Posterior

We model the likelihood of an image labeling y in a CRF (see Sec. 4.1.2.3)

$$p(y \mid I_{seg}, \Theta, I_{ref}) = \prod_{y_i \in \mathcal{C}_U} \varphi(y_i, I_{seg}, \Theta, I_{ref}) \prod_{(y_i, y_j) \in \mathcal{C}_P} \varphi(y_i, y_j, I_{seg}, \Theta, I_{ref}), \quad (4.49)$$

where $\varphi(y_i, I_{seg}, \Theta, I_{ref})$ are unary potentials on the image sites i, and the pairwise potentials $\varphi(y_i, y_j, I_{seg}, \Theta, I_{ref})$ model interactions between image sites i and j (see Fig. 4.6).

Unary Potentials: The unary potentials are given by the observation likelihood

$$\varphi(y_i, I_{seg}, \Theta, I_{ref}) := p(x_i \mid y_i, \Theta, I_{ref}) = p(x_i \mid \theta_{y_i}, I_{ref}), \quad (4.50)$$

which quantifies the likelihood to observe $x_i \in I_{seg}$ in I_{ref} under the motion estimate θ_{y_i} for label y_i. For the outlier label $l_i = O$, we set the observation likelihood to a constant p_O.

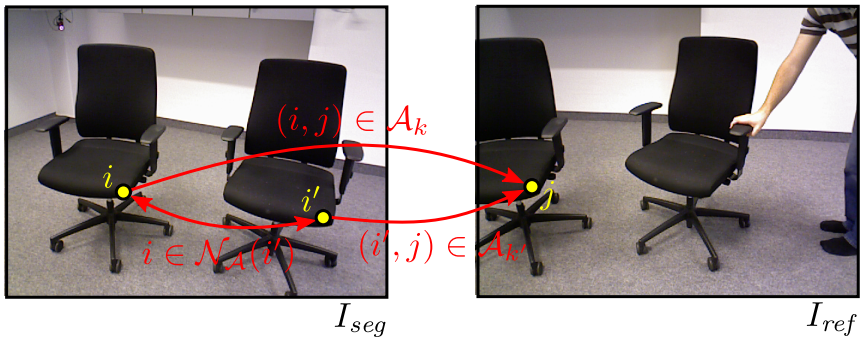

Figure 4.7.: Ambiguity resolution. If sites i and i' associate with the same site j in the reference image for motion segments k and k' (i.e. $(i,j) \in \mathcal{A}_k$ and $(i',j) \in \mathcal{A}_{k'}$), we include additional pairwise CRF terms between them. The likelihood of the assignment of both sites to labels k and k' is set to a small value (large negative log-likelihood α).

Pairwise Smoothness Potentials: Between direct neighbors i and j in the image representation, we use a contrast-sensitive Potts model (Boykov and Jolly, 2001)

$$\ln \varphi_{\mathcal{S}}(y_i, y_j, I_{seg}) = -\gamma(x_i, x_j)\ \delta(y_i, y_j), \tag{4.51}$$

where we define

$$\delta(y_i, y_j) := \begin{cases} 0 & ,\text{if } y_i = y_j, \\ 1 & ,\text{if } y_i \neq y_j, \end{cases} \tag{4.52}$$

and $\gamma(x_i, x_j) > 0$ controls the strength of the coupling in dependence on the difference between the observations at the image sites. We denote the set of cliques between direct neighbors i, j by $\mathcal{C}_{P,\mathcal{S}}$.

Pairwise Disambiguation Potentials: We also need to avoid multiple associations of image sites in the segmented image with the same image site in the reference image (see Fig. 4.7). Otherwise, our approach could explain different parts of the segmented image with the same part in the reference image, e.g., at missing image overlap or in occluded regions.

The image site labelings decide for an association of sites between both images. In order to prevent the graph cut optimization from establishing labelings that would associate multiple times to a site in the reference image, we introduce additional pairwise couplings. We consider sites i and j in the segmented image that map to the same site in the reference image for different motion segments k and k', respectively. We define the pairwise potential

$$\ln \varphi_{\mathcal{A}}(y_i, y_j) := \begin{cases} -\alpha & \text{if } y_i = k \wedge y_j = k' \\ 0 & \text{otherwise,} \end{cases} \tag{4.53}$$

where α sets the strength of the couplings. We refer to the set of sites with the same association like site i by $\mathcal{N}_{\mathcal{A}}(i)$ and denote the induced coupled pairs by

$$\mathcal{C}_{P,\mathcal{A}} := \{(y_i, y_j) \mid i \in \mathcal{N}_{\mathcal{A}}(j)\}. \tag{4.54}$$

In the CRF model, we use both types of pairwise couplings $\mathcal{C}_P = \mathcal{C}_{P,\mathcal{S}} \cup \mathcal{C}_{P,\mathcal{A}}$ concurrently to enforce spatial coherence and to handle ambiguous associations. The combined potential is

$$\varphi(y_i, y_j, I_{seg}, \Theta, I_{ref}) = \begin{cases} \varphi_{\mathcal{S}}(y_i, y_j, I_{seg}) & \text{if } (y_i, y_j) \in \mathcal{C}_{P,\mathcal{S}} \\ \varphi_{\mathcal{A}}(y_i, y_j) & \text{otherwise.} \end{cases} \tag{4.55}$$

4.2.3. Efficient Approximate Solution of the Expectation-Maximization Formulation

We propose an efficient approximate solution to the EM formulation. Firstly, we see that the observation likelihood of the segmented image in the reference image given motion estimates and labeling,

$$p(I_{seg} \mid \Theta, I_{ref}, y), \tag{4.56}$$

factorizes into the likelihood of the individual observations

$$p(I_{seg} \mid \Theta, I_{ref}, y) = \prod_{i=1}^{N} p(x_i \mid \theta_{y_i}, I_{ref}) \tag{4.57}$$

since we assume stochastic independence between the observations and each site is associated to exactly one segment given a specific labeling y. By this, Eq. (4.46) becomes

$$\arg\max_{\Theta} \sum_{y \in Y} p(y \mid I_{seg}, \overline{\Theta}, I_{ref}) \sum_{i=1}^{N} \ln p(x_i \mid \theta_{y_i}, I_{ref}). \tag{4.58}$$

Note that each term of the inner sum only depends on one of the image labels.

Since exact inference of the joint label likelihood $p(y \mid I_{seg}, \overline{\Theta}, I_{ref})$ in a CRF is not tractable even for a single labeling y, we need to resort to approximations. One possible crude approach would be to use inference algorithms such as LBP (Sec. 4.1.2.4) to infer the marginal distribution over site labelings $p(y_i \mid I_{seg}, \overline{\Theta}, I_{ref})$, and to optimize

$$\arg\max_{\Theta} \sum_{y \in Y} \sum_{i=1}^{N} p(y_i \mid I_{seg}, \overline{\Theta}, I_{ref}) \ln p(x_i \mid \theta_{y_i}, I_{ref}). \tag{4.59}$$

We apply a mean-field approximation (Sec. 4.1.2.6) to the joint label likelihood

$$p(y \mid I_{seg}, \overline{\Theta}, I_{ref}) \approx \prod_{i=1}^{N} q_i(y_i \mid I_{seg}, \overline{\Theta}, I_{ref}) \tag{4.60}$$

to write

$$\arg\max_{\Theta} \sum_{y_1 \in Y_1} \dots \sum_{y_N \in Y_N} \left(\prod_{i=1}^{N} q_i(y_i \mid I_{seg}, \overline{\Theta}, I_{ref}) \right) \left(\sum_{i=1}^{N} \ln p(x_i \mid \theta_{y_i}, I_{ref}) \right) \tag{4.61}$$

in a principled way. Rearranging terms yields

$$\arg\max_{\Theta} \sum_{i=1}^{N} \sum_{y_i \in Y_i} q_i \ln p(x_i \mid \theta_{y_i}, I_{ref}) \cdot \sum_{y_1 \in Y_1} q_1 \dots \sum_{y_{i-1} \in Y_{i-1}} q_{i-1} \sum_{y_{i+1} \in Y_{i+1}} q_{i+1} \dots \sum_{y_N \in Y_N} q_N, \tag{4.62}$$

where we use the shorthand $q_i := q_i(y_i \mid I_{seg}, \overline{\Theta}, I_{ref})$. Since the factors are normalized such that $\sum_{y_i \in Y_i} q_i(y_i \mid I_{seg}, \overline{\Theta}, I_{ref}) = 1$ (Sec. 4.1.2.6), we arrive at

$$\arg\max_{\Theta} \sum_{i=1}^{N} \sum_{y_i \in Y_i} q_i(y_i \mid I_{seg}, \overline{\Theta}, I_{ref}) \ln p(x_i \mid \theta_{y_i}, I_{ref}), \tag{4.63}$$

which is equivalent to

$$\arg\max_{\Theta} \sum_{k=0}^{M} \sum_{i=1}^{N} q_i(y_i = k \mid I_{seg}, \overline{\Theta}, I_{ref}) \ln p(x_i \mid \theta_k, I_{ref}). \tag{4.64}$$

In the E-step, the factors $q_i(y_i \mid I_{seg}, \overline{\Theta}, I_{ref})$ are estimated in an iterative process $q_i^{t-1} \rightsquigarrow q_i^t$ using Eq. (4.36). Since this process only performs local updates, the quality of the found local optimum strongly depends on the initial estimate $q_i^0(y_i)$. We therefore initialize the mean-field iterations with a ML-solution found by graph cuts (Sec. 4.1.2.7)

$$y_{ML} = \arg\max_{y \in Y} p(y \mid I_{seg}, \overline{\Theta}, I_{ref}) \tag{4.65}$$

such that

$$q_i^0(y_i \mid I_{seg}, \overline{\Theta}, I_{ref}) = \begin{cases} 1 & \text{if } y_i = y_{i,ML} \\ 0 & \text{otherwise.} \end{cases} \tag{4.66}$$

Due to the pairwise ambiguity-resolving potentials, the pairwise potentials define a semi-metric, since transitivity is not satisfied. While α-expansions require the pairwise potentials to be a metric, $\alpha\beta$-swaps are applicable for semi-metrics (see Sec. 4.1.2.7).

For an efficient algorithm, we are not required to run the mean-field iterations until convergence. A single iteration suffices to improve the estimate for $p(y \mid I_{seg}, \overline{\Theta}, I_{ref})$, which also improves the lower bound of the EM-algorithm. As we use graph cuts to seed the iterations, we typically obtain good solutions within a few cycles of EM by reducing the KL-divergence between $p(y \mid I_{seg}, \overline{\Theta}, I_{ref})$ and our approximation. We observe that according to Eq. (4.36), after a single iteration the factors are

$$q_i^1(y_i \mid I_{seg}, \overline{\Theta}, I_{ref}) = \eta_i \exp\Bigg(\ln p(x_i \mid y_i, \Theta, I_{ref}) + \sum_{(y_i,y_j)\in\mathcal{C}_P(y_i)} \sum_{y_j \in Y_j} q_j^0(y_j \mid I_{seg}, \overline{\Theta}, I_{ref}) \ \ln \varphi(y_i, y_j, I_{seg}) \Bigg), \quad (4.67)$$

where η_i is a normalization factor such that $\sum_{y_i \in Y_i} q_i^1(y_i \mid I_{seg}, \overline{\Theta}, I_{ref}) = 1$. Plugging our ML-seed (Eq. (4.66)) into Eq. (4.67) yields

$$q_i^1(y_i \mid I_{seg}, \overline{\Theta}, I_{ref}) = \eta_i \ p(x_i \mid y_i, \Theta, I_{ref}) \prod_{(y_i,y_j)\in\mathcal{C}_P(y_i)} \varphi(y_i, y_{j,ML}, I_{seg}). \quad (4.68)$$

Interestingly, the factors $q_i^1(y_i \mid I_{seg}, \overline{\Theta}, I_{ref})$ are local conditional probabilities

$$q_i^1(y_i \mid I_{seg}, \overline{\Theta}, I_{ref}) = p(y_i \mid y_{ML} \setminus \{y_i\}, I_{seg}, \overline{\Theta}, I_{ref}) \quad (4.69)$$

in the CRF conditioned on the ML-solution. Note that if the graph-cuts avoid ambiguous associations, the corresponding pairwise terms vanish from eq. (4.68).

In summary, each image site i is assigned a weight for the reestimation of the rigid-body motion θ_k in Eq. (4.64). The weight intuitively is the likelihood that site i belongs to the segment with respect to the ML-labeling.

4.2.4. Model Complexity

The pairwise interaction terms prefer large motion segments and naturally control the number of segments to be small. In the case that a single 3D motion segment occurs as multiple unconnected image segments in the image, our approach so far may still use different but redundant motion segments for the image segments. To control model complexity, we enhance the graph cut optimization in Sec. 4.2.3 with label costs (Delong et al., 2012), i.e., we use graph cuts to optimize the augmented CRF energy function

$$E(y) = - \sum_{y_i \in \mathcal{C}_U} \ln \varphi(y_i, I_{seg}, \Theta, I_{ref}) - \sum_{(y_i,y_j)\in\mathcal{C}_P} \ln \varphi(y_i, y_j, I_{seg}, \Theta, I_{ref}) - \sum_{l\in\mathcal{L}} \ln \varphi(l, y), \quad (4.70)$$

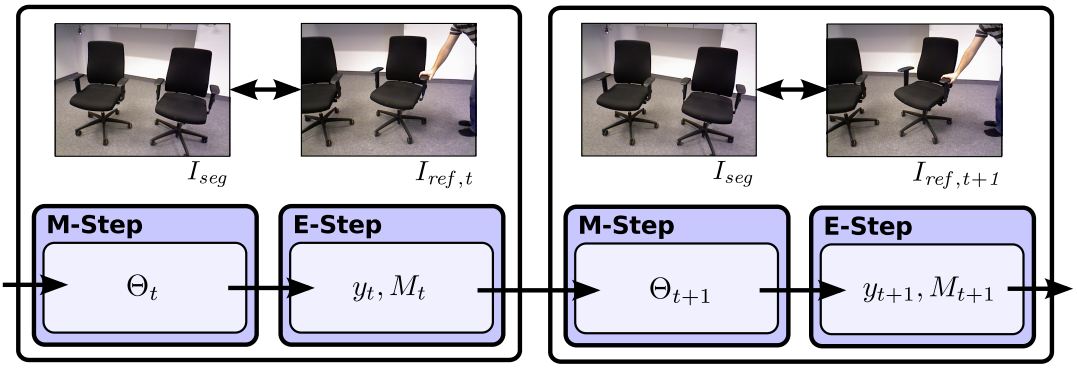

Figure 4.8.: Online EM. The EM framework is used to segment RGB-D images online by performing a few M- and E-steps per image. Typically, one iteration per image suffices.

with per-label-costs

$$\ln \varphi(l, y) := \begin{cases} -\lambda & \text{if } l \neq 0 \wedge \exists y_i \in y : y_i = l \\ 0 & \text{otherwise.} \end{cases} \tag{4.71}$$

Each label is assigned the same cost λ except the outlier label for which we impose no cost. Label costs have a natural interpretation of implementing information criteria such as the BIC (see Sec. 4.1.2.7).

We initialize the EM algorithm with a guess of the number of motion segments ($M = 1$ in our experiments). While this guess influences the number of required iterations, we found that it has only little effect on finding the correct number of segments. To let our approach possibly increase the number of segments, we append one additional, yet unsupported segment before the M-step. All sites in segments that are yet unsupported in the image are assigned the outlier data likelihood p_O. By this, our EM algorithm prefers to explain sites that misalign with the already existing segments by new motion segments. We define a motion segment to be supported if it labels sites in the image and reject very small segments as outliers. Unsupported segments (eventually the additional segment) are discarded after the E-step.

4.2.5. Sequential Segmentation

While our EM formulation may in principle segment motion between arbitrary images, we augment it to perform efficiently on image sequences. We segment the first image I_{seg} in a sequence iteratively towards subsequent images $I_{ref,t}$. At each new image at time t, our approach estimates the number of segments M_t, a new segmentation y_t, and new motion estimates Θ_t. Instead of starting our EM procedure all over for each new image, we initialize the approach with the

estimates from the last image $I_{ref,t-1}$. This way, the EM algorithm requires significantly less iterations per image to converge (typically one iteration suffices).

4.2.6. Image Representation

The performance of our EM approach depends on the underlying image representation. Any representation is suitable that defines observation likelihood $p(x_i \mid \theta_{y_i}, I_{ref})$, image site neighborhood $\mathcal{C}_{P,\mathcal{S}}$, and dissimilarity $\gamma(x_i, x_j)$ for the pairwise interaction terms. To solve for the motion estimates of the segments in Eq. (4.64), an image registration technique is required that allows to incorporate individual weights for the image sites. To these ends, our compact MRSMaps are an efficient choice.

4.2.6.1. Observation Likelihood

We interpret voxels x in the MRSMap as image sites. Given the labeling y_i, the surfel $s_{seg,i}$ in voxel $x_{seg,i}$ is observed at a corresponding surfel $s_{ref,j}$ in voxel $x_{ref,j}$ under the rigid-body motion estimate θ_{y_i}, i.e., we model the observation likelihood

$$\begin{aligned} p(s_{seg,i}|\theta_{y_i}, s_{ref,j}) &= \mathcal{N}\left(d^*(s_{seg,i}, s_{ref,j}, \theta_{y_i}); 0, \Sigma^*(s_{seg,i}, s_{ref,j}, \theta_{y_i})\right), \\ d^*(s_{seg,i}, s_{ref,j}, \theta_{y_i}) &:= \mu_{ref,j} - (R^*(\theta_{y_i})\mu_{seg,i} + t^*(\theta_{y_i})), \\ \Sigma^*(s_{seg,i}, s_{ref,j}, \theta_{y_i}) &:= \Sigma'_{ref,j} + R^*(\theta_{y_i})\Sigma'_{seg,i}R^*(\theta_{y_i})^T, \end{aligned} \tag{4.72}$$

If multiple surfels are contained within the voxels i and j for several view directions, we assign the best observation likelihood among all pairs of view directions. Here, we take spatial as well as color information into account such that

$$R^*(\theta_{y_i}) = \begin{pmatrix} R(\theta_{y_i}) & 0 \\ 0 & I_3 \end{pmatrix} \in \mathbb{R}^{6\times 6} \tag{4.73}$$

rotates the surfel coordinates according to the motion estimate, and

$$t^*(\theta_{y_i}) = \begin{pmatrix} t(\theta_{y_i}) \\ 0 \end{pmatrix} \in \mathbb{R}^{6\times 1} \tag{4.74}$$

is the corresponding translation. Correlations between the point and color distributions cannot be considered since the color distribution is not comparable for large spatial misalignments at which surface has not been measured. We hence remove these correlations by setting the corresponding entries in the surfel covariances $\Sigma'_{ref,j}$ and $\Sigma'_{seg,i}$ to zero. Furthermore, in order to improve robustness for illumination changes, we neglect small luminance and chrominance differences.

For the unary potentials, we additionally examine the consistency of the surfel normals in the combined likelihood

$$\varphi(y_i, I_{seg}, \Theta, I_{ref}) = \mathcal{N}\left(d^*(s_{seg,i}, s_{ref,j}, \theta_{y_i}); 0, \Sigma^*(s_{seg,i}, s_{ref,j}, \theta_{y_i})\right) \cdot \mathcal{N}\left(\arccos\left(n_{ref,j}, R(\theta_{y_i})\, n_{seg,i}\right), \sigma_n^2\right) \quad (4.75)$$

with standard deviation σ_n. Since the rotation around the surface normal is not observable, we do not use the term for pose optimization.

The evaluation of the observation likelihood involves the association of the surfel $s_{seg,i}$ with a surfel $s_{ref,j} = A_k(s_{seg,i})$ from the reference image. The mean position of the surfel $s_{seg,i}$ is transformed to the reference image according to the motion estimate θ_{y_i}. We then search for a matching surfel in the reference image from coarse to fine resolutions. We adapt the search radius $r = 2\,\rho(V(s_{seg,i}))^{-1}$ to the resolution and find the association on the finest resolution possible. Each motion segment requires its own set of associations

$$\mathcal{A}_k := \left\{\left(s_{seg}, s_{ref}\right) \in I_{seg} \times I_{ref} \mid s_{ref,j} = A_k(s_{seg,i})\right\}. \quad (4.76)$$

Care has to be taken at image borders, background at depth discontinuities, and occlusions, since no association can be made and assigning a low likelihood would be pessimistic. We assign the last observed data likelihood to such surfels.

4.2.6.2. Smoothness Cost Terms

We establish pairwise terms between all six direct neighbors of a voxel in the 3D grid. In addition, we couple a voxel with its children and its parent voxel within the octree. In this way, spatial coherence can be enforced despite the sparseness of the 3D representation and across the discrete changes of the depth-dependent resolution limit. We weaken pairwise couplings by the dissimilarity of surfels,

$$\gamma(x_i, x_j) := g_s \ \min\Big\{1, \max\Big\{0, \max\Big\{g_n(1 - n_i^T n_j), \\ g_L d_L(s_i, s_j),\ g_\alpha d_\alpha(s_i, s_j), g_\beta d_\beta(s_i, s_j)\Big\} - g_0\Big\}\Big\}, \quad (4.77)$$

where g_s, g_n, g_L, g_α, and g_β are scale parameters,

$$d_L(s_i, s_j) = \left|\mu_{L,i} - \mu_{L,j}\right|, \quad (4.78)$$

$$d_\alpha(s_i, s_j) = \left|\mu_{\alpha,i} - \mu_{\alpha,j}\right|, \quad (4.79)$$

$$d_\beta(s_i, s_j) = \left|\mu_{\beta,i} - \mu_{\beta,j}\right|, \quad (4.80)$$

and g_0 handles illumination differences and noise. Fig. 4.9 illustrates our smoothness terms in an example.

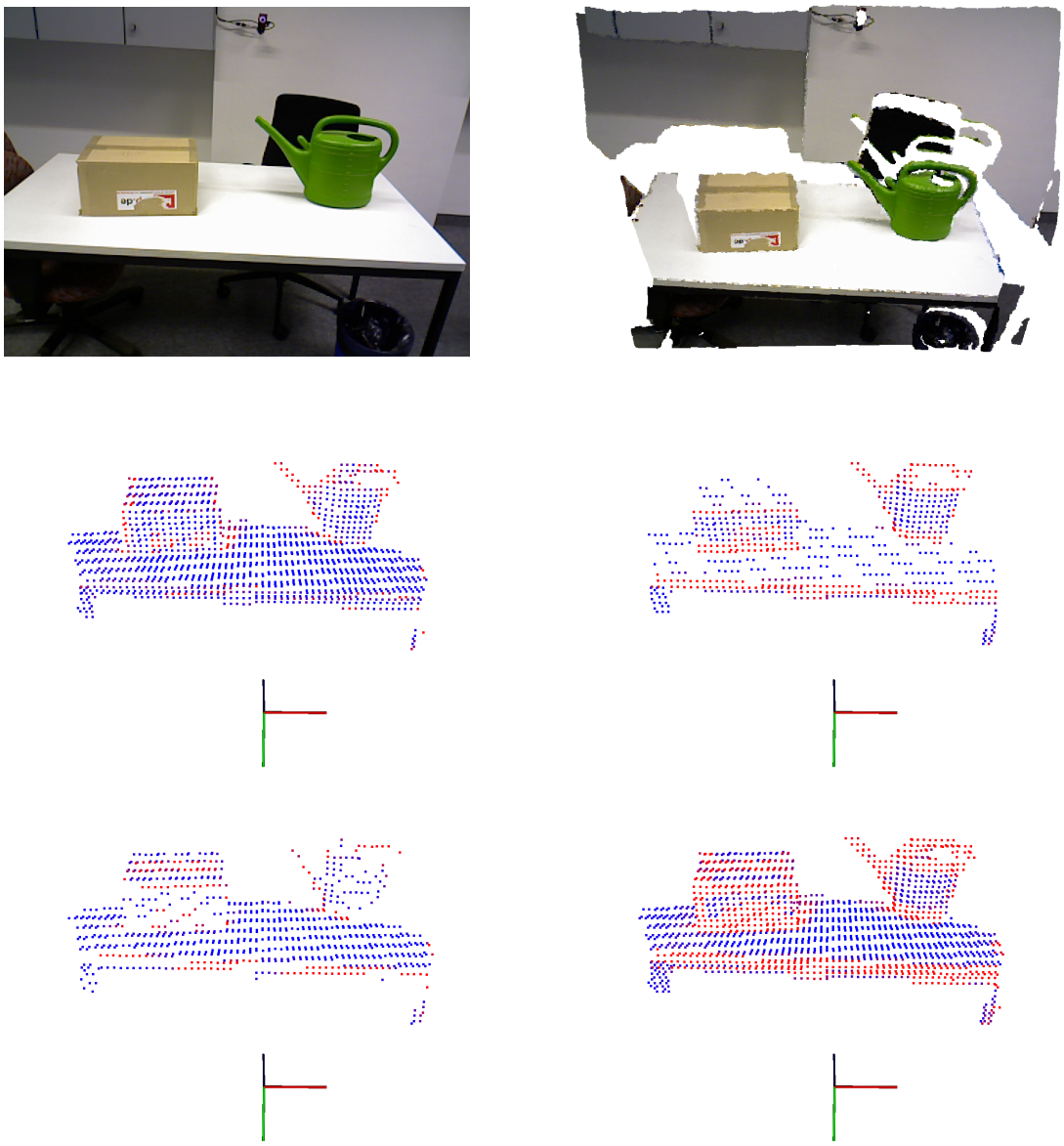

Figure 4.9.: Pairwise interactions in MRSMaps. We visualize the smoothness cost terms for direct voxel neighbors to the right (middle left), down (middle right), and forward (bottom left) directions. Directions are according to the shown camera frame (right: red, down: green, forward: blue axis). Bottom right: maximum cost over all neighbors. Costs are color-coded from blue (low) to red (high). Missing voxels either do not exist on the displayed resolution (0.025 m) or they have no valid neighbor in the specific direction.

Figure 4.10.: Example segmentations (top, outliers dark red) towards a reference image (bottom) from the test sequences (left: small, middle: medium, right: large).

4.2.6.3. Motion Estimation

The motion of the segments is estimated in the M-step. We apply our efficient rigid registration method for MRSMaps to the optimization of the EM-objective (Eq. (4.64)). We augment the algorithm to incorporate the weighting by the mean-field factors

$$\arg\max_{\theta_{y_i}} \sum_{(s_i,s_j)\in\mathcal{A}_{\theta_{y_i}}} q_i^1(y_i \mid I_{seg}, \overline{\Theta}, I_{ref}) \ln p(s_{seg,i} \mid \theta_{y_i}, s_{ref,j}). \tag{4.81}$$

This weighted log-likelihood is optimized analogous to the approach in Sec. 3.2.2.

Since this registration procedure performs local optimization, a good initialization is important. During incremental EM, parts of the scene may start to move at any time and split an existing segment. We initialize the motion estimate for yet unsupported segments m_k with an estimate of a supported segment $m_{\widehat{k}}$. We first identify which of the supported segments or the outlier set best explains m_k through

$$\widehat{k} = \arg\max_{k'\in\{0,1,\ldots,M\}} \sum_{y_i\in y_{ML}: y_{ML,i}=k'} q_i^1(y_i = k \mid I_{seg}, \overline{\Theta}, I_{ref}). \tag{4.82}$$

If this segment is not the outlier label, i.e., $\widehat{k} \neq 0$, we set $\theta_k = \theta_{\widehat{k}}$. Otherwise, we use the largest segment.

sequence	small	medium	large
run-time in ms	200.2±42.3	213.1±54.7	138.7±37.5
error in M	0.05±0.29	0.11±0.43	-0.58±1.01
avg. seg. acc.	0.95	0.94	0.63
median trans. acc. in m	0.012	0.018	0.034
median rot. acc. in rad	0.047	0.029	0.049

Table 4.1.: Mean ± standard deviation of run-time and the error in the number of segments, segmentation and motion estimate accuracy over all frames of the test sequences.

sequence	small	medium	large
error in M	-0.09±0.35	0.04±0.45	-0.43±0.92
avg. seg. acc.	0.91	0.91	0.65
median trans. acc. in m	0.013	0.020	0.030
median rot. acc. in rad	0.045	0.030	0.048

Table 4.2.: Mean ± standard deviation of the error in the number of segments, segmentation and motion estimate accuracy under real-time constraints.

4.3. Experiments

We evaluate segmentation and motion estimation accuracy of our approach on three RGB-D video sequences with ground-truth information[1]. We recorded two large objects (chairs), two medium sized objects (a watering can and a box), and two small objects (a cereal box and a tea can) (see Fig. 4.10). The objects as well as the camera have been moved during the recordings. The sequences contain 1,100 frames at 640×480 VGA resolution and at full 30 Hz frame-rate. Ground

	number of segments M			
sequence	1	2	3	4
small	139.5±15.9	181.5±27.9	232.6±36.9	–
medium	142.7±19.4	166.2±30.9	224.2±46.8	298.9±50.8
large	102.4±17.3	125.6±24.2	158.5±30.4	192.3±37.0

Table 4.3.: Mean ± standard deviation of run-time (ms) for different number of segments.

[1]available from http://www.ais.uni-bonn.de/download/rigidmultibody

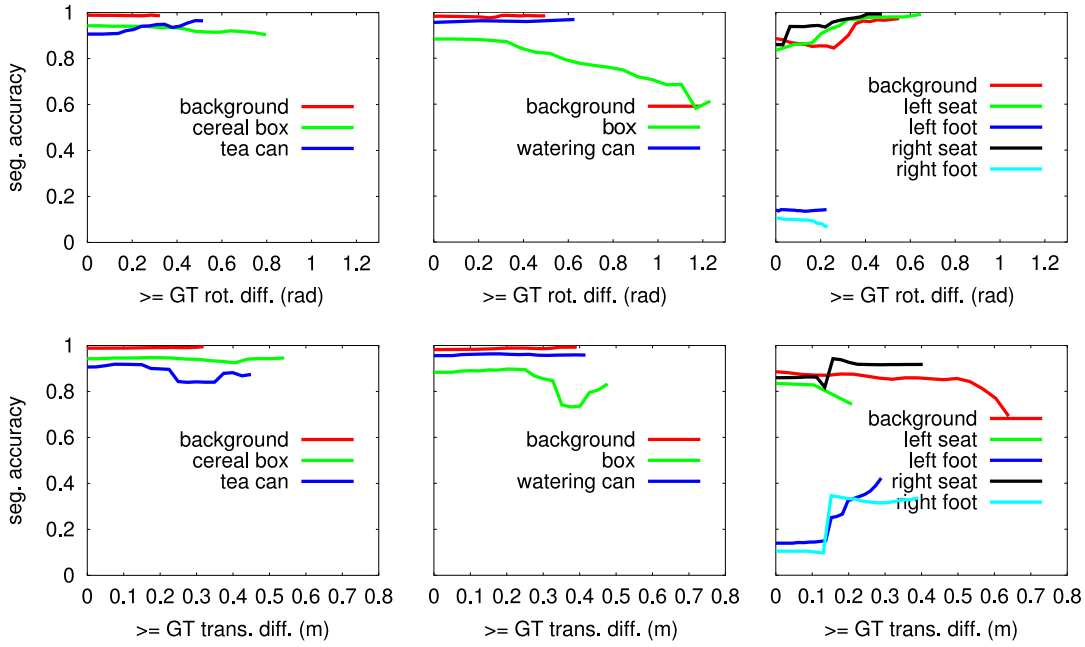

Figure 4.11.: Average segmentation accuracy vs. increasing rotational (top) and translational (bottom) ground-truth object motion (left: small, middle: medium, right: large objects). The mean is determined for segment motion greater or equal the value on the x-axis.

truth of the 3D rigid-body motion has been obtained with an OptiTrack motion capture system. We attached infrared reflective markers to the backside of the objects. While recording the data, we took care that the reflective markers were not visible for the RGB-D camera.

For frames at every 5 seconds, we manually annotated the individual object parts that move throughout the sequences. Invalid depth readings or non-rigid objects like arms and legs of persons are annotated with dont-care labels. Additionally, we set pixels to dont care in the ground truth that project outside the reference image due to camera motion. Not all annotated segments move between a ground-truth frame and an arbitrary frame in the sequence. We automatically determine groups of objects that move jointly between the frames (0.12 rad rotational and 0.05 m translational motion) and merge their segments.

The sequences are processed sequentially, starting from each ground-truth labeled image as the image to be segmented. If not stated otherwise, the sequences are processed frame-by-frame. In real-time mode, we drop frames if they would arrive during the processing of a frame. The experiments have been run on an Intel Core i7-4770K CPU at a maximum clock speed of 3.50 GHz. We determined the parameters of our approach empirically, while for the MRSMaps we use a maximum resolution of 0.0125 m at a distance dependency of $\lambda_\rho = 0.014$.

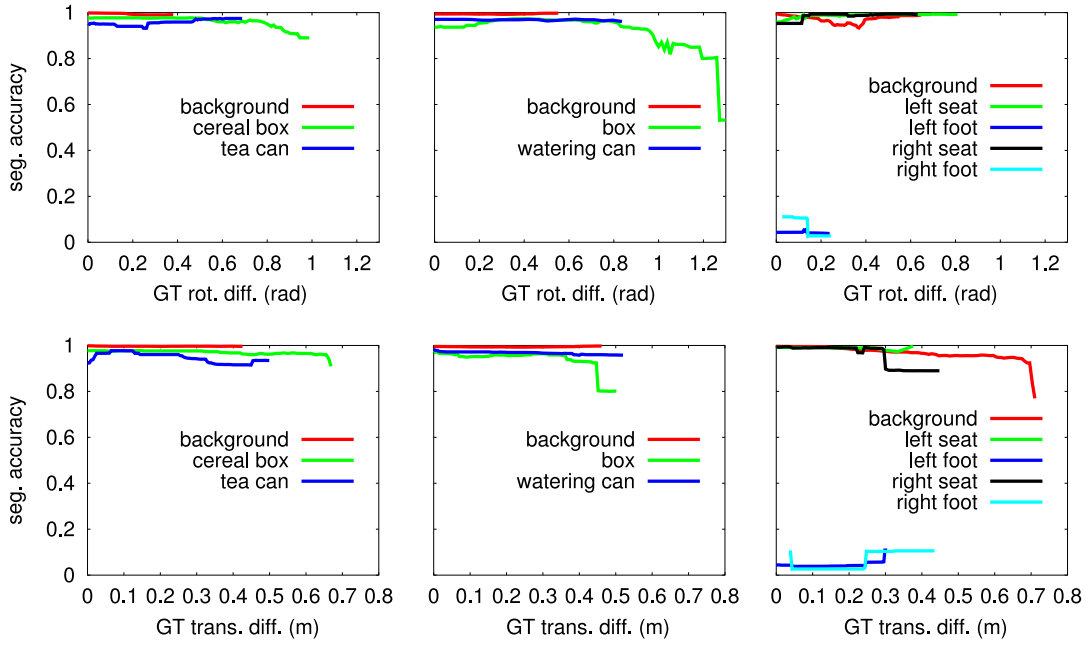

Figure 4.12.: Average segmentation accuracy vs. rotational (top) and translational (bottom) ground-truth object motion (left: small, middle: medium, right: large objects). The mean is determined in local windows of width 0.2.

4.3.1. Evaluation Measures

We quantify segmentation accuracy with the measure proposed by Everingham et al. (2010),

$$seg.\ acc. = \frac{true\ positives}{true\ pos. + false\ pos. + false\ negatives}, \tag{4.83}$$

for which we back-project the resulting motion segmentation from the MRSMaps into the segmented RGB-D images and account for the labeling of each pixel. For each object, we associate the estimated segment with highest segmentation accuracy. The average segmentation accuracy over objects in a sequence is determined by the mean over all individual object segmentation accuracies in all images. We also measure translational and rotational errors between ground-truth and estimated motion.

4.3.2. Run-Time

The run-time of our approach is given in Tables 4.1 and 4.2. It segments images fast at a frame rate of about 2 to 10 Hz. As can be seen from Table 4.3 the run-time depends on the number of segments. It also depends on the distance

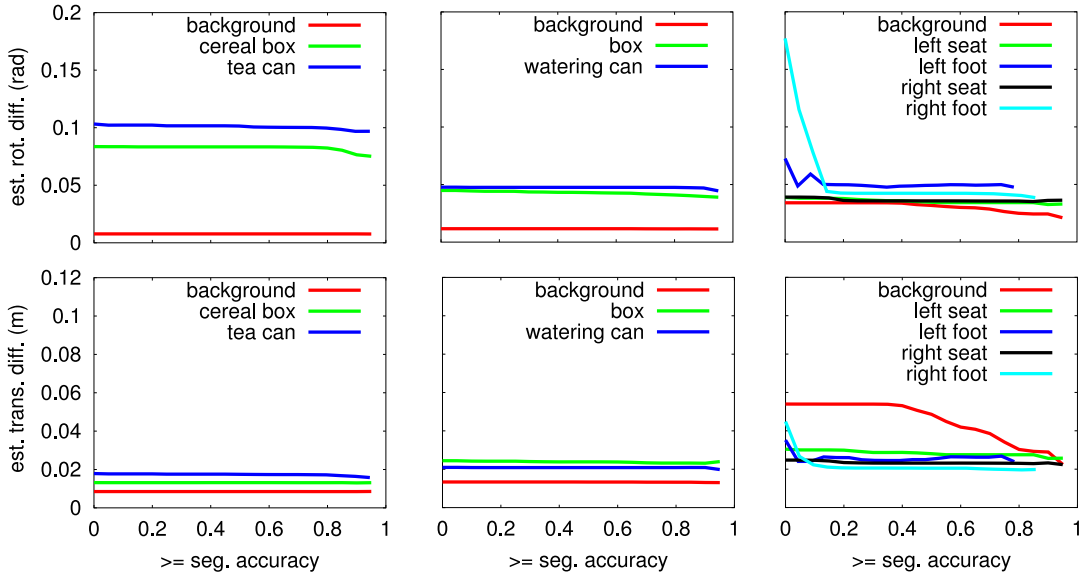

Figure 4.13.: Median rotational (top) and translational (bottom) error of the camera motion estimate vs. increasing object segmentation accuracy (left: small, middle: medium, right: large objects). The median is determined for segmentation accuracies greater or equal the value on the x-axis.

of the camera to the measured surfaces which explains the qualitative difference in run-time between the large objects sequence to the other two sequences.

4.3.3. Segmentation Accuracy

Figs. 4.11 and 4.12 show average segmentation accuracy in dependency on the actual translational and rotational motion of the objects. To visualize the effect of different degrees of object motion onto the segment accuracy, we vary a threshold for the translational and rotational motion and determine the avg. segmentation accuracy for those results for which the motion is above the threshold in Fig. 4.11. Fig. 4.12 shows the median in rotational and translational accuracy within a local window of size 0.2 in segmentation accuracy.

Most objects and the background in the sequences can be very well segmented. The box-shaped objects show a drop in segmentation accuracy with rotation since sides of the boxes become occluded. For the chairs (bottom row) it can be seen that moderate object motion facilitates high segmentation accuracy. This is explained by the distant hence noisy, structure-less, and untextured background which allows only coarse misalignments to be detected. The chair feet cannot be reliably segmented because of their thin and rotationally repetitive structure. Besides this, our approach recovers the number of segments well in the sequences, and achieves good overall accuracies in segmentation and motion estimates (see

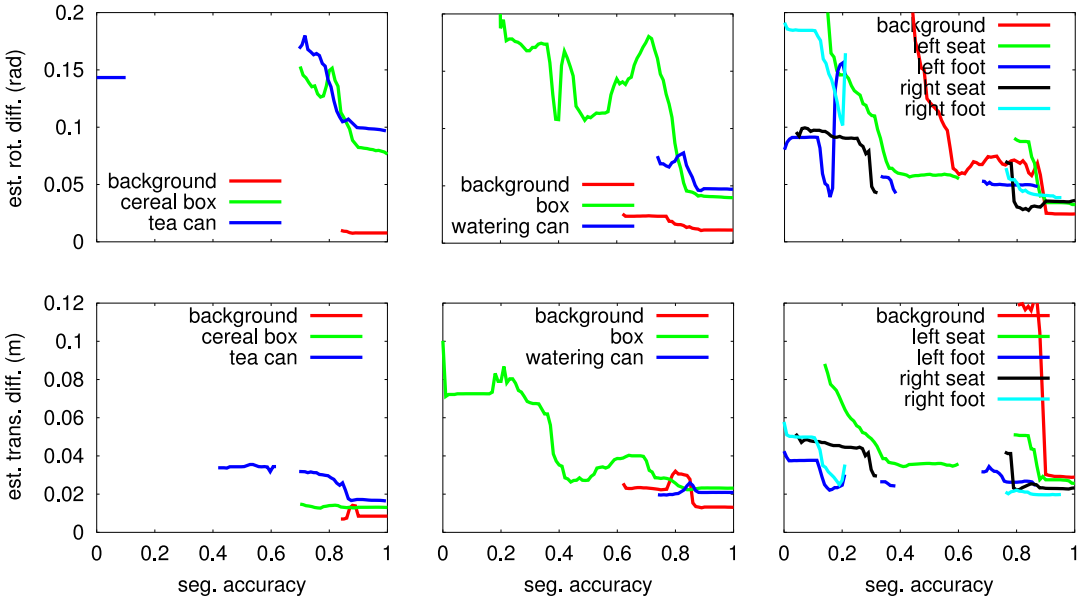

Figure 4.14.: Median rotational (top) and translational (bottom) error of the camera motion estimate vs. object segmentation accuracy (left: small, middle: medium, right: large objects). The median is determined in local windows of width 0.2.

Tables 4.1 and 4.2). Notably, if frames are dropped to operate in real-time, we obtain similar performance to processing all frames.

4.3.4. Motion Estimate Accuracy

The results in Figs. 4.13 and 4.14 demonstrate that our approach recovers camera motion towards the objects accurately. In Fig. 4.13, we determine the median pose accuracy for all results above the varied segmentation accuracy threshold, while in Fig. 4.14 we show the local median in motion accuracy in dependency of segmentation accuracy. While for many objects motion accuracy increases with segmentation accuracy, the motion also seems well estimated for low segmentation accuracies. Low segmentation accuracy often coincides with small displacements of the objects. For the small objects, or for the background at low segmentation accuracy, the pose estimates are less accurate. The small objects are difficult to track in angle with our depth-based registration method due to measurement noise and hands of persons that touch the object to move it. If the background is undersegmented, the registration arbitrates between the background and a foreground object until motion is sufficiently large to split the segment.

4.4. Related Work

Several approaches to 3D motion segmentation have been proposed that represent images sparsely through interest points. Multi-body factorization methods (Zelnik-Manor et al., 2006) find groups of points with common 3D rigid-body motion through factorization of the measurement matrix. These approaches have been extended to also cope with outliers and noisy observations (Gruber and Weiss, 2004; Schindler and Suter, 2006; Rothganger et al., 2007). Exploiting depth measurements for interest points from a calibrated stereo camera, Agrawal et al. (2005) propose a real-time capable framework for 3D motion segmentation based on RANSAC and structure-from-motion (SfM). These approaches, however, do not provide dense segmentations.

Some approaches segment 2D image motion densely based on optical flow. Cremers and Soatto (2005) propose motion competition, a variational framework for dense motion segmentation of monocular image sequences. They estimate the 2D parametric motion of multiple motion segments. Brox et al. (2006) extend this approach towards non-parametric motions. Occlusions and multiple data associations are explicitly modelled in the variational framework of Unger et al. (2012), but the method is far from real-time performance. In our approach, we also handle multiple data associations as additional pairwise labeling constraints during graph cut optimization of the motion segmentation. Kumar et al. (2005) segment scenes into 2D motion layers using a CRF model that incorporates occlusions and lighting conditions. The work by Ayvaci and Soatto (2009) defines an energy functional on a superpixel graph which is optimized using efficient graph cuts. While these methods yield impressive results, they estimate motion of 2D layers in the image and do not necessarily provide segments with consistent 3D rigid-body motion.

Weber and Malik (1997) proposed dense 3D motion segmentation between monocular images from optical flow assuming an affine camera model. Sekkati and Mitiche (2006) tackle dense 3D multibody SfM from monocular video in a variational framework and demonstrate qualitative results. Using a stereo camera, dense 3D scene flow aims at the concurrent 3D reconstruction and motion estimation in dynamic scenes (Huguet and Devernay, 2007; Wedel and Cremers, 2011). Superpixel segmentation can also be formulated based on color, stereo depth, and stereo 3D flow simultaneously (Van den Bergh and van Gool, 2012). This approach operates at about 2 Hz using a GPU for optical flow computation and is not designed to find coherent segments of rigid-body motion. With a stereo camera, Zhang et al. (2011) propose dense 3D multibody SfM using an energy minimization framework. The approach relies on plane fitting to make the segmentation robust and is reported to require ca. 10 min per frame. Wang et al. (2012) transfer the approach of Cremers and Soatto (2005) to 3D time-of-flight images. They formulate a 3D optical flow constraint, and optimize for the 3D motion segmentation using level sets, but do not report on computational

load. Recently, a variational framework has been proposed that integrates rigid-body motion segmentation with dense 3D reconstruction (Roussos et al., 2012) from monocular image sequences. The batch method requires about 8 to 9 sec per frame on a GPU. We make efficient use of dense depth in RGB-D images for 3D motion segmentation—also integrating texture cues. The frame-rate of our approach is between 2 to 10 Hz on a CPU.

Within the robotics community, early work on dense motion segmentation has been pursued for the mapping of static and dynamic parts in environments using 2D laser scanners. Hähnel et al. (2003) propose an EM algorithm that filters dynamic parts of the environment in order to make the 2D occupancy mapping of the static environment parts robust. In simultaneous localization mapping and moving object tracking (SLAMMOT) (Wang et al., 2004), dynamic objects are segmented in laser scans through distance comparisons, and subsequently tracked while concurrently mapping the environment statics in a SLAM framework. Van de Ven et al. (2010) recently proposed a graphical model that integrates CRF-Matching (Ramos et al., 2007) and CRF-Clustering (Tipaldi and Ramos, 2009) within a single framework for 2D scan-matching, moving object detection, and motion estimation. They infer associations, motion segmentation, and 2D rigid-body motion through inference in the model using max-product LBP. We formulate dense 3D motion segmentation of RGB-D images using EM and perform fast approximate inference using graph cuts.

Interactive vision is a line of research in robotics that frequently uses motion cues to identify novel objects (Fitzpatrick, 2003; Kenney et al., 2009). Fitzpatrick (2003) proposed a background subtraction method in color images which segments the image into robot and object parts while the robot pokes objects. He finds the point of first contact in an image sequence and determines the moving parts beforehand (robot) and afterwards (object). Kenney et al. (2009) also perform background subtraction and find coherent object segments using graph cuts. For segmentation, these approaches assume a static camera pose, whereas our approach recovers camera and object motion concurrently. Furthermore, our segmentation method is suitable for mobile manipulation scenarios, where keeping the moved object within the field of view would involve camera motion.

4.5. Summary

We developed a general and efficient EM framework for dense sequential 3D rigid-body motion segmentation in RGB-D video. We employ EM to infer image labeling and motion estimates, and propose efficient approximations based on variational inference and graph cuts. Our approach recovers the number of motion segments and is suited for online operation in real-time. Our efficient probabilistic image representation in MRSMaps and rapid registration method facilitate fast performance. In experiments, we demonstrated high accuracy of

our method with regards to segmentation and motion estimates. Our approach also recovers the number of motion segments well.

The performance of our motion segmentation approach strongly depends on the underlying image representation. In order to improve the segmentation of fine-detailed structure and to increase the accuracy of motion estimation for small objects, we could integrate interest points into our dense segmentation approach. It could also be useful to adapt an oversegmentation of the image such as superpixels or supervoxels to our approach. While we handle degrading image overlap, segmentation evidence from multiple views could be beneficial to increase overlap.

Future research could investigate the application of our EM framework to different image representations and registration methods. For instance, motion could be segmented in an efficient GPU implementation, in which image pixels are directly used as image sites. Registration could be performed through RGB-D image warping as proposed by Steinbruecker et al. (2011). A further interesting application of our approach could be the dense 3D motion segmentation of video of a monocular camera. E.g., our method could be applied to the dense tracking and mapping approach recently proposed by Newcombe et al. (2011b).

As the rigid registration method used for the M-step has local convergence properties, also motion estimation converges locally. By using a global alignment method for registration as we will propose in Ch. 6 for object detection, global convergence could be achieved. Salas-Moreno et al. (2013) recently demonstrated that such a global registration method can be implemented for real-time operation on GPUs which could facilitate real-time motion segmentation with a global alignment method within our EM framework.

5. Deformable Registration

In deformable registration, we do not need to make the assumption that the whole scene or parts move rigidly between two images. Our approach is an efficient, multi-resolution extension of the coherent point drift (CPD) method (Myronenko and Song, 2010), making CPD well suitable for the deformable registration of RGB-D measurements. The CPD method determines a displacement field which assigns a motion to each point in a point cloud. It imposes local smoothness on the displacement field to make the estimation procedure well-posed and robust.

We transfer these concepts to our efficient multi-resolution surfel representation of RGB-D images. Each surfel moves within a displacement field which we regularize for local smoothness. We utilize the compact local multi-resolution structure of our maps to devise an efficient coarse-to-fine deformable registration algorithm. Resolution decreases with distance in our MRSMaps, which creates borders between adjacent resolutions. We extend the CPD approach to consider registration constraints found for adjacent surfels on coarser resolutions. The registration on finer resolutions is initialized from the result on the coarser one. In addition to depth, we also utilize color and contour cues. We improve robustness and efficiency of our algorithm by using a modified Gaussian kernel with compact support. Finally, we present means to estimate the local rigid transformation part of the displacement field at arbitrary points which will be useful for transfering object manipulation skills of robots (Ch. 7).

5.1. Background: Coherent Point Drift

The CPD method proposed by Myronenko and Song (2010) performs non-rigid deformable registration between two point clouds: We denote $X = (x_1 \dots x_N)^T$ as the scene and $Y = (y_1 \dots y_M)^T$ as the model point cloud with D-dimensional points $x_i, y_j \in \mathbb{R}^D$. We assume that the surface underlying the model point cloud has been deformed towards the scene surface according to the displacement

field $v : \mathbb{R}^D \to \mathbb{R}^D$ such that points y_j in the model cloud transform to a point $y_j + v(y_j)$ on the scene surface. The aim of the CPD method is to recover this displacement field. Fig. 5.1 illustrates the CPD algorithm.

5.1.1. Mixture Model for Observations

CPD explains the scene point cloud X as a set of samples from a mixture model on the deformed model cloud Y,

$$p(x_i \mid v, \sigma) = \sum_{j=1}^{M+1} p(c_{i,j})\ p(x_i \mid c_{i,j}, v, \sigma), \tag{5.1}$$

where $c_{i,j}$ is a shorthand for the 1-of-$(M+1)$ encoding binary variable $c_i \in \mathbb{B}^{M+1}$ with the j-th entry set to 1. Naturally, c_i indicates the association of x_i to exactly one of the mixture components. The model is a Gaussian mixture on the M deformed model points and an additional uniform mixture component,

$$p(x_i \mid v, \sigma) = \sum_{j=1}^{M} p(c_{i,j})\ \mathcal{N}(x_i; y_j + v(y_j), \sigma^2) + p(c_{i,M+1})\ p(x_i \mid c_{i,M+1}), \tag{5.2}$$

where σ is a standard deviation which is shared across all Gaussian mixture components. The uniform component generates each sample in X with equal probability $p(x_i \mid c_{i,M+1}) = \frac{1}{N}$. Its prior probability $w := p(c_{i,M+1})$ is a parameter that is chosen according to the noise inherent to the data. If we further assume equal prior likelihood for the association to each Gaussian mixture component, we obtain $p(c_{i,j}) = (1-w)\frac{1}{M}$ for all $j \in \{1, \ldots, M\}$.

By modeling the scene points as samples from a mixture model on the model cloud, the CPD method does not make a hard association decision between the point sets, but a scene point is theoretically associated to every model point. The probability $p(c_{i,j} \mid x_i, v, \sigma)$ quantifies the likelihood of the assignment of x_i to the model point y_j. The closer the displacement field v transforms y_j towards x_i, the more likely is the assignment of x_i to y_j.

5.1.2. Registration through Expectation-Maximization

The displacement field v is estimated through maximization of the logarithm of the joint data-likelihood

$$\ln p(X \mid v, \sigma) = \sum_{i=1}^{N} \ln \sum_{j=1}^{M+1} p(c_{i,j})\ p(x_i \mid c_{i,j}, v, \sigma). \tag{5.3}$$

While a direct optimization of this objective function is not feasible, it lends itself to the EM method (Sec. 4.1.1). The component associations $c = \{c_1, \ldots, c_N\}$ are

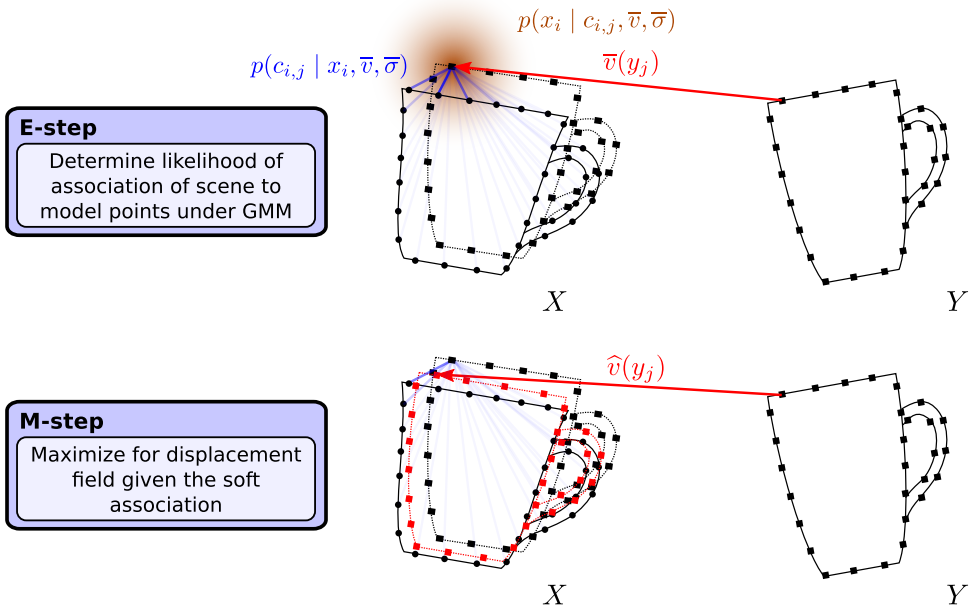

Figure 5.1.: The CPD method deformably registers scene X and model cloud Y in an EM framework. A GMM is imposed on the deformed model cloud. In the E-step (top), each scene point x_i is softly assigned to all model points y_j according to the assignment probability $p(c_{i,j} \mid x_i, \overline{v}, \overline{\sigma})$ of the scene point to the model point. The M-step (bottom) then determines a new displacement field $\widehat{v}$ from these soft associations.

treated as the latent variables to yield the EM objective

$$L(q, v, \sigma) := \sum_{i=1}^{N} \sum_{j=1}^{M+1} q(c_{i,j}) \ln \frac{p(c_{i,j})\ p(x_i \mid c_{i,j}, v, \sigma)}{q(c_{i,j})}, \tag{5.4}$$

by exploiting $q(c) = \prod_{i=1}^{N} \prod_{j=1}^{M+1} q(c_{i,j})$. In the M-step, the latest estimate $\overline{q}$ for the distribution over component associations is held fixed to optimize for the displacement field v and standard deviation σ

$$\{\widehat{v}, \widehat{\sigma}\} = \arg\max_{v,\sigma} L(\overline{q}, v, \sigma) \tag{5.5}$$

with

$$L(\overline{q}, v, \sigma) := \sum_{i=1}^{N} \sum_{j=1}^{M+1} \overline{q}(c_{i,j}) \ln p(c_{i,j})\ p(x_i \mid c_{i,j}, v, \sigma) =$$

$$const. - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{M} \overline{q}(c_{i,j}) \left(D \ln(2\pi\sigma^2) + \frac{1}{\sigma^2} \left\| x_i - (y_j + v(y_j)) \right\|_2^2 \right). \tag{5.6}$$

The E-step obtains a new optimum $\widehat{q}$ for the distribution q by the conditional likelihood of the cluster associations given the latest displacement field estimate $\overline{v}$ and standard deviation $\overline{\sigma}$

$$\widehat{q}(c_{i,j}) = p(c_{i,j} \mid x_i, \overline{v}, \overline{\sigma}) = \frac{p(c_{i,j})\ p(x_i \mid c_{i,j}, \overline{v}, \overline{\sigma})}{\sum_{j'=1}^{M+1} p(c_{i,j'})\ p(x_i \mid c_{i,j'}, \overline{v}, \overline{\sigma})}. \tag{5.7}$$

For the Gaussian mixture components this corresponds to

$$\widehat{q}(c_{i,j}) = \frac{\exp\left(-\frac{1}{2\overline{\sigma}^2}\left\|x_i - (y_j + v(y_j))\right\|_2^2\right)}{(2\pi\overline{\sigma}^2)^{D/2}\frac{w}{1-w}\frac{M}{N} + \sum_{j'=1}^{M}\exp\left(-\frac{1}{2\overline{\sigma}^2}\left\|x_i - (y_j + v(y_j))\right\|_2^2\right)}. \tag{5.8}$$

5.1.3. Regularized Deformation Field

It is a well known fact that estimating a function from a set of samples purely from the data-likelihood easily is an ill-posed problem (Tikhonov and Arsenin, 1977). In our specific setting of the estimation of a displacement field, continuity of the solution of Eq. (5.3) is violated since surfels would be assigned to their closest counterparts in the other surfel set. A small perturbation of the surfels' positions may lead to discontinuous changes in the solution. Hence, we need to constrain the displacement field by either restricting it to a specific parametric form or by using a regularizing prior that enforces smoothness.

Myronenko and Song (2010) augment the joint data-likelihood in Eq. (5.3) with a prior $p(v) = \exp\left(-\frac{\lambda}{2}\left\|v\right\|_{\mathcal{H}}^2\right)$:

$$\ln p(X, v \mid \sigma) = \ln p(X \mid \sigma, v) - \frac{\lambda}{2}\left\|v\right\|_{\mathcal{H}}^2. \tag{5.9}$$

which implements Tikhonov regularization (Tikhonov and Arsenin, 1977) by choosing the norm in a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$. It is straightforward to extend the EM approach of the previous Sec. 5.1.2 to the joint likelihood of data and displacement field,

$$L^{regularized}(q, v, \sigma) := \ln p(v) + \sum_{i=1}^{N}\sum_{j=1}^{M+1} q(c_{i,j}) \ln \frac{p(c_{i,j})\ p(x_i \mid c_{i,j}, v, \sigma)}{q(c_{i,j})}, \tag{5.10}$$

i.e. the lower bound constructed by EM is added a term $\ln p(v)$ that neither depends on the scene points nor the mixture component assignments. In the E-step, the prior is a constant term and has no influence on the q that best improves the lower bound (see Sec. 4.1.1), hence, Eq. (5.8) still applies. The M-step optimizes the regularized conditional expectation in Eq. (5.10) for the displacement field v and the standard deviation σ with fixed q.

It is possible to show that applying a Gaussian reproducing kernel $g(y, y') := \exp\left(-\frac{\left\|y-y'\right\|_2^2}{2\beta^2}\right)$ is equivalent to the regularization proposed in motion coherence

theory (Myronenko, 2010). The Gaussian regularizer penalizes high frequencies in the displacement field which can be seen from the equivalent formulation of the norm in frequency domain

$$\|v\|_{\mathcal{H}}^2 = \int \frac{|V(\omega)|^2}{g(\omega)} d\omega, \tag{5.11}$$

where $V(\omega)$ is the Fourier transform of v, $g(\omega)$ is the Fourier transform of the Gaussian function, and ω is a frequency.

A norm $\|Pv\|^2$ on the outcome of a linear differential operator P applied to v also induces a RKHS (Smola et al., 1998). The reproducing kernel $k(y, y')$ is equivalent to the Green's function of the differential operator P^*P

$$P^*Pk(y, y') = \delta(y - y'), \tag{5.12}$$

where P^* is the adjoint operator to P and δ is the Dirac function. A Green's function can be interpreted as defining a right-inverse integral operator to a differential operator which is utilizable to solve the partial differential equation $Lv(y) = f(y)$ for v. A solution is $v(y) = \int k(y, y')f(y')dy'$ since

$$L \int k(y, y')f(y')dy' = \int Lk(y, y')f(y')dy' = \int \delta(y - y')f(y')dy' = f(y), \tag{5.13}$$

for which we exploit the linearity of L to move it inside the integral. Conversely, we can find a linear differential operator P for any RKHS (Smola et al., 1998; Chen and Haykin, 2002). This alternative view in terms of differential operators will be useful to derive a solution for v in the M-step (Eq. (5.9)).

For instance, the RKHS induced by a Gaussian kernel can be defined in terms of the differential operator P with

$$\|Pv\|^2 = \sum_{k=1}^{K} a_k \int \sum_{j_1 + \cdots + j_D = k} \left(\frac{\partial^k v(y)}{\partial y_1^{j_1} \dots \partial y_D^{j_D}} \right) dy \tag{5.14}$$

and $a_k := \frac{\sigma^{2k}}{k! 2^k}$ (Chen and Haykin, 2002; Rasmussen and Williams, 2005).

5.1.4. Regularized Maximization Step

In the M-step, we optimize (5.10) for the displacement field v and the standard deviation σ. Since a joint closed-form solution is not available, Myronenko and Song (2010) optimize for v and σ alternately.

Standard Deviation: Setting the derivative of Eq. (5.10) for the standard deviation σ to zero yields

$$\hat{\sigma}^2 = \frac{1}{N_P D} \sum_{i=1}^{N} \sum_{j=1}^{M} \overline{q}(c_{i,j}) \ \|x_i - (y_j + v(y_j))\|_2^2, \tag{5.15}$$

where we define $N_P := \sum_{i=1}^{N} \sum_{j=1}^{M} \overline{q}(c_{i,j})$.

Deformation Field: Analogous to the derivation in (Chen and Haykin, 2002), we obtain the Euler-Lagrange equation for the functional in Eq. (5.10),

$$P^*P\ \widehat{v}(y) = \frac{1}{\sigma^2\lambda}\sum_{i=1}^{N}\sum_{j=1}^{M}\overline{q}(c_{i,j})\ (x_i - (y_j + \widehat{v}(y_j)))\ \delta(y - y_j). \tag{5.16}$$

With the choice of a Gaussian kernel, this partial differential equation can be solved using the Green's function $k(y,y') \equiv g(y,y')$ of the operator P^*P

$$\widehat{v}(y) = \int k(y,y')\frac{1}{\sigma^2\lambda}\sum_{i=1}^{N}\sum_{j=1}^{M}\overline{q}(c_{i,j})\ (x_i - (y_j + \widehat{v}(y_j)))\ \delta(y' - y_j)dy' \tag{5.17}$$

$$= \frac{1}{\sigma^2\lambda}\sum_{i=1}^{N}\sum_{j=1}^{M}\overline{q}(c_{i,j})\ (x_i - (y_j + \widehat{v}(y_j)))\ k(y,y_j) \tag{5.18}$$

$$= \sum_{j=1}^{M} w_j k(y,y_j), \tag{5.19}$$

with weights $w_j := \frac{1}{\sigma^2\lambda}\sum_{i=1}^{N}\overline{q}(c_{i,j})\ (x_i - (y_j + \widehat{v}(y_j))) \in \mathbb{R}^D$. Note the resemblance to the representer's theorem (Schölkopf et al., 2001): the solution is a linear combination of data-dependent terms w_j weighted with the kernel evaluated at the data points y_j.

To obtain a solution we need to evaluate the solution $\widehat{v}(y)$ at the model points y_j and solve for the weights w_j. Let $W := (w_1, \ldots, w_M)^T \in \mathbb{R}^{M\times D}$ to write $v(y) = GW$ using the Gram matrix $G \in \mathbb{R}^{M\times M}$ with $G_{ij} := k(y_i, y_j)$. The weights for the solution $\widehat{v}(y)$ are

$$W = (\lambda\sigma^2 I + dP1\,G)^{-1}\,(PX - dP1\,Y)\,, \tag{5.20}$$

where $P \in \mathbb{R}^{M\times N}$ with $P_{ji} := \overline{q}(c_{i,j})$ and $dP1 := \mathrm{diag}(P\,1_{N\times 1})$ (Myronenko and Song, 2010).

The solution for the weights W in Eq. (5.20) requires the inversion of a potentially large $M \times M$-matrix whose size depends on the size of the model point cloud. To reduce complexity, Myronenko and Song (2010) propose to utilize a low-rank approximation of G, $\widetilde{G} := Q\Lambda Q^T$ with the matrix Q of eigenvectors and the diagonal matrix Λ containing the K largest eigenvalues of G. Using the Woodbury identity, Eq. (5.20) is reformulated to arrive at

$$W \approx \frac{1}{\lambda\sigma^2}\left(I - dP1\,Q\left(\lambda\sigma^2\Lambda^{-1} + Q^T dP1\,Q\right)^{-1}Q^T\right)(PX - dP1\,Y). \tag{5.21}$$

The outer inversion acts on a $K \times K$-matrix, such that we can drastically improve run-time over the $M \times M$-matrix inversion in Eq. (5.20) by choosing $K \ll M$. The low-rank approximation constrains the solution for the displacement field in a low-dimensional embedding, which further regularizes the displacement field.

Myronenko and Song (2010) further propose to use the fast Gauss transform (FGT) (Greengard and Strain, 1991) to efficiently evaluate matrix product expressions PZ that involve the matrix P of mixture component association probabilities between X and Y. The FGT utilizes a truncated series expansion of the Gaussian to evaluate a weighted sum of Gaussians centered at a set of source positions at a set of target positions. If the number of sources and targets is N and M, respectively, the approach reduces the run-time complexity from $\mathcal{O}(NM)$ to only $\mathcal{O}(N+M)$.

5.2. Efficient Coarse-To-Fine Deformable Registration of Multi-Resolution Surfel Maps

The run-time complexity of the CPD algorithm depends at least quadratically on the size of the model point set through the construction of the Gram matrix. If we do not apply the low-rank approximation it is even cubic in the size of the model cloud due to the inversion of the Gram matrix. By processing the resolutions from coarse to fine we can keep the size of the point clouds as small as possible. The displacement field of coarse resolutions can be used to initialize the displacement on the next finer resolution such that the number of iterations required to converge is dramatically decreased.

We represent the RGB-D measurements by a scene and model MRSMap. The means of the surfels within each resolution $\rho(d)$ at depth d of the maps define scene and model point clouds $X_d := \left(x_{d,1}, \ldots, x_{d,N_d}\right)$ and $Y_d := \left(y_{d,1}, \ldots, y_{d,M_d}\right)$. Note that we assume that the view-point of the camera onto both maps is known and that we can extract the surfels that the camera views onto.

We iterate from coarse to fine resolutions, starting at the coarsest resolution $\rho(0)$ at depth 0 in the map. Let d be the current depth processed. Our aim is to find the displacement field v_d from scene to model point clouds X_d, Y_d and the standard deviation σ_d.

5.2.1. Per-Resolution Initialization

We initialize the registration on each depth with the displacement field v_{d-1} of the previous coarser resolution. Each mean $y_{d,i}$ on the current depth is mapped to its displacement

$$v_{d-1}(y_{d,i}) = \sum_{j=1}^{M_{d-1}} w_{d-1,j}\ k(y_{d,i}, y_{d-1,j}) \tag{5.22}$$

according to the coarser resolution displacement field which we abbreviate as

$$v_{d-1}(Y_d) = G(Y_d, Y_{d-1})\ W_{d-1}, \tag{5.23}$$

where $G(Y_d, Y_{d-1}) \in \mathbb{R}^{M_d \times M_{d-1}}$ is a Gram matrix with $g_{ij} := k(y_{d,i}, y_{d-1,j})$. Subsequently, we utilize $v(Y_d) = G_d W_d$ to solve for the initial weight matrix

$$W_d \leftarrow G_d^{-1}\ G(Y_d, Y_{d-1})\ W_{d-1}. \tag{5.24}$$

on the current depth.

If we use a low-rank approximation, we have two alternatives to initialize W_d. We may compensate for the effect of the low-rank approximation on the found weights through

$$W_d \leftarrow \widehat{G}_d^{-1}\ G(Y_d, Y_{d-1})\ G_{d-1}^{-1}\ \widehat{G}_{d-1} W_{d-1}. \tag{5.25}$$

This approach requires the inversion of the low-rank approximation $\widehat{G}_d$ and the full-rank Gram matrix G_{d-1}^{-1}. While the former is in $\mathcal{O}(K^3)$ due to $\widehat{G}_d^{-1} = Q\Lambda^{-1}Q^T$, the latter is in $\mathcal{O}(M^3)$. Notably, both inversion could be precomputed once, for instance, if the model cloud is an object map, or for sequential registration of scene maps towards a persisting model map.

Alternatively, we can exploit that in the MRSMap the surfels at the current resolution are descendants of surfels on depth $d-1$ with corresponding displacements $\widehat{G}_{d-1} W_{d-1}$. Let $\phi : \mathbb{N} \to \mathbb{N}$ be an index function that maps each $y_{d,i}$ in Y_d to its parent surfel $y_{d-1,j}$ in Y_{d-1}. We define the mapping $\Phi : y_{d,i} \mapsto y_{d,\phi(i)}$ and establish $v_d(Y_d) = v_{d-1}(\Phi(Y_d))$ through

$$W_d \leftarrow \widehat{G}_d^{-1}\ v_{d-1}(\Phi(Y_d)). \tag{5.26}$$

The standard deviation $\sigma_d \leftarrow \sigma_{d-1}$ is simply initialized from the result σ_{d-1} of the previous iteration.

5.2.2. Resolution-Dependent Kernel with Compact Support

Gaussian kernels produce a dense Gram matrix with potentially very small entries (see Fig. 5.2). The smaller the scale β the larger the condition number of the Gram matrix and, hence, the less numerically stable is the inversion of the Gram matrix (Fornberg and Zuev, 2007). Furthermore, sparse matrices can be inverted much more efficiently than dense matrices using sparse matrix factorizations such as the LU- or Cholesky-decompositions. We therefore use a modified Gaussian kernel with compact support (Genton, 2002) instead, i.e.,

$$k(y, y') = \varphi_{l,k}(y, y')\ g(y, y'), \tag{5.27}$$

where $\varphi_{l,k} \in \mathcal{C}^{2k}$ is a Wendland kernel (Wendland, 1995) with $l = \lfloor D/2 \rfloor + k + 1 \in \mathbb{N}$.

The family of Wendland kernels is positive-definite and has compact support, hence also our modified Gaussian kernel is a valid kernel with compact support. Since our points are 7-dimensional and we require a kernel that is once

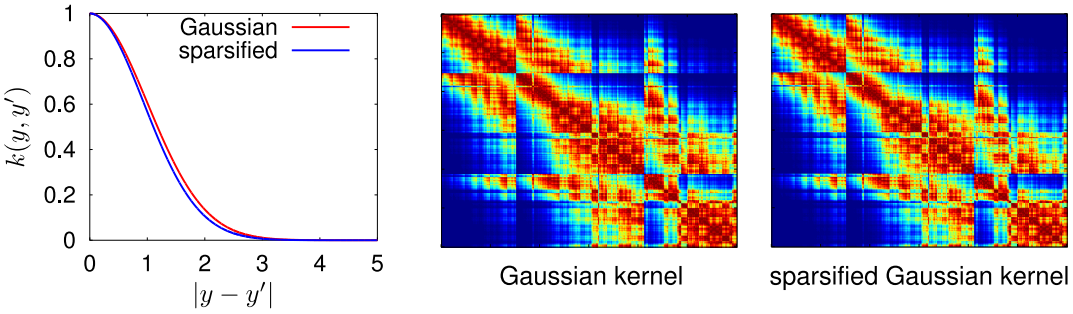

Figure 5.2.: Gaussian kernels produce a dense Gram matrix whose condition number decreases quickly with size. We use a modified Gaussian kernel with compact support to sparsify the Gram matrix. Left: kernels for $\beta = 1$. Center: Example gram matrix at resolution $\rho(d)^{-1} = 0.025\,m$ for Gaussian kernel ($\beta = 10$). Right: Example gram matrix at resolution $\rho(d)^{-1} = 0.025\,m$ for sparsified Gaussian kernel ($\beta = 10$).

differentiable ($\varphi_{l,k} \in \mathcal{C}^1$), we utilize

$$\varphi_{5,1}(y, y') = \max\left\{0, \left(1 - \frac{\|y - y'\|_2}{16\beta}\right)^6 \left(\frac{6\,\|y - y'\|_2}{16\beta} + 1\right)\right\}. \tag{5.28}$$

We adapt the scale $\beta_d = \beta_0\,\rho(d)^{-1}$ of the kernel $k_d(y, y')$ to the current resolution $\rho(d)$. This way spatial smoothing is performed from low to high frequencies which is required as high frequencies in the displacement field are only observable on fine resolutions due to the sampling theorem. The required amount of smoothing, i.e., the magnitude of β_0 depends on the strength of deformations in the observations.

5.2.3. Handling of Resolution-Borders

Since we use a distance-dependent resolution limit in MRSMaps, surfels have redundant counterparts in ancestor nodes on coarser resolutions, but they may not be represented at finer resolutions. This leads to surfels whose local context is in parts only present at coarser resolutions. We denote the set of surfels with this property as resolution border surfels.

We still constrain the deformation of resolution border surfels to the displacement field in the complete local context of the surfels (see Fig. 5.3). We include the means X_{d-1} of the scene surfels from the previous coarser resolution into the scene point set. Secondly, we add a further prior on the displacement field v_d to Eq. (5.9),

$$\ln p(X_d, v_d \mid \sigma_d, v_{d-1}) = \ln p(X_d \mid \sigma_d, v_d) + \ln p(v_d \mid v_{d-1}) - \frac{\lambda}{2}\,\|v_d\|^2_{\mathcal{H}}. \tag{5.29}$$

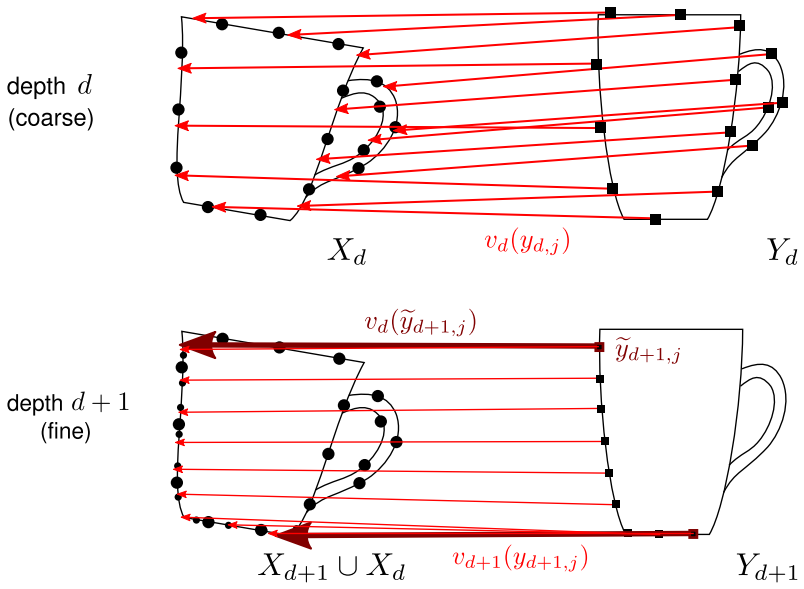

Figure 5.3.: Since we adapt the maximum resolution in MRSMaps with distance from the sensor, the represented parts of the surface may reduce with resolution. In our coarse-to-fine scheme, we condition the displacement field at resolution borders $\widetilde{y}_{d+1,j}$ on the deformation field of the coarser resolution. We also include scene points from the coarser resolution to include missing support in the fine resolution scene cloud.

to favor compatibility with the displacement field v_{d-1} of the coarser resolution at the resolution border surfels. While the E-step is unchanged, we need to consider this prior in the M-step.

Let $\tilde{Y}_d \subseteq Y_d$ be the means of the resolution border surfels at the current resolution. We model the prior

$$\ln p(v_d \mid v_{d-1}) := -\frac{1}{2} \sum_{j=1}^{M_d} \gamma(y_{d,j}) \left\| v_d(y_{d,j}) - v_{d-1}(y_{d,j}) \right\|_2^2, \tag{5.30}$$

with

$$\gamma(y_{d,j}) := \begin{cases} \sigma_\gamma^{-2} & \text{if } y_{d,j} \in \tilde{Y}_d \\ 0 & \text{otherwise.} \end{cases} \tag{5.31}$$

Again, we adapt $\sigma_\gamma := \sigma_{\gamma,0}\, \rho(d)^{-1}$ to the current resolution.

With this additional prior term, we obtain the Euler-Lagrange equation

$$P^* P\ \widehat{v}_d(y) = \frac{1}{\sigma_d^2 \lambda} \sum_{j=1}^{M_d} w'_{d,j}\ \delta(y - y_j), \tag{5.32}$$

where we now define

$$w'_{d,j} := \frac{1}{\sigma_d^2 \lambda} \left(\sum_{i=1}^{N_d} \overline{q}(c_{i,j}) \; (x_{d,i} - (y_{d,j} + \widehat{v}_d(y_{d,j}))) \right) + \frac{1}{\lambda} \gamma(y_{d,j}) \left(v_{d-1}(y_{d,j}) - \widehat{v}_d(y_{d,j}) \right). \quad (5.33)$$

Using the Green's function $k(y, y')$ we solve for $\widehat{v}_d(y)$:

$$\widehat{v}_d(y) = \sum_{j=1}^{M_d} w'_{d,j} \; k(y, y_{d,j}). \quad (5.34)$$

The weights are determined by evaluating the displacement field at Y_d,

$$\widehat{v}_d(y_{d,j'}) = \sum_{j=1}^{M_d} w'_{d,j} \; k(y_{d,j'}, y_{d,j}) \quad (5.35)$$

such that we substitute $\widehat{v}_d(y_{d,j'})$ in Eq. (5.33) to yield

$$w'_{d,j} = \frac{1}{\sigma_d^2 \lambda} \left(\sum_{i=1}^{N_d} P_{ji} x_{d,i} \right) - \frac{1}{\sigma_d^2 \lambda} \left(\sum_{i=1}^{N_d} P_{ji} \right) y_{d,j} + \frac{1}{\sigma_d^2 \lambda} \left[\left(\sum_{i=1}^{N_d} P_{ji} \right) + \gamma(y_{d,j}) \right] \widehat{v}_d(y_{d,j}) + \frac{1}{\lambda} \gamma(y_{d,j}) \, v_{d-1}(y_{d,j}). \quad (5.36)$$

By rearranging terms and taking the transpose we have

$$\sigma_d^2 \lambda \, w'^T_{d,j} + \left[\left(\sum_{i=1}^{N_d} P_{ji} \right) + \sigma^2 \gamma \right] \; G(y_{d,j}, Y_d) \; W'_d = \left[\left(\sum_{i=1}^{N_d} P_{ji} x_{d,i} \right) - \left(\sum_{i=1}^{N_d} P_{ji} \right) \; y_{d,j} + \sigma_d^2 \gamma(y_{d,j}) \, v_{d-1}(y_{d,j}) \right]^T, \quad (5.37)$$

such that we obtain the system of linear equations

$$\left(\sigma_d^2 \lambda I + \left(dP1 + \sigma_d^2 \, d\Gamma \right) G_d \right) W'_d = PX_d - dP1 \, Y_d + \sigma_d^2 \, d\Gamma \, v_{d-1}(Y_d), \quad (5.38)$$

where we used the shorthand $d\Gamma := \mathrm{diag}(\gamma(Y_d))$. We finally arrive at the update formula for the weights

$$W'_d = \left(\sigma_d^2 \lambda I + \left(dP1 + \sigma_d^2 \, d\Gamma \right) G_d \right)^{-1} \left(PX_d - dP1 \, Y_d + \sigma_d^2 \, d\Gamma \, v_{d-1}(Y_d) \right), \quad (5.39)$$

using the full-rank Gram matrix and

$$W'_d \approx \frac{1}{\lambda \sigma_d^2} \left(I - \left(dP1 + \sigma_d^2 \, d\Gamma \right) Q_d \left(\lambda \sigma_d^2 \Lambda_d^{-1} + Q_d^T \left(dP1 + \sigma_d^2 \, d\Gamma \right) Q_d \right)^{-1} Q_d^T \right) \left(PX_d - dP1 \, Y_d + \sigma_d^2 \, d\Gamma \, v_{d-1}(Y_d) \right). \quad (5.40)$$

with the low-rank approximation $\widehat{G}_d = Q_d \Lambda_d Q_d^T$.

5.2.4. Convergence Criteria

We iterate the EM steps on each resolution until convergence. One condition examines the relative change

$$\Delta L_t := \left| \frac{L_t - L_{t-1}}{L_{t-1}} \right|, L_t := \frac{1}{2}\lambda \left\| v_{d,t} \right\|_{\mathcal{H}}^2 \tag{5.41}$$

in the norm of the displacement field

$$\left\| v_{d,t} \right\|_{\mathcal{H}}^2 = \mathrm{tr}(W_{d,t}^T G_{d,t} W_{d,t}). \tag{5.42}$$

If this rate decreases below a threshold, the estimate of the displacement field is assumed to have converged.

A second criterion is required due to the FGT approximation for the evaluation of matrix product expressions that involve $\overline{q}$. Since the Gaussian is truncated there may exist a $y_{d,j}$ for which $\overline{q}(c_{i,j}) = 0$ for all $x_{d,i}$ at sufficiently small σ_d. If this is the case, we assume that we reached the smallest achievable σ_d on the current resolution and resume the registration on the next finer resolution. At the beginning of the EM steps on each new resolution, we search for an adequate σ_d by scaling it up by a factor of 2 until all $y_{d,j}$ have non-zero weighting for some $x_{d,i}$ through the FGT.

5.2.5. Color and Contour Cues

The CPD method is not limited to registration in the spatial domain. We use the full six-dimensional spatial and color mean of the surfels. In addition, we add contours determined as surfels at foreground borders as a seventh point dimension. We set the contour value of a point to β_d if it is on a foreground border, or 0 otherwise. This places points closer in feature space that are either on or off contours.

5.3. Local Deformations

The continuous displacement field allows us to estimate the local infinitesimal deformation at any point in terms of translation and rotation between both surfaces (see Fig. 5.4). These local deformation quantities can be estimated in each direction between scene and model surface. Since the displacement field is defined to act on points on the model surface, we first investigate the direction from model to scene.

5.3.1. Local Deformations from Model to Scene

It is well known in continuum mechanics (e.g., Batra, 2006) how infinitesimal local deformations can be estimated from a continuous deformation function

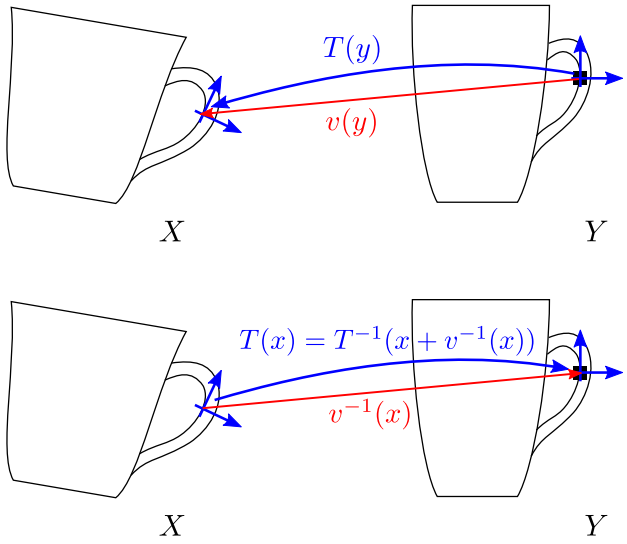

Figure 5.4.: We estimate local transformations T, i.e., rotation and translation, from model to scene (top) and scene to model (bottom).

$\phi : \mathbb{R}^3 \mapsto \mathbb{R}^3$ that maps the position of infinitesimal particles in an elastic body to their deformed location. Our displacement field v defines such a deformation function in a straightforward way,

$$\phi(y) := y + v(y). \tag{5.43}$$

The infinitesimal deformation at a point y is then specified by the Jacobian of the deformation function at y,

$$\nabla_y \phi(y) = I + \nabla_y v(y). \tag{5.44}$$

As long as we use differentiable kernels in our estimation algorithm, we may write

$$\nabla_y \phi(y) = I + \sum_{i=1}^{M} w_i \nabla_y k(y_i, y). \tag{5.45}$$

Rotation $R(y)$ and strain $S(y)$ are obtained through polar decomposition of the Jacobian $\nabla_y \phi(y) = RU$, i.e.,

$$R(y) = UV^T, \tag{5.46}$$

$$S(y) = V \Sigma V^T, \tag{5.47}$$

where $\nabla_y \phi(y) = U \Sigma V^T$ is the singular value decomposition (SVD) of the Jacobian. The translation $t(y) = v(y)$ is set to the displacement at y.

To query the local deformation of a point y from a deformable registration result for MRSMaps, we first find the finest resolution $\rho(d)$ in which the point

y is represented in the model map. Translation, rotation, and strain are then determined via the displacement field v_d.

In the case of the use of a low-rank approximation, the weights W of the displacement field v are computed with respect to a low-dimensional embedding of the kernel $k(y, y')$. Hence, Eq. (5.45) is not directly applicable. Analogously to the problem of estimating the displacements at fine resolutions from the displacement field of coarser resolution, we propose two alternatives. It is possible to establish a weight matrix W' that establishes the same displacement field v like W with the original Gram matrix G through $W' := G^{-1}\,\widehat{G}\,W$. We can then calculate the Jacobian as

$$\nabla_y \phi(y) = I + \sum_{i=1}^{M} w_i' \nabla_y k(y_i, y). \tag{5.48}$$

Alternatively, the inversion of G can be avoided by estimating translation and rotation from the local displacements around y (Arun et al., 1987). We determine the local means of model points y_i and deformed model points $y_{v,i} := y_i + v(y_i)$

$$\mu_y := \frac{\sum_{i=1}^{M} g(y, y_i, r)\, y_i}{\sum_{i=1}^{M} g(y, y_i, r)}, \tag{5.49}$$

$$\mu_{y_v} := \frac{\sum_{i=1}^{M} g(y, y_i, r)\, y_{v,i}}{\sum_{i=1}^{M} g(y, y_i, r)}, \tag{5.50}$$

and the centered model points and their deformed counterparts

$$\overline{y}_i := y_i - \mu_y, \tag{5.51}$$

$$\overline{y}_{v,i} := y_{v,i} - \mu_{v_y}. \tag{5.52}$$

Singular value decomposition of the scatter matrix

$$D := \sum_{i=1}^{M} g(y, y_i, r)\, \overline{y}_{v,i}\, \overline{y}_i^T = U \Sigma V^T \tag{5.53}$$

yields rotation $R(y) \approx U\,\tilde{I}\,V^T$, where

$$\tilde{I} := \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(UV^T) \end{pmatrix} \tag{5.54}$$

establishes $det(R(y)) = 1$. The translation is recovered from $t(y) \approx \mu_{v_y} - R(y)\mu_y$.

5.3.2. Local Deformations from Scene to Model

A closed-form solution to the local deformations from scene to model would require the inverse $v^{-1}(x)$ of the displacement field v for a scene point x. Since

such an inverse is not available, we approximate the inverse displacement

$$v^{-1}(x) = -\frac{1}{\sum_{i=1}^{M} g(x, y_i + v(y_i), r)} \sum_{i=1}^{M} g(x, y_i + v(y_i), r)\, v(y_i). \tag{5.55}$$

with the displacements of model points y_i that deform close to x. We can then use the closed-form approach in Eqs. (5.45) and (5.46) to determine the local rotation $R(x) = R(x + v^{-1}(x))^T$. The translation is $t(x) = v^{-1}(x)$.

For estimating rotation and translation while using low-rank approximations, we again have both options to estimate the local deformation as in Sec. 5.3.1. While the first approach is unchanged, in the second one we modify Eqs. (5.49) and (5.53) to consider the spatial distance between x and $y_i + v(y_i)$ for the weighting, i.e.,

$$\mu_y := \frac{\sum_{i=1}^{M} g(x, y_i + v(y_i), r)\, y_i}{\sum_{i=1}^{M} g(x, y_i + v(y_i), r)}, \tag{5.56}$$

$$\mu_{y_v} := \frac{\sum_{i=1}^{M} g(x, y_i + v(y_i), r)\, y_{v,i}}{\sum_{i=1}^{M} g(x, y_i + v(y_i), r)}, \tag{5.57}$$

$$D := \sum_{i=1}^{M} g(x, y_i + v(y_i), r)\, \overline{y}_{v,i}\, \overline{y}_i^T. \tag{5.58}$$

Rotation $R(x) = R(y)^T$ and translation $t(x) = -R(y)^T t(y)$ are obtained from the inverse of the transformation result $T(y)$.

5.4. Experiments

5.4.1. Quantitative Evaluation

We evaluate accuracy and run-time of our registration approach on synthetically deformed RGB-D images. For our experiments we used an Intel Core i7-4770K CPU (max. 3.50 GHz) and 32 GB of RAM. We chose two sequences of the RGB-D benchmark dataset (Sturm et al., 2012) for our experiments. In the freiburg2_desk sequence the camera observes a table-top scene. The planar surfaces create local aperture problems that need to be adressed by smoothness regularization. The freiburg3_teddy sequence contains views on a teddy bear with salient yellow and brown coloring. We process 500 randomly deformed frames per sequence to assess the accuracy of our method in recovering deformation as well as the run-time required to align the images.

We synthetically generate deformations in order to have ground truth available for assessing registration accuracy (see Figs. 5.5 and 5.6). Each frame is randomly deformed by adding Gaussian noise to the 3D Euclidean dimensions.

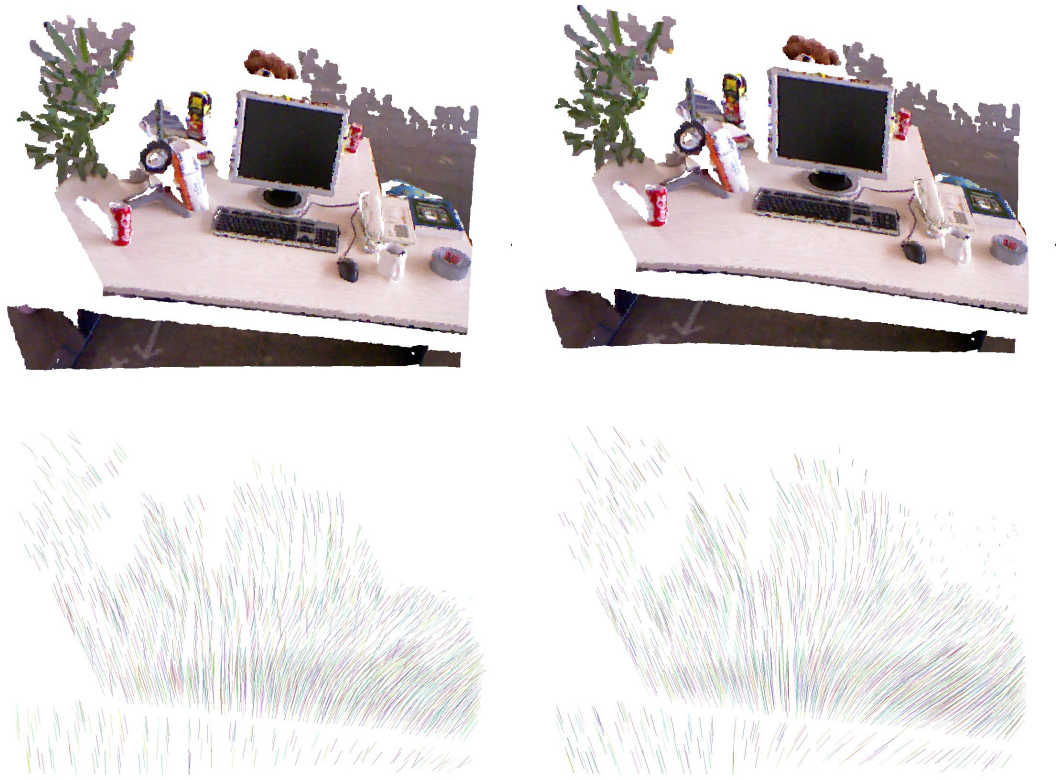

Figure 5.5.: Top row: Example RGB-D image (left) of the freiburg2_desk sequence with synthetic deformations (right). Bottom row: Estimated (left) and ground-truth (right) displacement field.

We sample the Gaussian noise in image coordinates and choose a standard deviation uniformly between 100 and 200 pixels in the x- and y- direction of the image separately. Each of ten Gaussians applies up to 0.1 m distortion. In total we normalize the applied deformation to a maximum of 0.1 m in each direction.

We assess the performance of several variants of our approach. Full-rank methods are marked by "F", whereas we denote low-rank approximations by "L". The variants F−− and F−+ do not use color for registration, while the second sign indicates the use of the contour cue. The methods tagged with "*" do not include surfels from coarser resolutions from the scene cloud and do not constrain the displacement field on the resulting field of the coarser resolution. However, we initialize weights from the coarser resolutions and iterate from coarse to fine. For all full-rank approaches we set $\beta_0 = 160$. The low-rank approximations have been run with $\beta_0 = 20$.

Tables 5.1 and 5.2 summarize the average run-time in milliseconds spent per frame. Using additional cues such as color and contours increases the run-time slightly. The variants utilizing low-rank approximations are significantly faster

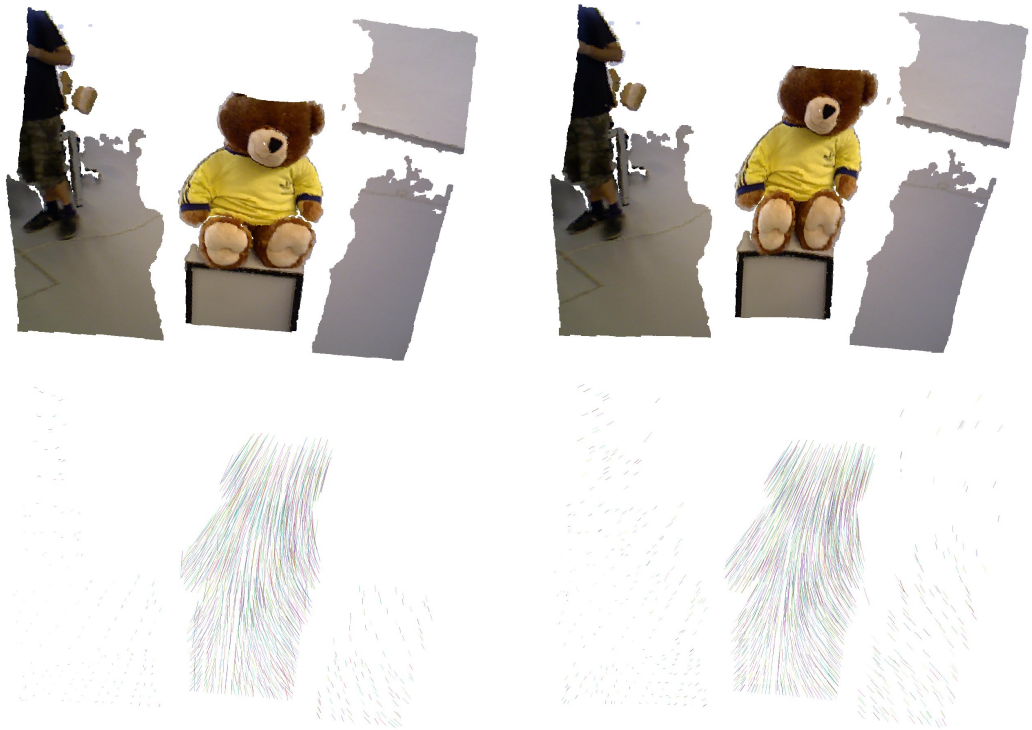

Figure 5.6.: Top row: Example RGB-D image (left) of the freiburg3_teddy sequence with synthetic deformations (right). Bottom row: Estimated (left) and ground-truth (right) displacement field.

in the registration step, while the preparation step is more expensive. This preparation step would only be needed to be executed once for a fixed object model. In this case, our low-rank coarse-to-fine registration method achieves a frame rate between 1 to 5 Hz. The run-time of plain concurrent processing of all the surfels in the MRSMap requires run-time of 10 to 30 seconds per image using low-rank approximations. Similarly plain registration of RGB-D images takes several seconds at a downsampling factor of 8 (resolution 80×60). Larger image resolutions could not be processed due to memory limitations.

Figs. 5.7 and 5.8 demonstrate the accuracy of our approach. Using color and contour cues gives best performance on the finest resolution (0.025 m). Not using color, contours, or coarse-to-fine registration degrades performance. We notice that using low-rank approximations only slighly decreases accuracy.

5.4.2. Deformable Registration and Local Transformation Examples

In Fig. 5.9 we show typical results of our low-rank deformable registration method on RGB-D image segments of objects. Examples for estimated local

Table 5.1.: Comparison of average run-time in milliseconds per image using full-rank Gram matrices.

sequence	F	F−+	F+−	F−−	F*
fr2 desk, prepare	344	330	340	**325**	344
fr2 desk, register	7802	6621	**1826**	1848	5278
fr2 desk, total	8216	7020	**2235**	2243	5693
fr3 teddy, prepare	141	135	139	**133**	141
fr3 teddy, register	3697	3340	**1367**	1494	3435
fr3 teddy, total	3921	3559	**1589**	1711	3659

Table 5.2.: Comparison of average run-time in milliseconds per image using the low-rank approximation to the Gram matrix.

sequence	L	L−+	L+−	L−−	L*
fr2 desk, prepare	437	423	433	**417**	438
fr2 desk, register	643	464	553	**348**	425
fr2 desk, total	1149	957	1056	**835**	933
fr3 teddy, prepare	222	216	220	**214**	221
fr3 teddy, register	467	335	390	**268**	290
fr3 teddy, total	772	634	693	**565**	594

transformations can be found in Fig. 5.10. The local transformations are estimated from scene to model using our sample-based approach (Sec. 5.3.2). The local coordinate frames are well positioned at their counterparts in both image segments. Also the orientation reflects the local bending of the surface, if it is sufficiently densely sampled. In the example of the humanoid robot, the orientation change seems to be underestimated: At the maximum resolution available, the sampling of the surface is not dense enough to recover the orientation change from the sample displacements. The sampling rate could be improved with an RGB-D sensor with higher depth resolution and less noisy measurements. Also in future work, one could investigate the deformation of the surfel covariances and normals for the registration and the estimation of local transformations.

5.5. Related Work

Many approaches to deformable registration represent scene and model surface by meshes or point clouds and estimate the local deformation of vertices or

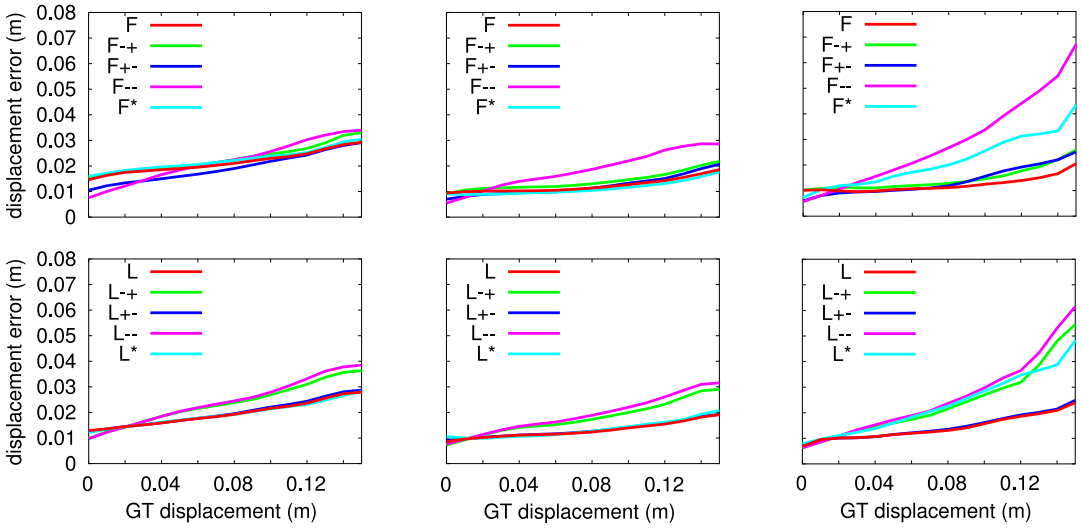

Figure 5.7.: Median accuracy in m for deformable registration of synthetically deformed RGB-D images on the freiburg2_desk dataset. Left: 0.1 m, middle: 0.05 m, right: 0.025 m resolutions.

points. These methods can be characterized by the way they determine correspondences between the surfaces, the space of local transformations they model, and the type of regularization they apply to enforce local smoothness or rigidity. Allen et al. (2003) learn a shape-space of human bodies through deformable registration. They adapt the ICP algorithm to perform deformable registration between measured meshes of persons. Instead of estimating a single global rigid transformation, they determine a local rigid transformation at each vertex through energy minimization. The data terms of the energy measure the squared distance of vertices towards the closest counterparts in the other mesh after the transformation has been applied. To enforce smoothness of local transformations of neighboring vertices in the mesh, the difference between the local transforms is minimized concurrently. Amberg et al. (2007) take a similar approach to align arbitrary meshes. They, however, allow for local affine transformations at the vertices. Moreover, they propose an exact solution to the quadratic optimization problem that improves convergence. In addition to local transforms at each vertex, Li et al. (2008) include a global rigid transformation that applies on the complete mesh. Their energy formulation facilitates rigidness of the local affine transformations. The approach in Willimon et al. (2013) enforces alignment of boundaries to register RGB-D images of clothing. Their data term takes the difference in depth in the image into account.

The above methods establish only a single correspondence for each point or vertex. It has been observed that the basin of convergence and accuracy can be improved by allowing each surface element to be softly assigned with multiple elements of the other surface. Anguelov et al. (2004) model the correspondence

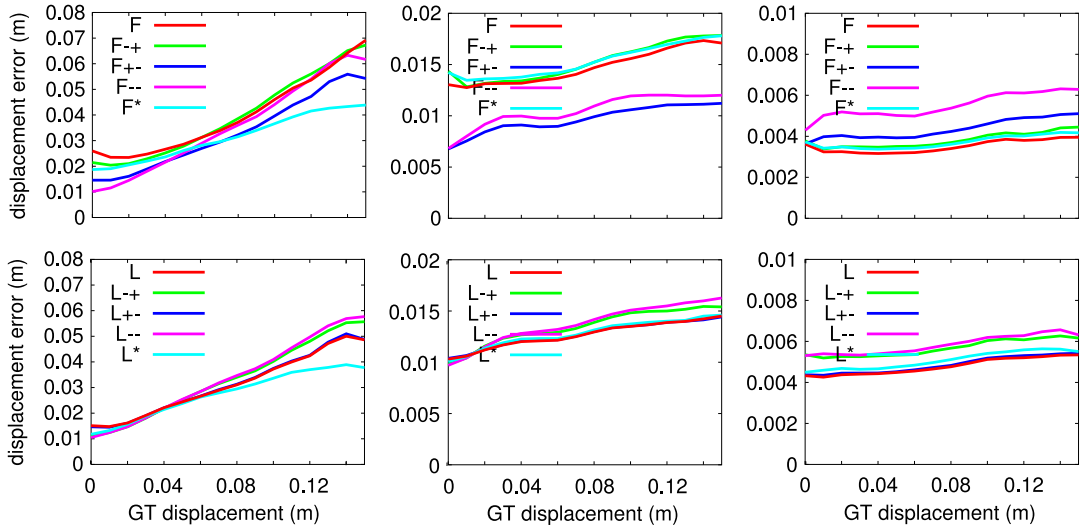

Figure 5.8.: Median accuracy in m for deformable registration of synthetically deformed RGB-D images on the freiburg3_teddy dataset. Left: 0.1 m, middle: 0.05 m, right: 0.025 m resolutions.

of vertices between scene and model in a MRF and infer the ML correspondences through LBP. The unary potentials measure the similarity in spin image descriptors (Johnson, 1997), while pairwise potentials prefer to keep discrete nearness and farness relations. Myronenko and Song (2010) and Jian and Vemuri (2011) model the point clouds in Gaussian mixture models(GMMs). As detailed, the CPD method by Myronenko and Song (2010) estimates probabilistic assignments of points and optimizes for the displacement field between source and model. Spatial smoothness of the solution is obtained through regularizing higher-order derivatives in the displacement field using a Gaussian kernel. Jian and Vemuri (2011) impose GMMs on both point sets and minimize the L_2-norm between the mixture densities. They regularize the displacements using thin-plate splines as an alternative regularization kernel that minimizes only up to second-order derivatives (Chui and Rangarajan, 2003). Sagawa et al. (2009) extend the non-rigid ICP method of Allen et al. (2003) with soft assignments. The local transforms of each point are applied to its local neighbors and weighted averages of their errors with the other surface are used as data terms. Within a modeling system for deformable objects, Wand et al. (2009) perform registration by estimating the deformation field. The displacement at a point is modeled as a linear combination of the displacement vectors at control points. The weights themselves adapt to the local deformation in the source cloud and decay with distance of the control points from the query point.

A quite different approach has been utilized by Santa and Kato (2012). They represent the displacement field as polynomial functions and optimize for the coefficients to align the surfaces.

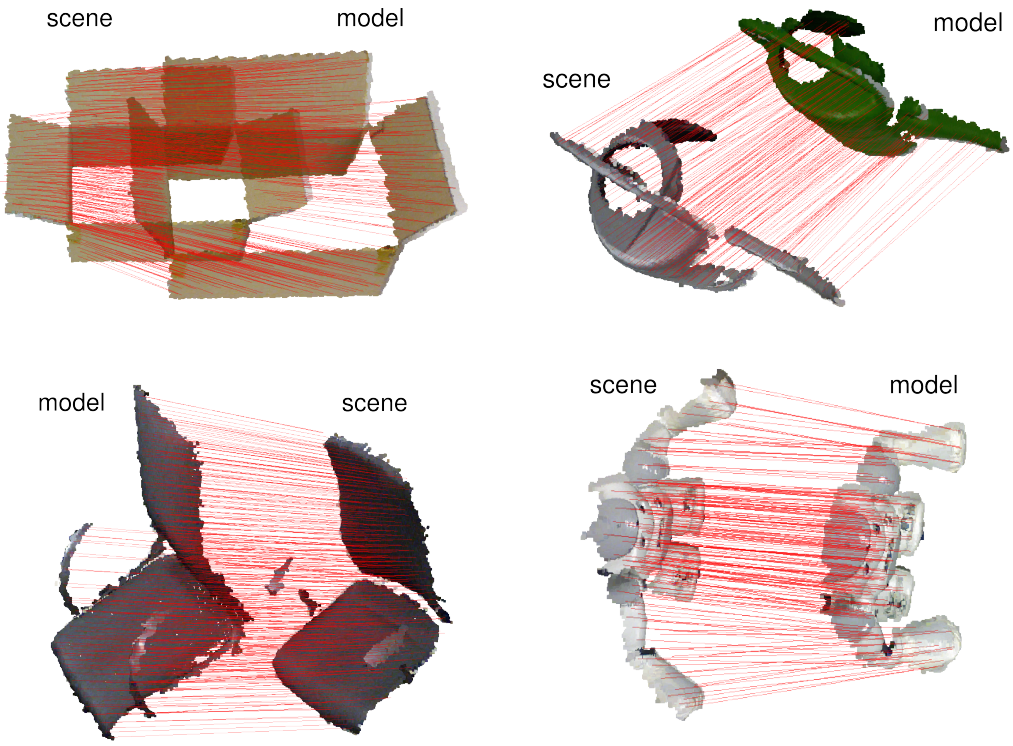

Figure 5.9.: Deformable registration examples.

Since these methods are susceptible to finding local optima, several approaches add constraints on the registration to correct for coarse misalignments. Such constraints are found using local features or by optimizing for isometric consistency. The non-rigid ICP approaches in (Allen et al., 2003) and (Amberg et al., 2007) support further energy terms that align points with feature correspondences. Anguelov et al. (2004) use similarity in spin images directly for the data terms. Tevs et al. (2009) extract slippage interest points (Bokeloh et al., 2008), describe the local surrounding of the interest points using mean curvature histograms, and determine isometrically consistent correspondences through RANSAC. Local descriptors based on heat diffusion recently attracted attention due to their affine-invariance (Sun et al., 2009; Raviv et al., 2011). The approach in (Sahillioglu and Yemez, 2012) finds isometrically consistent correspondences for points with locally maximal Gaussian curvature in an EM framework. In (Sahillioglu and Yemez, 2011), the authors use a similar approach to determine dense correspondences in a mesh through a coarse-to-fine search.

In the context of stereo and depth image processing, scene flow methods also recover displacement fields. For instance, the approach by Herbst et al. (2013) computes the 3D flow of RGB-D image pixels that maximizes color and depth

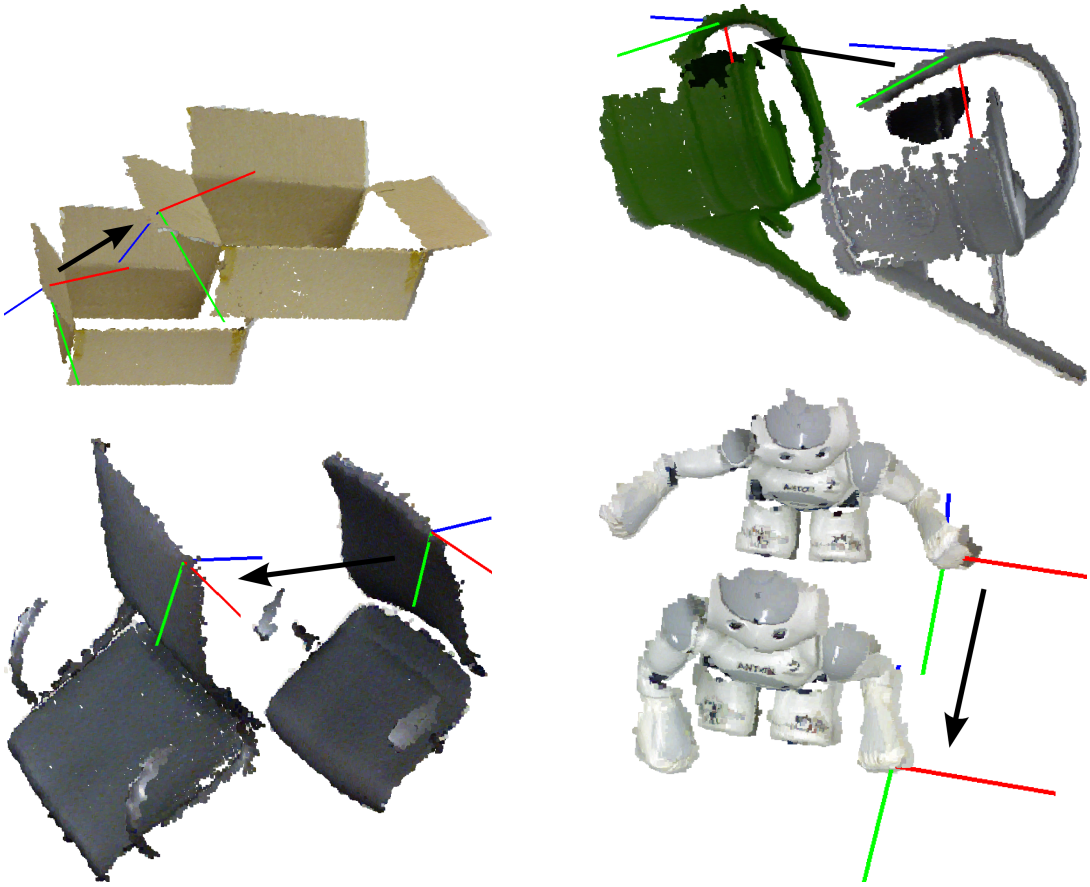

Figure 5.10.: We estimate local transformations from the displacement field from model to scene. The arrows point into the direction of the transformation from scene to model.

consistency in a regularized variational framework. It requires about 8 to 30 seconds on a CPU for processing a 320×240 image.

Most of the presented methods focus on best accuracy but often consider run-time efficiency as a secondary objective. We develop an efficient deformable registration method based on CPD that aligns RGB-D images efficiently while being sufficiently accurate for robotic applications.

5.6. Summary

We developed an efficient deformable registration method that non-rigidly aligns MRSMaps. It extends CPD with coarse-to-fine processing to keep the amount of points per resolution low and the algorithm well tractable. We also utilize color and contours as further features for data association.

For coarse-to-fine registration, the displacement fields of finer resolutions are

initialized with the coarser registration results. The resolutions in our maps are not fully redundantly represented, but the modeled surface parts may shrink with resolution due to the distance-dependent maximum resolution in MRSMaps. We propose means to respect this property in our coarse-to-fine registration method and condition the displacement field at the borders of a resolution on the solution from the coarser resolution.

As proposed by Myronenko and Song (2010) we apply the FGT and low-rank approximations to the Gram matrix. The latter implies complications in evaluating the displacement field at points that are not contained in the model point set. To still obtain a general solution of the displacement field, the Gram matrix needs to be inverted. As the dense Gram matrices of Gaussian kernels quickly become ill-conditioned, we sparsify the Gaussian kernel for improved conditioning. The sparsity of the Gram matrix can also be exploited to further improve the efficiency of matrix manipulations such as multiplication and inversion.

From the displacement fields, we derive local deformations, i.e., local rotation, translation, and strain using results from continuum mechanics. Again, the use of low-rank approximations induces difficulties, for which we propose two alternative solutions. The first involves the inversion of the Gram matrix, while the second approach approximates local deformation from the displacement of the model points. We also propose a way to estimate local deformations in the inverse direction from scene to model. We apply these local deformations to transfer grasp poses and motion trajectories for object manipulation skill transfer in Ch. 7.

In experiments, we evaluate the accuracy and run-time of our registration method. We utilize two image sequences that observe a table-top scene and an object with strong curvature and coloring. The images are synthetically deformed which provides ground-truth deformations. Finally, we demonstrate typical results for registration of views on deformed objects and local deformations estimated from the displacement field.

The high efficiency of our approach would allow for sequential registration of the current frame in a video towards a model map. If a model is given a-priori, significant computational load can be transfered to pre-processing that only needs to be done once for the model. Our method then aligns images at a rate of 1 to 5 Hz on a CPU. In future work, we will explore potential applications, for instance, to track the pose of human bodies or hands.

Our current approach does not consider the covariances of surfels in the GMM. In order for the covariances to be utilized, an efficient variant of the Gauss transform with variable, non-diagonal covariances would have to be investigated. Alternatively, the evaluation of the likelihood of the surfels under the Gaussian mixture model could be implemented on GPU. The deformation of the covariances could also be considered in future extensions.

Part II.

Scene and Object Perception

6. Modeling and Tracking of Rigid Scenes and Objects

Scene and object perception becomes challenging for robots if they shall perform tasks in environments that have not been specifically arranged. Robots need to be able to determine their own location in the environment and the types and placement of objects around them. This allows them to execute tasks, manipulate things for a purpose, or understand the actions of others.

A further requirement frequently is that robots should act immediately and fluently. This chapter develops a method for efficient 3D perception of objects and indoor scenes using RGB-D cameras. We learn dense 3D models and track the pose of the camera with respect to these models in real-time at high frame rate. Our method efficiently runs on CPUs—in contrast to many approaches that strongly rely on massive parallel computation on GPUs. While GPUs may not be available on light-weight or low-cost robot platforms, our main argument is that we devised an efficient way to represent 3D models and to align RGB-D images to them.

Our approach to 3D modeling and tracking is based on our MRSMaps and our efficient yet robust rigid registration method. Since we represent surfels for multiple distinct view-directions, we can integrate many images into multi-view 3D models.

We propose a SLAM approach for this purpose that constructs a graph of spatial constraints between key views onto the scene or an object. These spatial constraints are obtained with our registration method. In order to recover the camera trajectory, we optimize the joint likelihood of the view poses. For online mapping, we propose a loop-closing strategy that finds new spatial constraints between key views efficiently. The key views can then be overlayed in one multi-view 3D model from their optimized poses.

The acquired models can be used for tracking the camera motion in real-time on a CPU by registering the current RGB-D image to the model. By the multi-resolution nature of our maps, our method keeps track of objects in a wide range

of distances and speeds.

In many applications, the initial pose for tracking objects is not known a-priori. We adapt a state-of-the-art approach to our MRSMap representation for estimating the pose of models in a 3D point cloud. We integrate this method with tracking in a particle filter framework for global object localization. To cope with the six-dimensional pose space, we utilize registration to keep the particles focussed on the relevant part of the state space.

6.1. Background

6.1.1. Simultaneous Localization and Mapping

SLAM refers to the problem of estimating the trajectory of a sensor from observations of map elements, while concurrently building and estimating the state of the map. The problem is formally stated as the estimation of sensor poses $x_p = \left\{x_{p,0}, \ldots, x_{p,T}\right\}$ and states of map elements $x_m = \left\{x_{m,1}, \ldots, x_{m,M}\right\}$ given measurements $z = \{z_1, \ldots, z_Z\}$ and controls $u = \{u_1, \ldots, u_U\}$,

$$p(x_p, x_m \mid z, u). \tag{6.1}$$

While for bundle adjustment (BA) controls are usually not considered, the posterior of SLAM is a hidden Markov model (HMM) with motion model

$$p(x_{p,t} \mid u_t, x_{p,t-1}), \tag{6.2}$$

where u_t is the control applied in $x_{p,t-1}$ to reach $x_{p,t}$. The observation likelihood is

$$p(z_{tk} \mid x_{p,t}, x_{m,k}), \tag{6.3}$$

where z_{tk} is the observation of map element $x_{m,k}$ made in pose $x_{p,t}$. Usually not all map elements are observable in each pose. The observation of a map element is often assumed to be independent from other elements in the map. By this, the joint posterior factorizes into

$$p(x_p, x_m \mid z, u) = p(x_0) \prod_{t=1}^{T} p(x_{p,t} \mid u_t, x_{p,t-1}) \prod_{k \in \mathcal{Z}_t} p(z_{tk} \mid x_{p,t}, x_{m,k}), \tag{6.4}$$

where $p(x_0)$ is a prior on the reference frame of the first sensor pose, and $k \in \mathcal{Z}_t$ indexes map elements that are observed as z_{tk} in pose $x_{p,t}$.

The estimates of the map elements are only correlated with other map elements through sensor poses that observe both elements simultaneously. Also, sensor poses are correlated through the poses they originate from by controls, or by observing the same map element. This makes the correlation structure sparse

which is exploited for efficient optimization. In the SLAM community, the solution of such sparse non-linear least squares problems is also referred to as graph optimization, as the sparse structure is also reflected in a graph $G = (\mathcal{V}, \mathcal{E})$ of constraints between the optimization variables. Sensor poses and states of map elements are vertices $v \in \mathcal{V}$ in this constaint graph. Edges $e \in \mathcal{E}$ between vertices correspond to optimization constraints that are due to controls between sensor poses, or observations of map elements from a sensor pose.

6.1.2. SLAM Graph Optimization as Sparse Non-Linear Least Squares

We explain motions and observations by non-linear functions of the state variables, affected by Gaussian noise,

$$x_{p,t} \sim \mathcal{N}\left(g(x_{p,t-1}, u_t), \Sigma_{u,t}\right) \tag{6.5}$$

$$z_{tk} \sim \mathcal{N}\left(h(x_{p,t}, x_{m,k}), \Sigma_{z,tk}\right) \tag{6.6}$$

The prior $p(x_0) = \mathcal{N}(\mu_{x,0}, \Sigma_{x,0})$ is also modeled normal distributed around a prior mean $\mu_{x,0}$ with covariance $\Sigma_{x,0}$. With these probability distributions, the log-likelihood of the SLAM posterior has a quadratic form, i.e.,

$$\ln p(x_p, x_m \mid z, u) \approx const. - \frac{1}{2}(x_0 - \mu_{x,0})^T \Sigma_{x,0}^{-1}(x_0 - \mu_{x,0}) \tag{6.7}$$

$$- \frac{1}{2}\sum_{t=1}^{T}(x_{p,t} - g(x_{p,t-1}, u_t))^T \Sigma_{u,t}^{-1}(x_{p,t} - g(x_{p,t-1}, u_t)) \tag{6.8}$$

$$- \frac{1}{2}\sum_{t=1}^{T}\sum_{k \in \mathcal{Z}_t}(z_{tk} - h(x_{p,t}, x_{m,k}))^T \Sigma_{z,tk}^{-1}(z_{tk} - h(x_{p,t}, x_{m,k})). \tag{6.9}$$

Let $x = \left(x_{p,1}^T, \ldots, x_{p,T}^T, x_{m,1}^T, \ldots, x_{m,M}^T\right)^T$ be the full state of the SLAM problem. We define the residuals

$$e_0(x) := x_0 - \mu_{x,0}, \tag{6.10}$$

$$e_{u,t}(x) := x_{p,t} - g(x_{p,t-1}, u_t), \tag{6.11}$$

$$e_{z,tk}(x) := z_{tk} - h(x_{p,t}, x_{m,k}), \tag{6.12}$$

$$\tag{6.13}$$

and matrices $W_0 := \Sigma^{-1}$, $W_{u,t} := \Sigma_{u,t}^{-1}$, and $W_{z,tk} := \Sigma_{z,tk}^{-1}$ to write

$$\ln p(x_p, x_m \mid z, u) \approx const. - \frac{1}{2} e_0(x)^T W_0 e_0(x) \tag{6.14}$$

$$- \frac{1}{2} \sum_{t=1}^{T} e_{u,t}(x)^T W_{u,t} e_{u,t}(x) \tag{6.15}$$

$$- \frac{1}{2} \sum_{t=1}^{T} \sum_{k \in \mathcal{Z}_t} e_{z,tk}(x)^T W_{z,tk} e_{z,tk}(x), \tag{6.16}$$

which has the form of non-linear least squares problems introduced in Sec. 3.1.2. As in Sec. 3.1.2.3, we stack the residuals in a vector of residuals $e(x)$, and form a block-diagonal weight matrix W from the weights of the individual residuals to solve this multi-objective optimization problem. The sparsity of this non-linear least squares problem is apparent from the dependency of the residuals on only a few poses or map elements. Kuemmerle et al. (2011) provide the g^{2}o software framework to solve sparse non-linear least squares problems with a variety of solvers such as preconditioned conjugate gradient descent (Kelley, 1995), or using efficient sparse matrix inversion such as sparse LU- or Cholesky-decomposition (Davis, 2006) within LM optimization.

6.1.3. Particle Filters

Particle filters (e.g. Thrun et al., 2005) implement recursive Bayesian filters with discrete weighted samples $\mathcal{X}_\tau = \left\{ x_\tau^{[i]} \right\}_{i=1}^{N}$ of the state trajectory posterior

$$p\left(x_{0:t} \mid z_{1:t}, u_{1:t}\right), \tag{6.17}$$

with importance weights $\mathcal{W}_\tau = \left\{ w_\tau^{[i]} \right\}_{i=1}^{N}$, where $x_{0:t} := \{x_1, \ldots, x_t\}$ is the state trajectory to be estimated, $z_{1:t} := \{z_1, \ldots, z_t\}$ is the set of image observations from the start of filtering up to time t, and $u_{1:t} := \{u_1, \ldots, u_t\}$ are the corresponding controls. The filtered estimate of $p(x_t \mid z_{1:t}, u_{1:t})$ for the latest time step t is given by $\mathcal{X}_t$.

In each time step t, particles $\overline{\mathcal{X}}_t$ are sampled from a proposal distribution $q(x_t)$ by propagating the set of samples $\mathcal{X}_{t-1}$ from the previous time step in an informed way. In order to match the resulting distribution of particles $\mathcal{X}_t$ with the state posterior for x_t, the particles are reweighted by the mismatch between target, i.e. state posterior, and proposal distribution,

$$w_t^{[i]} = \frac{\text{target distribution}}{\text{proposal distribution}} \, w_{t-1}^{[i]}. \tag{6.18}$$

A simple approach is to propagate the particles according to the state-transition model $p(x_t \mid x_{t-1}, u_t)$. By this, the weights

$$w_t^{[i]} = \eta p\left(z_t \mid x_t^{[i]}\right) \tag{6.19}$$

are determined by the observation likelihood for the individual particles. The constant η is a normalization factor that is shared among the particles. It vanishes since the scale of the weights can be normalized to sum to one.

Since the particles are not directly sampled from the state posterior but from an intermediate proposal distribution, the particle distribution may degenerate over time and spread the particles in areas of the state space that are unlikely under the state posterior. Only a few particles receive high weights and a large fraction of the particles is dissipated with low weights. This effect can be reduced by using a more informed proposal distribution that also considers the measurement instead of only controls. Doucet et al. (2000) prove that the optimal proposal distribution in the sense of minimizing the variance of the importance weights is

$$q(x_t) = p\left(x_t \mid x_{t-1}^{[i]}, z_t, u_t\right). \tag{6.20}$$

The importance weights in this case can be shown to be

$$w_t^{[i]} := \eta p\left(z_t \mid x_{t-1}^{[i]}, u_t\right) = \eta \int p(z_t \mid x_t)\ p\left(x_t \mid x_{t-1}^{[i]}, u_t\right) dx_t. \tag{6.21}$$

See (Arulampalam et al., 2002) for a derivation. Often, a closed-form solution to the improved proposal distribution as well as to the importance weights is not available and both need to be approximated.

6.2. Scene and Object Modeling with Multi-Resolution Surfel Maps

Our MRSMap representation and rigid registration techniques are efficient tools for learning 3D models of objects or indoor scenes. Such 3D models are, for instance, useful for robot perception to detect and track objects, to plan grasps on the object, or to localize the robot in an environment. Our approach does not require prior knowledge on objects and scenes such as CAD-models. Instead our method learns models from RGB-D image sequences. We devise an efficient SLAM approach that estimates the motion of the camera, such that the images can be overlayed into a consistent multi-view 3D model, represented in a MRSMap. Once the model is available, it can be tracked in the live images of the camera.

Naïve sequential registration of images, i.e., visual odometry, would be prone to drift, which would not allow to overlay the images in a consistent map. Such

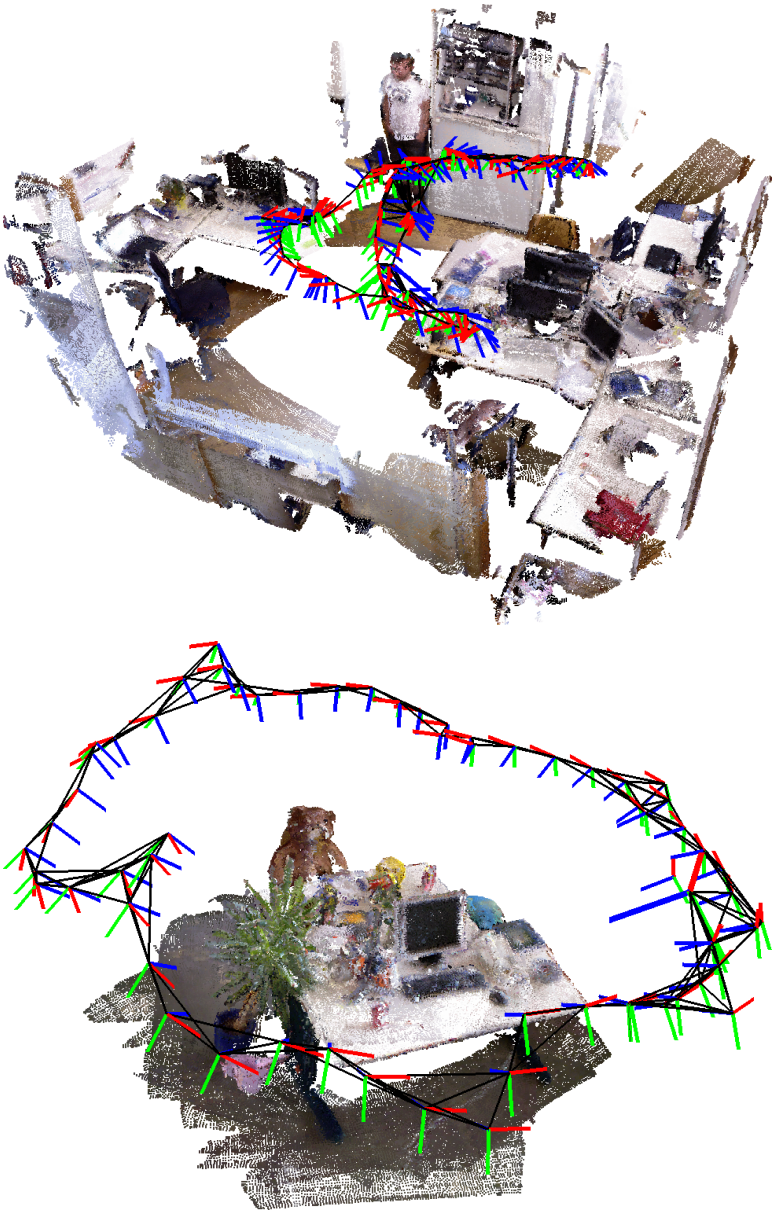

Figure 6.1.: Learned MRSMaps of indoor scenes and associated key view graphs. Top: freiburg1_room. Bottom: freiburg2_desk. Key views (poses visualized by coordinate frames) are extracted along the camera trajectory. Spatial constraints between key views (black lines) are established using our registration method. The surfel distributions are visualized by samples from the surfels at $\rho^{-1} = 0.05\,\mathrm{m}$ for two distinct view directions.

drift can be avoided, if images are not only registered to each other in temporal sequence, but if further images are related to each other that view overlapping

parts of the scene. Registration inaccuracies can then be traded in a SLAM framework. Larger drifts in the trajectory estimate typically occur when the sensor views new, previously unmapped volume. As soon as the sensor reaches known volume again, the drift-prone part of the trajectory is linked with the older parts of the trajectory through image registration. SLAM then distributes the accumulated drift along the trajectory to balance the registration estimates and to adjust the pose differences. This event is denoted as loop closure.

Our modeling approach is a variant of graph-based SLAM. We extract a set of key views $v_i \in \mathcal{V}$ along the camera trajectory. Each key view $v_i = (x_i, m_i)$ is described by a camera pose x_i and a MRSMap m_i. In order to keep track of the camera motion, the current image is registered towards the closest key view in the map which we denote as reference key view v_{ref}. Since registration quality degrades with the view pose difference between images, a new key view is generated from the current image if the camera moved sufficiently far, i.e., if the rotational or translational distance towards the reference key view reached a threshold. The rigid registration result x_i^j (Sec. 3.2.2.2) with its covariance estimate $\Sigma(x_i^j)$ (Sec. 3.2.2.3) between the new key view v_i and its reference v_j is a spatial constraint that we maintain as edges $e_{ij} \in \mathcal{E}$ in a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ of key views (see Fig. 6.1).

We find additional spatial constraints between key views in a hypothesis-and-test scheme that also detects loop-closures. It tests one spatial constraint per frame to enable online operation.

Graph optimization then yields an estimate of the key view poses. On-line operation is achieved by iterating the graph optimization only once per frame. Finally, we fuse the key views by overlaying their images in one multi-view MRSMap from their resulting pose estimates.

6.2.1. Constraint Detection

On each frame, we test for one new constraint between the current reference v_{ref} and other key views v_{cmp}. We choose any key view whose pose estimate is sufficiently close to the reference key view. It is important to validate the registration of the key views, since if the key views barely or fully not overlap, our registration method may find suboptimal solutions that—if included—could let the pose graph optimization diverge. We examine the matching likelihood of each of the MRSMaps to the other. In order to test a constraint only once, we maintain lists of key view pairs that have already been tested. If a new constraint is added, we empty these lists for the involved key views such that constraints can be tested again from the optimized poses that now consider the new constraint.

6.2.1.1. Map Matching Likelihood

Let m_s be the map for which we evaluate the likelihood of being observed in the other map m_m, and let x be the relative pose estimated through registration. Analogous to Sec. 4.2.6.1, we define the observation likelihood of the surfel as

$$p(s_s \mid s_m, x) = \mathcal{N}\left(d^*(s_s, s_m, x); 0, \Sigma^*(s_s, s_m, x)\right) \mathcal{N}\left(\arccos(n_m^T R(x)\, n_s), (\sigma_0^n)^2\right), \tag{6.22}$$

where we take the spatial and color distribution of the surfels into account. We also measure the compatibility of the surface normals (Eq. (4.75)) with a uncertainty parameter σ_0^n.

We limit the observation likelihood of outlier matches of surfels and surfels without matches to p_O. In this way, the matching likelihood

$$p(m_s \mid m_m, x) = \prod_{s_s \in m_s} p(s_s \mid s_m, x) \tag{6.23}$$

accounts for the overlap between the views. If a scene surfel is behind a model surfel along its view direction, i.e., the scene surfel is seen through the model surfel, we assign a probability $p_T < p_O$ to penalize such unreasonable observations.

We associate each surfel $s_s \in m_s$ with a surfel $s_m \in m_m$ that yields best observation likelihood within a cubic search volume. If multiple view directions are represented, we choose the surfel s_m for which the view direction v_m best matches to the rotated view direction $R(x)\, v_s$. We adapt the cube length $l = 2\,\rho(s_s)^{-1}$ of the search volume to the resolution of the surfel s_s. We only query surfels in m_m that are on the same resolution like s_s, since we want to measure the overlap of the level of detail of both maps.

6.2.1.2. Constraint Validation

The matching likelihood in Eq. (6.23) is directional and, hence, we evaluate it in both directions. We cannot use a global threshold for deciding if a constraint x_i^j should be added, as the matching likelihood clearly depends on image content which leads to varying distributions of surfels across resolutions. Instead we require the matching likelihoods $p(m_i \mid m_j, x_i^j)$ and $p(m_j \mid m_i, (x_i^j)^{-1})$ of a new constraint to be at least a fraction γ_m of the matching likelihoods of the key views with themselves,

$$\left(p(m_i \mid m_j, x_i^j) \geq \gamma_m\, p(m_i \mid m_i, id)\right) \wedge \left(p(m_j \mid m_i, x_j^i) \geq \gamma_m\, p(m_j \mid m_j, id)\right), \tag{6.24}$$

where id is the pose corresponding to the identity transform.

6.2.2. Key-View Pose Graph Optimization

In our key-view-based SLAM approach, the map elements correspond to the complete MRSMaps attached to the key views. A key view's sensor pose and the reference frame of its map coincide, such that in our case the SLAM posterior reduces to

$$p(x \mid z) = p(x_0) \prod_{i=1}^{N} \prod_{j \in \mathcal{Z}_i} p(z_{ij} \mid x_i, x_j) \tag{6.25}$$

where N is the number of key views, $z_{ij} := x_i^j$ are registration estimates from key views i to j, and $j \in \mathcal{Z}_i$ indexes the set of key views j that key view i has been registered to.

Our registration approach yields an estimate of the covariance $\Sigma(x_i^j)$ of the pose estimate, such that the probability of the relative pose observation z_{ij} is

$$p(z_i^j \mid x_i, x_j) = \mathcal{N}\left(x_i^j; x_j \ominus x_i, \Sigma(x_i^j)\right), \tag{6.26}$$

where $\ominus$ finds the relative pose from key view i to key view j, i.e., $T(x_j \ominus x_i) = T(x_j)^{-1} T(x_i)$. This operator is non-linear for our pose parametrization in translations and quaternions.

By taking the logarithm of Eq. (6.25) we obtain

$$\ln p(x \mid z) = const. - \frac{1}{2}(x_0 - \mu_{x,0})^T \Sigma_{x,0}^{-1} (x_0 - \mu_{x,0}) \tag{6.27}$$

$$- \frac{1}{2} \sum_{i=1}^{N} \sum_{j \in \mathcal{Z}_i} (x_i^j - (x_j \ominus x_i))^T \Sigma_{z,ik}^{-1} (x_i^j - (x_j \ominus x_i)). \tag{6.28}$$

We see that our SLAM problem transforms into a sparse non-linear least squares problem as in Sec. 6.1.2. We solve this graph optimization problem within the g^{2}o framework (Kuemmerle et al., 2011) by the LM method with sparse Cholesky decomposition for matrix inversion.

6.2.3. Obtaining Scene and Object Models from Key View Graphs

Once the pose of each key view is known, we overlay the key views in a single MRSMap. The resulting map is multi-view, i.e., it contains a scene or object model from various view points. A multi-view map does not exhibit the local multi-resolution structure of a map created from a single view-point. Rather the local multi-resolution maps of each image are overlayed in a complex multi-resolution structure. For this fusion process we construct the tree directly from the original RGB-D measurements. Each key view corresponds to an RGB-D image which is transformed into the common map frame according to the pose estimate of the key view.

Figure 6.2.: Learned MRSMaps of objects. The surfel distributions are visualized by samples from the surfels at $\rho^{-1} = 0.05\,\mathrm{m}$ (top row) and $\rho^{-1} = 0.025\,\mathrm{m}$ (bottom row) for two distinct view directions.

For creating object models, we restrict the fused model to the measurements on the object. A simple approach to segment the object out of the images is to place the object on a planar surface and to manually select a convex hull around the object on the support plane. Those RGB-D measurements are then extracted that project onto the plane from a specific height interval and into the convex hull.

6.3. Object Detection and Real-Time Tracking

Once a model of a scene or an object is available, it can be used for detecting the object in a scene and estimating its pose, and keeping track of the object pose.

6.3.1. Detecting Objects and Estimating Pose with Multi-Resolution Surfel Maps

Tracking requires an initial guess on the object pose. In many applications, however, the object's pose is not known a-priori and needs to be estimated from the images. We adapt a state-of-the-art approach to object detection and pose estimation in point clouds to our MRSMap framework.

Our object detection method is based on the surfel-pair voting algorithm proposed by Drost et al. (2010) which has been recently extended for RGB-D images with color by Choi and Christensen (2012a). Our contribution is a pose voting

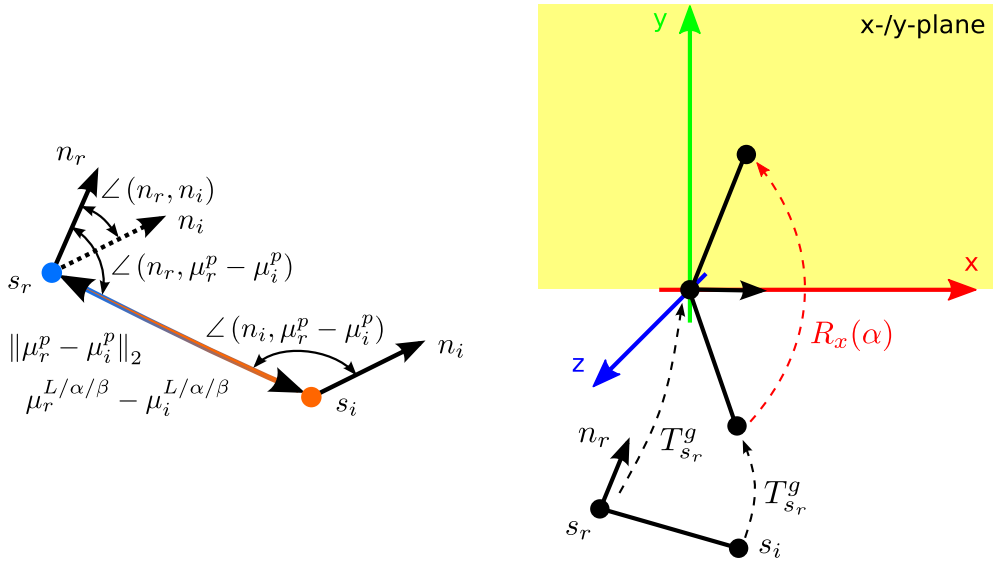

Figure 6.3.: Surfel-pairs, features, and constructed reference frames. Left: We describe geometry, luminance, and color between two surfels by distance, angular relations of normals, and luminance and color contrasts. Right: A surfel-pair defines a unique pose in the map frame by aligning the normal of the reference surfel s_r with the x-axis of the map frame and rotating the paired surfel s_i by an angle α around the x-axis onto the half-plane spanned by the x- and positive y-direction.

scheme that utilizes surfel-pairs at multiple resolutions in varying local neighborhoods. The aim of object detection and pose estimation is to find an object in an RGB-D image by aligning a MRSMap m_s of the image with an object model MRSMap m_m.

6.3.1.1. Local Colored Surfel-Pair Relations at Multiple Resolutions

As in (Drost et al., 2010), we describe the geometric relation between a pair of surfels $f(s_r, s_i) := (f_s(s_r, s_i), f_c(s_r, s_i))$ with the geometric descriptors

$$f_s(s_r, s_i) := \left(\left\| \mu_r^p - \mu_i^p \right\|_2, \ \angle\left(n_r, \mu_r^p - \mu_i^p\right), \ \angle\left(n_i, \mu_r^p - \mu_i^p\right), \ \angle(n_r, n_i) \right) \quad (6.29)$$

that measure distance and angles between means and normals (see Fig. 6.3, left). In addition, we incorporate color by the three luminance and chrominance contrasts

$$f_c(s_r, s_i) := \left(\mu_r^L - \mu_i^L, \ \mu_r^\alpha - \mu_i^\alpha, \ \mu_r^\beta - \mu_i^\beta \right). \quad (6.30)$$

Different to the approach of Drost et al. (2010), we only consider surfel-pairings for a reference surfel s_r in a local neighborhood around the surfel. The

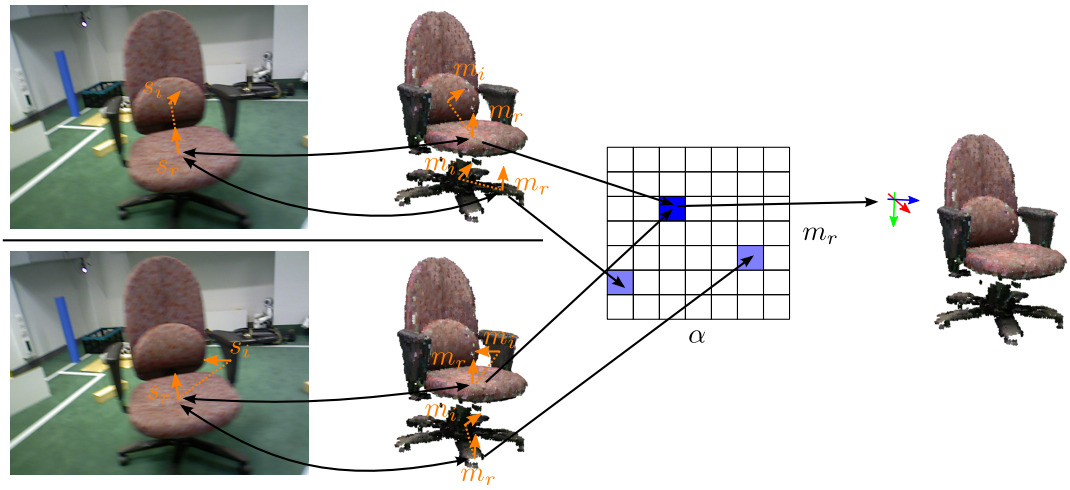

Figure 6.4.: Surfel-pair voting. Each association of surfel-pairs between scene and model votes for a 6-DoF camera pose relative to the model in a two-dimensional Hough space.

radius $r_\rho = \lambda_r \, \rho(s_r)^{-1}$ of the neighborhood is set in relation with the surfel's resolution. We also neglect surfel-pairs with similar normals, luminance, and chrominances to avoid ambiguous pose voting from planar, textureless regions.

6.3.1.2. Multi-Resolution Pose Voting

A surfel-pair defines a unique coordinate frame through the normal direction of the reference surfel and the difference between the means as long as the difference is not parallel to the normal, which is unlikely to happen in practice. This frame is used to define the pose of the surfel-pair relative to the reference frame of a map. We follow the approach of Drost et al. (2010) and decompose this pose into a transformation $T^g_{s_r}$ that moves the mean μ_r of the reference surfel into the map origin and aligns its normal n_r with the map x-axis (see Fig. 6.3, right). A final rotation around the x-axis with angle $\alpha(s_r, s_i)$ moves the paired surfel mean μ_i into the half-plane spanned by the x- and y-axes with positive y-values. If we decompose the pose in this way, all pairings of the reference surfel share the same transformation $T^g_{s_r}$ and only differ in angle α.

From a correct match of surfel-pairs between two maps we are able to estimate the pose difference between the maps. Let $(s_{s,r}, s_{s,i})$ and $(s_{m,r}, s_{m,i})$ be two matching surfel-pairs in scene and model map, respectively. The pose difference T^m_s between the map reference frames can be determined from

$$\begin{aligned} T^m_s &= \left(R_x(\alpha(s_{m,r}, s_{m,i}))\, T^g_{s_{m,r}}\right)^{-1} \left(R_x(\alpha(s_{s,r}, s_{s,i}))\, T^g_{s_{s,r}}\right) \\ &= \left(T^g_{s_{m,r}}\right)^{-1} \; R_x(\alpha) \; T^g_{s_{s,r}} \end{aligned} \tag{6.31}$$

with $\alpha = \alpha(s_{s,r}, s_{s,i}) - \alpha(s_{m,r}, s_{m,i})$.

For object detection problems, however, correct matches of surfel-pairs between scene and model map are not known a-priori, but need to be estimated with the object pose. Drost et al. (2010) propose a Hough voting scheme in which surfel-pairs are matched according to their geometric descriptor and cast votes for the object pose. For efficient matching, hash keys are determined from the descriptors to map surfel-pairs in a hash table. The descriptors of the surfel-pairs are quantized into a number of bins per dimension to form the keys.

From a matching of surfel-pairs, a potential object pose is determined by the index r of the matched model reference surfel $s_{m,r}$ and the angle $\alpha = \alpha(s_{s,r}, s_{s,i}) - \alpha(s_{m,r}, s_{m,i})$ that aligns the surfel-pairs (see Fig. 6.4). Hough voting is efficiently performed in this two-dimensional pose space. Each model reference surfel is considered individually in the Hough space, while the angles α are discretized into a number of bins. To increase the precision of the Hough procedure, we attribute a continuous angle estimate for a surfel match to the two closest angle bins.

We process scene reference surfels per available resolution, and, to achieve fast run-time, sample a fraction of the scene reference surfels uniformly without replacement. Pose votes are separately accumulated in a Hough space for each scene reference surfel $s_{s,r}$. The local surfel-pairings of the reference surfel $s_{s,r}$ with other scene surfels $s_{s,i}$ are matched with surfel-pairs $(s_{m,r}, s_{m,i})$ via their descriptors through efficient hash map look-ups. Multiple matchings may be retrieved for the scene pair. Each matching votes for a pose in the two-dimensional Hough space. After all pairings for the scene reference surfel $s_{s,r}$ have been processed, the bin that accumulated most pose votes is determined and a pose hypothesis is extracted. We also include pose hypotheses from Hough space bins that received a fraction of votes below the maximum. Each pose hypothesis is assigned a score that corresponds to its accumulated votes.

In order to find the most consistent pose hypotheses across all scene reference surfels, we merge the pose hypotheses using agglomerative clustering with a threshold on the linear and angular distance of the poses. Since agglomerative clustering depends on the ordering of the pose hypotheses, we sort the hypotheses for their scores in descending order. The algorithm finally returns the top C clusters which accumulate the highest score of pose hypotheses.

6.3.1.3. Pose Verfication

The resulting pose hypotheses of our voting method are only coarse object pose estimates. Also, the voting method does only consider positive information for matching. It does not validate if pose hypotheses would observe parts of the model in front of actual measurements, i.e. the measurements would be seen through the model. We therefore perform a pose verification step to increase the rate of retrieving correct hypothesis.

Each pose hypothesis is registered towards the model from its pose estimates using a few iterations of LM registration (see Sec. 3.2.2). We determine the matching likelihood of scene to model map for the optimized poses according to Sec. 6.2.1.1 and reorder the pose hypotheses by their matching likelihood.

6.3.2. Tracking through Registration

The aim of tracking is to maintain an estimate of the camera pose towards a model in real-time while the camera is continuously streaming new images. This necessitates efficient and robust means to align the current image with the model.

A simple approach is to aggregate the current RGB-D image in a MRSMap and align it with the model using our rigid registration method. As we perform tracking, we have a pose estimate x^m_{t-1} available from the last frame $t-1$ to initialize the registration of the current frame. In the first frame, we obtain an estimate of the object pose through our object detection method (see Sec. 6.3.1)

Map aggregation and registration is made even more efficient by saving unnecessary computations in image regions that are unlikely to view the model. We only aggregate image points into a MRSMap that are likely under the spatial distribution of the object model, given the last camera pose estimate x^m_{t-1}. Mean μ_O and covariance Σ_O of this distribution are readily obtained from the sum of surfel statistics $|\mathcal{P}|$, $\mathcal{S}(\mathcal{P})$, and $\mathcal{S}^2(\mathcal{P})$ over all view directions in the root node of the tree.

We transform this distribution into the camera frame,

$$\mu^c_O := R(x^m_{t-1})^T \mu_O, \tag{6.32}$$

$$\Sigma^c_O := R(x^m_{t-1})^T \Sigma_O R(x^m_{t-1}), \tag{6.33}$$

$$\tag{6.34}$$

and find those image pixels p that are likely under the distribution, i.e.

$$-\frac{1}{2}(p-\mu^c_O)^T(\Sigma^c_O)^{-1}(p-\mu^c_O) \geq -\chi_m, \tag{6.35}$$

We adapt the threshold $\chi_m := k_m \|\mu^c_O\|_2$ to the distance of the model mean from the camera. This compensates for the effects of camera rotation, i.e., the farer the camera from the object, the larger the object model is displaced in the camera image by the same amount of camera rotation.

6.3.3. Object Tracking with Particle Filters

Our detection method often yields multiple object hypotheses that need to be verified further through post-processing. We improve the robustness of our pose verification method by evaluating the matching likelihood of the pose hypotheses

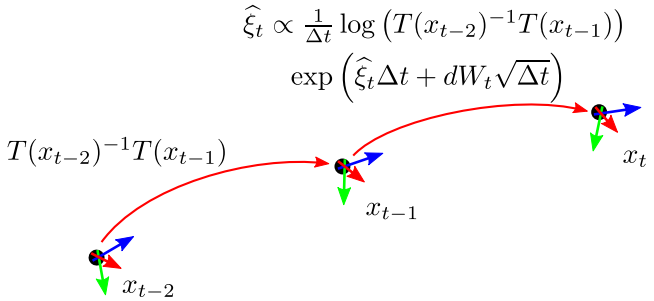

Figure 6.5.: Auto-regressive state-transition model. Particles are propagated according to the twist ξ_t estimated from the previous two time steps, affected by Wiener process noise dW_t.

over multiple frames within a particle filter framework. For instance, details that allow for disambiguating views on the object may not be immediately visible in the first frame. The particle filter resumes tracking from the detected pose hypotheses using an auto-regressive motion model and an improved proposal distribution that utilizes our MRSMap registration method.

A further advantage of using a particle filter over tracking-by-optimization is the maintenance of multiple pose hypotheses instead of only a single one. In difficult situations such as fast camera motions, partial occlusions, or ambiguous views on the object, tracking with a single hypothesis may fail, since it does not represent uncertainty in pose. In our particle filter framework, tracking-by-optimization is performed with several pose hypotheses. It is integrated with object detection to initialize the tracked poses or to reinitialize the filter if tracking cannot be resumed.

6.3.3.1. State-Transition Model

We propagate each particle with a guess of its current velocity using an auto-regressive (AR) state dynamics model. For representing 6-DoF poses and velocities we choose the $\mathbb{SE}(3)$ group and its associated Lie algebra $\mathfrak{se}(3)$. Members $T \in \mathbb{SE}(3)$ are homogenous transformation matrices while elements $\xi \in \mathfrak{se}(3)$ are twists $\xi = \left(v^T, \omega^T\right)^T$ with linear and angular velocities v and ω. The exponential map $T = \exp(\widehat{\xi}\,\Delta t)$ transforms twists into transformation matrices. Its inverse is the logarithmic map $\widehat{\xi}\,\Delta t = \log(T)$. With $\widehat{\xi}$ we denote the representation of twists as matrices in $\mathbb{R}^{4\times 4}$,

$$\widehat{\xi} := \begin{pmatrix} 0 & -\omega_z & \omega_y & v_x \\ \omega_z & 0 & -\omega_x & v_y \\ -\omega_y & \omega_x & 0 & v_z \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{6.36}$$

There exists a one-to-one mapping $T(x)$ between poses x parametrized in translation vector and quaternion for rotation, and homogeneous transformation matrices, thus we will continue to refer to the particle state as poses x.

As in Choi and Christensen (2012b) we model the state-transition by the first-order, discrete-time AR state dynamics

$$\begin{aligned} T(x_t) &= T(x_{t-1}) \ \exp\left(\widehat{\xi}_{t-1}\,\Delta t + dW_t\sqrt{\Delta t}\right) \\ \widehat{\xi}_{t-1} &= \lambda_{AR}\ \frac{1}{\Delta t}\ \log\left(T(x_{t-2})^{-1}\,T(x_{t-1})\right), \end{aligned} \tag{6.37}$$

with process parameter λ_{AR}. Uncertainty in the state transition is introduced through the Wiener process noise $dW_t\sqrt{\Delta t}$ with $dW_t = \sum_{i=1}^{6} \epsilon_{i,t} E_i$. The random variable $\epsilon_t \sim \mathcal{N}(0, \Sigma_\xi)$ is normal distributed and adds noise in the twist coordinates through the basis elements E_i of $\mathfrak{se}(3)$,

$$E_1 := \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, E_2 := \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad E_3 := \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \tag{6.38}$$

$$E_4 := \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, E_5 := \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \quad E_6 := \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}. \tag{6.39}$$

This state-transition model estimates the velocity of a particle from the poses of the last two time steps. The process parameter λ_{AR} allows for adjusting the scale of this velocity according to the confidence in the velocity estimate. We parametrize the noise $\epsilon_{i,t} = \epsilon^0_{i,t} + \epsilon^v_{i,t}\,|v|$ with constant noise $\epsilon^0_{i,t}$ and a component $\epsilon^v_{i,t}$ that scales with linear or rotational velocity.

6.3.3.2. Observation Model

Observations are RGB-D images z_t. We transform the current image into a MRSMap $m_{s,t}$ and determine the likelihood of observing the current image in the model map using the matching likelihood in Sec. 6.2.1.1,

$$p(z_t \mid x_t) = p(m_{s,t} \mid x_t, m_m). \tag{6.40}$$

6.3.3.3. Improved Proposal Distribution

If we would utilize the state-transition model for the proposal distribution, many particles would be required to cover the 7-dimensional pose space well for accurate and robust tracking. Instead, we propose to use an improved proposal

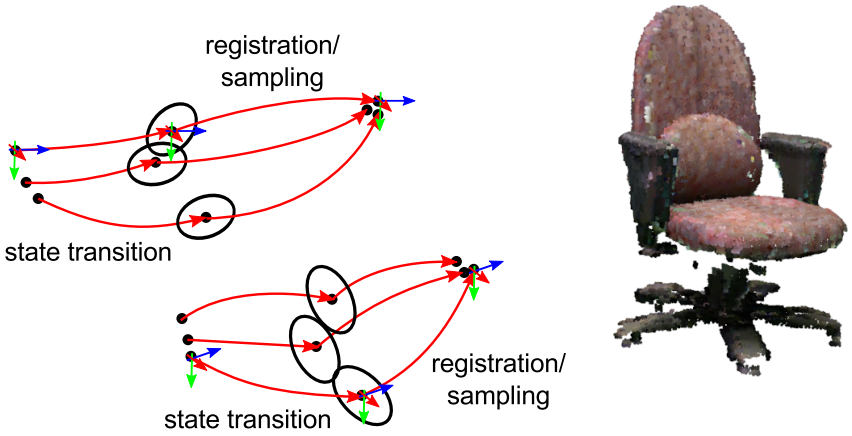

Figure 6.6.: Particle filtering with improved proposal distributions. Each particle is registered from its predicted pose. The registration is regularized by the pose distribution determined from the state-transition model. Regularized registration yields an improved proposal that the particles are sampled from.

distribution that also considers the current RGB-D image z_t to obtain a good guess on the pose of the particles already in the sampling step. The particles $\mathcal{X}_{t-1}$ are first propagated individually according to the motion model towards new predictions $\overline{\mathcal{X}}_{t-1}$. We optimize the predicted particle poses to align the current image with the model using our registration method (see Fig. 6.6). The improved proposal distribution

$$\begin{aligned} p\left(x_t^{[i]} \mid x_{t-1}^{[i]}, z_t, u_t\right) &= p\left(x_t^{[i]} \mid m_m, x_{t-1}^{[i]}, m_{s,t}\right) \\ &= \eta^{[i]}\ p(m_{s,t} \mid x_t^{[i]}, m_m)\, p\left(x_t^{[i]} \mid x_{t-1}^{[i]}\right). \end{aligned} \tag{6.41}$$

is normal distributed with the regularized registration estimate $\widetilde{x}_t^{[i]}$ as mean with associated uncertainty $\Sigma\left(\widetilde{x}_t^{[i]}\right)$ (see Sec. 3.2.2.4). In order to approximate $p\left(x_t^{[i]} \mid x_{t-1}^{[i]}\right)$ with a normal distribution in $x_t^{[i]}$, we apply the unscented transform (Julier and Uhlmann, 1997). We propagate sigma points of the process noise through the state-transition model and recover mean and covariance of the pose distribution from the propagated sigma points.

6.3.3.4. Importance Weights

The importance weights $\mathcal{W}_t$ of the particles are reweighted according to the mismatch between the target and the proposal distribution (see Sec. 6.1.3).

With our choice of proposal distribution, the importance weights are

$$\begin{aligned} w^{[i]} &= p\left(z_t \mid x_{t-1}^{[i]}, u_t\right) \\ &= \left(\eta^{[i]}\right)^{-1} = \int p(m_{s,t} \mid x_t, m_m)\, p\left(x_t \mid x_{t-1}^{[i]}, \xi_t\right) dx_t. \end{aligned} \tag{6.42}$$

The weights correspond to the observation likelihood under the predicted distribution for the particle's pose according to the state-transition model. We consider the uncertainty in the predicted pose for our observation likelihood in Eq. (6.23), and propagate the uncertainty in twist to the difference measures between surfels and normals.

The matching likelihood of surfels in Eq. (6.22) has two factors which both involve the pose variable in a non-linear mapping. Due to the neglectance of correlations between spatial and color dimensions, we can focus on the spatial dimensions and define the differences between the spatial and the color surfel distributions as

$$\begin{aligned} d^p(s_s, s_m, x_t) &:= \mu_{s,m}^p - T(x_t)\mu_{s,s}^p, \text{ and} \\ d^c(s_s, s_m, x_t) &:= \mu_{s,m}^c - \mu_{s,s}^c, \end{aligned} \tag{6.43}$$

respectively. Pose uncertainty only propagates to the spatial difference. In order to propagate twist uncertainty, we reformulate the spatial difference as a function of twist ξ, the pose from the previous time step x_{t-1}, and the time increment Δt,

$$d^p(s_s, s_m, \xi, x_{t-1}, \Delta t) := \mu_{s,m}^p - T\left(x_{t-1}\right) \exp\left(\widehat{\xi}\Delta t\right) \mu_{s,s}^p. \tag{6.44}$$

Using first-order error propagation, we obtain the covariance contributed to the spatial difference

$$\Sigma_\xi^p(s_s, s_m, \xi, x_{t-1}, \Delta t) := J_\xi^p \Sigma_\xi \left(J_\xi^p\right)^T, \tag{6.45}$$

with $J_\xi^p := \nabla_\xi d^p(s_s, s_m, \xi, x_{t-1}, \Delta t)$. It adds to the spatial covariances of the surfels, such that the total covariance of the spatial difference is

$$\Sigma^p(s_s, s_m, \xi, x_{t-1}, \Delta t) := \Sigma_{s,m}^p + R(x_t)\Sigma_{s,s}^p R(x_t)^T + \Sigma_\xi^p(s_s, s_m, \xi, x_{t-1}, \Delta t), \tag{6.46}$$

where $T(x_t) = T(x_{t-1}) \exp\left(\widehat{\xi}\Delta t\right)$. To determine the derivative J_ξ^p, we approximate the spatial difference by

$$d^p(s_s, s_m, \xi, x_{t-1}, \Delta t) \approx \mu_{s,m}^p - T\left(x_{t-1}\right) \left(I + \widehat{\xi}\Delta t\right) \mu_{s,s}^p \tag{6.47}$$

through truncating the series expansion of the exponential map. The derivative for ξ then approximately is

$$J_\xi^p \approx -\Delta t\; T\left(x_{t-1}\right) \left(\nabla_\xi \widehat{\xi}\right) \mu_{s,s}^p. \tag{6.48}$$

We also consider twist uncertainty for the angular difference $d^n(s_s, s_m, x_t) := \arccos(n_m^T R(x_t)\, n_s)$ of the normals. We rephrase the angular difference in terms of the rotational velocity ω of the twist, previous pose, and time difference, i.e.,

$$d^n(s_s, s_m, \omega, x_{t-1}, \Delta t) := \arccos(n_m^T R(x_{t-1}) \exp(\widehat{\omega}\Delta t)\, n_s), \tag{6.49}$$

and approximate the exponential map such that

$$d^n(s_s, s_m, \omega, x_{t-1}, \Delta t) \approx \arccos(n_m^T R(x_{t-1})\, (I + \widehat{\omega}\Delta t)\, n_s), \tag{6.50}$$

Through first-order error propagation, we determine the variance

$$\sigma_\xi^n(s_s, s_m, \omega, x_{t-1}, \Delta t)^2 := J_\xi^n \Sigma_\xi \left(J_\xi^n\right)^T, \tag{6.51}$$

where we defined $J_\xi^n := \nabla_\omega d^n(s_s, s_m, \omega, x_{t-1}, \Delta t)$. It contributes to the total variance of the normal estimate

$$\left(\sigma^n(s_s, s_m, \omega, x_{t-1}, \Delta t)\right)^2 := (\sigma_0^n)^2 + \left(\sigma_\xi^n(s_s, s_m, \omega, x_{t-1}, \Delta t)\right)^2. \tag{6.52}$$

The derivative approximately is

$$J_\xi^n \approx -\frac{\Delta t}{\sqrt{1 - d^n(s_s, s_m, x_t)^2}}\, n_m^T R(x_{t-1})\, (\nabla_\omega \widehat{\omega})\, n_s. \tag{6.53}$$

In summary, the resulting observation likelihood for the scene map m_s of the current image z_t is

$$p(m_s \mid m_m, x_{t-1}, \xi_t) \approx \prod_{s_s \in m_s} p(s_s \mid s_m, x_{t-1}, \xi_t) \tag{6.54}$$

$$= \prod_{s_s \in m_s} \int p(s_s \mid s_m, x_t)\, p(x_t \mid x_{t-1}, \xi_t) dx_{t-1} \tag{6.55}$$

with

$$\int p(s_s \mid s_m, x_t)\, p(x_t \mid x_{t-1}, \xi_t) dx_{t-1} = \tag{6.56}$$

$$\mathcal{N}\left(d^p(s_s, s_m, x_t); 0, \Sigma^p(s_s, s_m, \xi_t, x_{t-1}, \Delta t)\right) \tag{6.57}$$

$$\cdot \mathcal{N}\left(d^c(s_s, s_m); 0, \Sigma^c(s_s, s_m)\right) \tag{6.58}$$

$$\cdot \mathcal{N}\left(d^n(s_s, s_m, x_t), (\sigma^n(s_s, s_m, \omega_t, x_{t-1}, \Delta t))^2\right), \tag{6.59}$$

and $\Sigma^c(s_s, s_m) := \Sigma^c(s_s) + \Sigma^c(s_m)$.

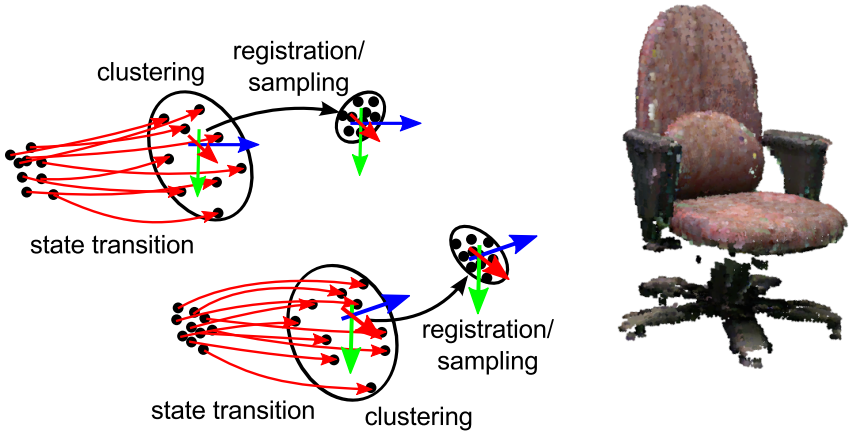

Figure 6.7.: Improved proposals on particle clusters. We gain computational efficiency by clustering closeby particles and establishing an improved proposal per cluster.

6.3.3.5. Efficient Approximation to the Improved Proposal Distribution

Registering each particle individually at high frame rates would be computationally demanding. Instead, we propose to identify modes of the density estimate $p(x_t \mid \mathcal{X}_{t-1})$, to cluster the particles that belong to the mode, and to perform only a single registration per cluster. Fig. 6.7 illustrates this approach. We employ a clustering of the particles with a fixed threshold on translation and rotation. For efficient clustering, a kd-tree is constructed from the position estimates of the particles. Particles in a limited volume and with similar orientations are clustered together until all particles have been assigned.

In order to construct an improved proposal for each cluster as in Eq. (6.41), registration is performed starting from the mean of a cluster. The pose distribution for the state-transition model is approximated using the mean velocity of the particles in the cluster and their mean pose from the previous time step.

The resulting improved proposal is used to sample the particles in the same cluster. The importance weights of each particle are still evaluated separately for each particle by using individually predicted pose estimates $\overline{x}_t^{[i]}$ in Eq. (6.54). Surfel associations are shared between the particles within a cluster to further increase efficiency. If the particles are distributed within a single cluster, we limit the processing of the RGB-D image to the relevant parts as in Sec. 6.3.2.

We further note that when the estimate of the tracker is good, the discrete distribution given by the particles typically has a single mode. However, after initialization or when uncertainy increases, multiple modes need to be considered.

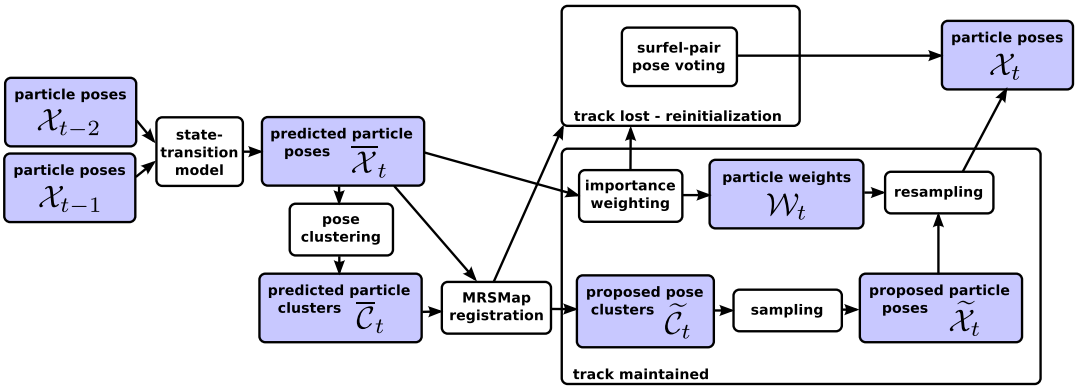

Figure 6.8.: Joint object detection, pose estimation, and tracking in a particle filter framework.

6.3.4. Joint Object Detection, Pose Estimation, and Tracking in a Particle Filter Framework

We integrate object detection and pose estimation with our particle filter tracking approach in a joint framework (see Fig. 6.8). It not only allows for tracking without a-priori knowledge of the object pose, but also makes tracking robust for occlusions and registration failures.

6.3.4.1. Initialization

We use our surfel-pair voting algorithm (Sec. 6.3.1) to efficiently find C_0 pose hypotheses for an object in a scene. Pose verification as detailed in Sec. 6.3.1.3 is not immediately required. Instead, the multiple pose hypotheses will be resolved by the particle filter over time. From each pose hypothesis, we sample N_C particles from a normal distribution centered at the pose hypothesis.

6.3.4.2. Tracking

Once initialized, tracking proceeds using the particle filter approach described in Sec. 6.3.3. In each iteration, we set the number of particles N_t to

$$N_t = \max\left\{N_{\min}, N_C \cdot \min\left\{C_{t-1}, C_0\right\}\right\}, \tag{6.60}$$

where $N_{\min}$ is the minimum number of particles used, and C_{t-1} is the number of clusters tracked in the last iteration. Limiting the considered number of clusters to C_0 prevents from unbound growth of the number of particles.

6.3.4.3. Reinitialization

For extreme camera movements, the motion model could propagate the particles far away from the actual camera pose. Also if the object is occluded in large

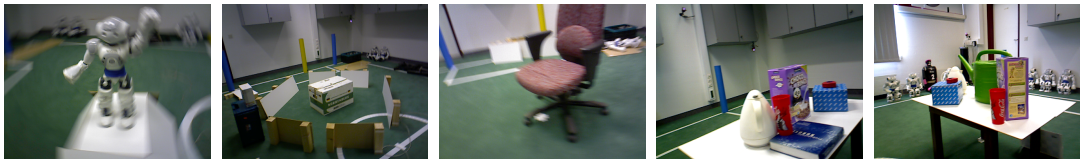

Figure 6.9.: Example images from object tracking sequences (from left to right: humanoid fast, box medium, chair fast, cereal1, watering can2).

	angular velocity (rad/s)		linear velocity (m/s)		object distance (m)		occlusions
sequence	mean	std	mean	std	mean	max	
humanoid slow	0.37	0.39	0.26	0.23	0.97	1.26	none
humanoid medium	0.53	0.55	0.40	0.36	0.99	1.42	none
humanoid fast	0.59	0.79	0.47	0.46	1.01	1.52	none
box slow	0.37	0.36	0.23	0.22	1.23	1.42	none
box medium	0.41	0.43	0.36	0.33	1.35	2.49	none
box fast	0.63	0.75	0.47	0.49	1.39	1.89	none
chair slow	0.37	0.59	0.32	0.53	1.37	1.75	none
chair medium	0.46	0.51	0.38	0.34	1.37	1.51	none
chair fast	0.58	0.65	0.52	0.53	1.46	1.91	none
watering can1	0.23	0.24	0.20	0.20	1.21	1.44	full
watering can2	0.26	0.24	0.24	0.20	1.13	1.38	partial
cereal1	0.26	0.25	0.25	0.20	1.09	1.57	partial
cereal2	0.23	0.23	0.18	0.16	0.82	1.01	partial

Table 6.1.: Properties of the object tracking sequences.

parts, or the object leaves the field-of-view of the camera, tracking may not be possible. We detect tracking failures if too few surfels could be matched between scene and model, or if the maximum observation likelihood drops. We reinitialize the particle filter using our object detection method until the track is maintained again.

6.4. Experiments

We use the RGB-D benchmark dataset (Sturm et al., 2012) to evaluate our SLAM approach in indoor scenes. This dataset has also been used to assess our rigid registration approach in Ch. 3.

For object modeling and tracking, we use the RGB-D object tracking dataset[1]. It also provides 640×480 VGA resolution image sequences recorded with an Asus Xtion Pro Live camera at 30 Hz. The sequences contain up to 1100 frames, and

[1]available from http://www.ais.uni-bonn.de/download/objecttracking

part	parameter	setting
MRSMaps	distance-dependent resolution factor λ_ρ	0.02
object detection	neighborhood radius factor λ_r	16
object detection	detected top hypotheses C_0	5
object tracking	image cut-out, object proximity k_m	0.5
object tracking	AR process parameter λ_{AR}	0.4
object tracking	constant angular variance $\epsilon^0_{i,t}$	0.001
object tracking	velocity-dependent angular variance $\epsilon^v_{i,t}$	0.1
object tracking	constant linear variance $\epsilon^0_{i,t}$	0.01
object tracking	velocity-dependent linear variance $\epsilon^v_{i,t}$	0.1
object tracking	minimum number of particles $N_{\min}$	25
object tracking	number of particles per cluster N_C	10

Table 6.2.: Default parameter settings.

have ground truth trajectories available that have been recorded with an OptiTrack motion capture system. The dataset consists of 13 sequences of 5 objects. Fig. 6.9 shows example images from the sequences. Three objects of varying sizes (a humanoid robot, a box, and a chair) are contained in test sequences with slow, moderate, and fast camera motion. The dataset also includes two test sequences with difficult occlusion situations on a cereal box and a watering can. Table 6.1 lists properties of the sequences such as angular and linear camera velocity and distance to the objects.

Run-time and real-time evaluation have been conducted on a PC with an Intel Core i7-4770K QuadCore CPU at a maximum clock speed of 3.50 GHz. If not stated otherwise, we use the parameter settings in Table 6.2 which have been determined empirically.

6.4.1. Evaluation Measures

For the evaluation of SLAM systems, Sturm et al. (2012) propose to assess the translational average root mean squared error (RMSE) of the RPE measure over all time steps and time differences,

$$\frac{1}{T}\sum_{\Delta=1}^{T}\left(\frac{1}{T}\sum_{t=1}^{T}\left\|\mathrm{trans}(E_{\Delta,t})\right\|_2^2\right)^{\frac{1}{2}} \tag{6.61}$$

where T is the length of the trajectory in time steps, and trans extracts the translational components of the pose difference $E_{\Delta,t}$. The measure quantifies not only frame-to-frame pose differences or the difference of pose estimates at the start and end of the trajectory, but also considers all intermediate time differences between poses.

The root mean squared absolute trajectory error (ATE) (Sturm et al., 2012) is an alternative measure that has a more intuitive interpretation: The position trajectory estimate is aligned with the ground-truth trajectory, and the RMSE of the position differences is calculated. Alignment is required since the reference frames of both trajectory estimates are different. It is simplified, since the measurement times in both trajectories are known such that the association of the poses is determined.

We also employ the RPE and the ATE measures to quantify tracking accuracy.

6.4.2. SLAM in Indoor Scenes

We demonstrate accuracy and run-time efficiency of our SLAM approach on sequences of the RGB-D benchmark dataset. In Table 6.3 and Fig. 6.10, we compare our approach with RGB-D SLAM, a method that matches and aligns interest points between frames and also performs pose graph-optimization using the g2o framework. In eight out of eleven sequences, our method outperforms RGB-D SLAM in terms of RPE, if we process all frames in the sequences. When run under real-time constraints, our approach is required to drop frames. As we do not use a real-time operating system, the real-time constraint is a source of randomness. Hence, we average the performance of our method over 10 runs. In real-time mode, our method achieves similar accuracy in ATE and RPE than if we process every frame. When using all frames, our dense approach in average achieves an improvement towards RGB-D SLAM in ATE by about 0.013 m and in RPE by about 0.028 m. For real-time, our method yields an average improvement of 0.012 m and 0.024 m, respectively.

Typical trajectories and maps obtained can be seen in Figs. 6.11 and 6.1. The estimates well follow the ground truth trajectory. In both sequences, the camera is moving on a long trajectory loop through office scenes. In freiburg1_room the camera moves on a loopy trajectory while mostly pointing outward of the motion curve, while in freiburg2_desk it looks inwards onto a table-top scene. The freiburg1_room sequence contains many smaller trajectory loops which are not closed within the sequence. Hence, the overall trajectory loop is closed by our approach, but drift remains in some parts of the trajectory.

Table 6.4 details the run-time required by our method. In average, it is ca. 61 ms and at most 223 ms. The average run-time per sequence mostly depends on camera pose tracking which requires the registration of the current image with the closest key view in the map. The variability in time consumed for registration is explained by the diversity of distance of the camera from the measured surfaces in the sequences. Adding a new key view involves the estimation of the pose covariance and the evaluation of the base-line matching likelihood for constraint validation. This takes about 5.6 ms in average. We do not check for new constraints though when adding a new key view. To check for a new constraint, the key views need to be registered and the matching likelihood is

Table 6.3.: Accuracy of our SLAM approach and RGB-D SLAM in absolute trajectory (ATE) and relative pose error (RPE).

	RMSE ATE in m			RMSE RPE in m		
sequence	ours all fames	ours real-time	RGB-D SLAM	ours all frames	ours real-time	RGB-D SLAM
freiburg1_360	0.076	**0.074**	0.079	0.115	0.115	**0.103**
freiburg1_desk	0.028	0.032	**0.023**	**0.048**	0.055	0.049
freiburg1_desk2	**0.038**	0.050	0.043	**0.077**	0.091	0.102
freiburg1_plant	**0.025**	0.026	0.091	**0.040**	0.042	0.142
freiburg1_room	**0.056**	0.062	0.084	**0.112**	0.123	0.219
freiburg1_rpy	0.028	0.029	**0.026**	**0.039**	0.041	0.042
freiburg1_teddy	0.061	**0.051**	0.076	**0.089**	0.091	0.138
freiburg1_xyz	0.015	**0.013**	**0.014**	0.022	**0.021**	**0.021**
freiburg2_desk	0.058	**0.057**	0.095	**0.103**	0.106	0.143
freiburg2_rpy	0.029	0.029	**0.019**	0.040	0.042	**0.026**
freiburg2_xyz	**0.023**	**0.023**	0.026	**0.031**	0.032	0.037
average difference to RGB-D SLAM	**-0.013**	-0.012		**-0.028**	-0.024	

evaluated between the key views in both directions. This amounts to approximately 27.3 ms in average. Fig. 6.12 gives further insights into the evolution of run-time in the freiburg1_room and freiburg2_desk sequences. While the run-time for pose graph optimization exhibits approximately linear growth, it uses only a small fraction of the total time. Clearly, for mapping larger volumes, a different SLAM back-end would be required that limits run-time consumption to a constant. Such back-ends are still subject to active research.

6.4.3. Learning 3D Object Models

For learning object models, we process the sequences off-line. In each frame, new constraints between all pairs of key views are tested and added, if their map matching is valid (see Sec. 6.2.1.2). The registration of the 5% edges with worst residuals in the SLAM graph are reestimated. If the camera trajectory returns close to the camera view pose of the start of the sequence, we include an edge between the first and the last key view. After all frames are processed, the registration estimates are recomputed from the new relative pose estimates.

We show the recovered camera trajectories for all 5 objects in Fig. 6.13. Fig. 6.2 shows models learned with our approach. The estimated trajectory follows the ground truth estimate accurately. For comparison, we show the trajectory obtained, if we only tracked towards a closest key view. The trajectory estimate would drift in scenes that contain extended periods in which the camera

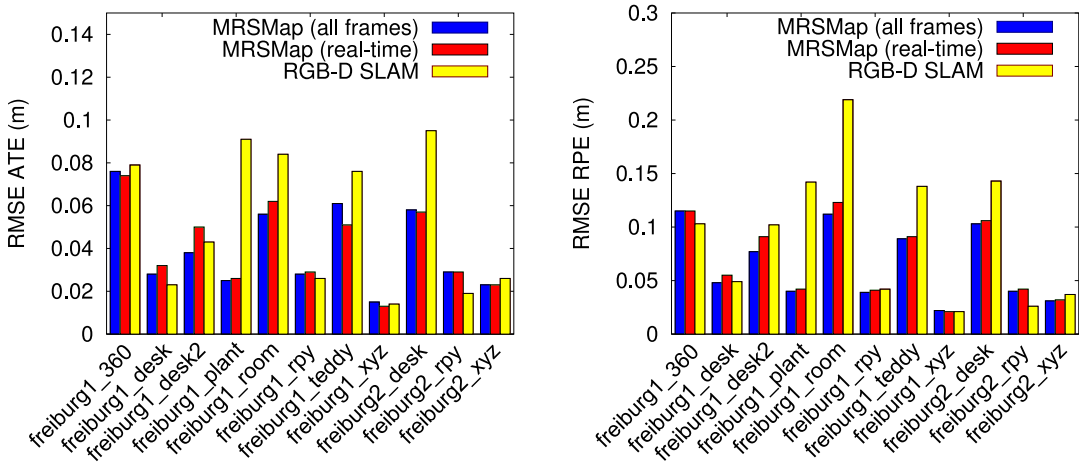

Figure 6.10.: Accuracy of our SLAM approach and RGB-D SLAM in absolute trajectory (ATE) and relative pose error (RPE).

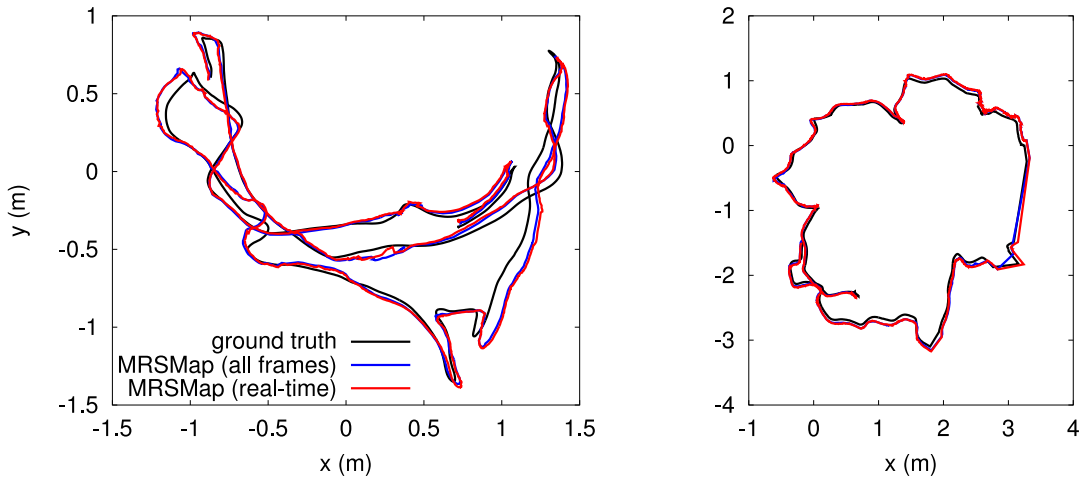

Figure 6.11.: Ground truth (black) and trajectory estimates obtained using all frames (blue) and in real-time (red) on the freiburg1_room (left) and freiburg2_desk (right) sequences.

views novel scene content. Pose graph optimization corrects for the drift.

We provide the minimum, median, and maximum ATE in Table 6.5. The median accuracy is between 1-3 cm. In sequences without loops, pure registration performs similar well as graph optimization. Graph optimization quantitatively improves the trajectory estimate for sequences with long loops.

Table 6.4.: Average (max.) run-time in ms and max. graph size of our SLAM approach on RGB-D benchmark sequences.

sequence	tracking	key view addition	constraint addition	graph optim.	total
freiburg1_360	45.3 (118.1)	4.5 (8.7)	23.6 (64.5)	1.6 (5.8)	51.3 (142.3)
freiburg1_desk	65.8 (127.9)	6.5 (9.4)	32.8 (70.4)	1.0 (3.4)	74.7 (157.3)
freiburg1_desk2	66.0 (133.1)	6.6 (9.1)	40.1 (146.2)	1.3 (4.3)	80.0 (222.7)
freiburg1_plant	49.2 (92.5)	5.1 (7.6)	23.7 (57.8)	1.9 (8.7)	54.4 (122.1)
freiburg1_room	56.1 (133.0)	5.7 (9.9)	28.6 (76.0)	1.8 (6.7)	63.3 (153.5)
freiburg1_rpy	62.5 (96.9)	6.6 (8.2)	33.2 (50.9)	1.1 (5.1)	69.6 (134.1)
freiburg1_teddy	46.3 (123.5)	4.3 (6.3)	19.4 (73.7)	3.1 (13.1)	54.4 (133.1)
freiburg1_xyz	67.3 (114.1)	6.4 (8.3)	28.2 (37.9)	0.2 (0.8)	68.2 (121.3)
freiburg2_desk	49.7 (107.7)	5.1 (6.3)	24.7 (65.8)	1.4 (6.1)	52.2 (121.4)
freiburg2_rpy	46.1 (72.8)	5.3 (6.8)	21.8 (33.3)	0.4 (1.4)	46.8 (94.2)
freiburg2_xyz	54.1 (89.7)	5.4 (8.3)	24.3 (65.0)	0.3 (0.9)	54.7 (119.1)
overall	55.3 (133.1)	5.6 (9.9)	27.3 (146.2)	1.3 (13.1)	60.9 (222.7)

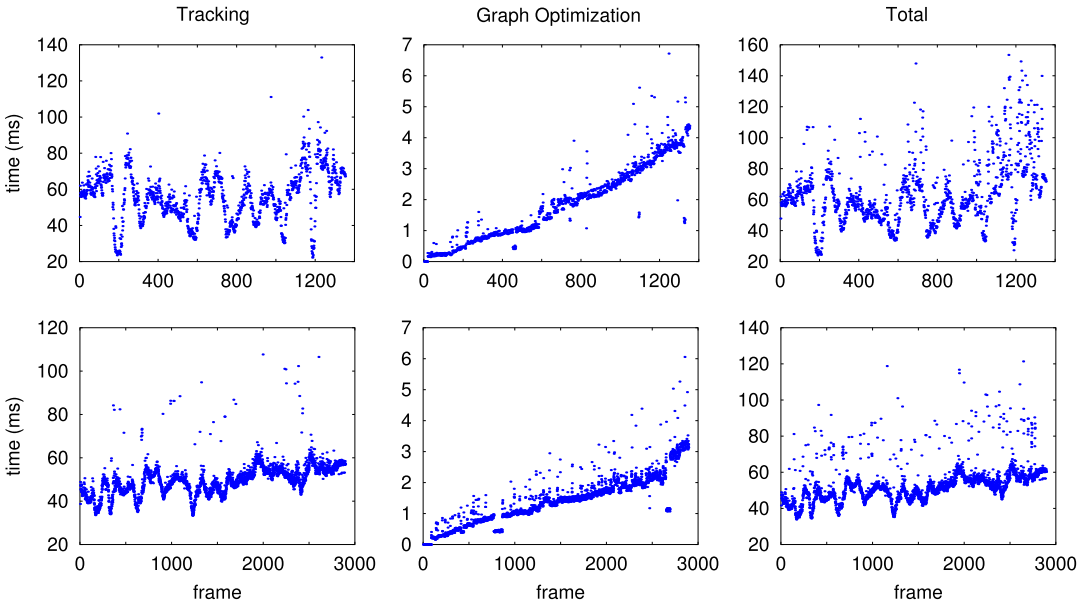

Figure 6.12.: Timing on the freiburg1_room (top) and freiburg2_desk (bottom) sequences.

6.4.4. Object Detection and Pose-Estimation

In Table 6.6, we compare several variants of our object detection and pose estimation method to demonstrate the effects of multi-resolution processing, pose validation, and subsampling. We detect up to 50 pose hypotheses in 1,000 frames

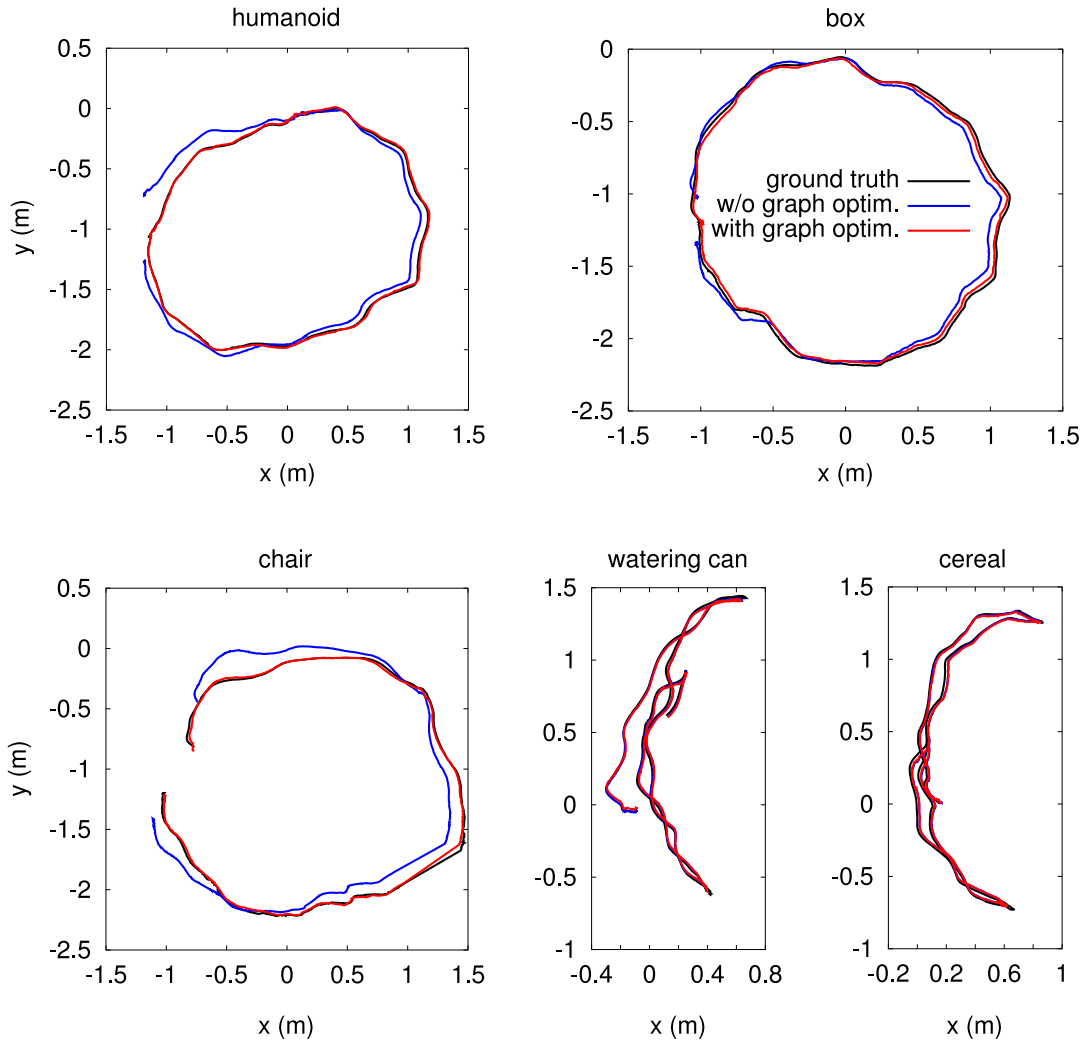

Figure 6.13.: Ground truth (black) and trajectory estimates obtained without graph optimization (blue) and with graph optimization (red) on the object model training sequences.

of each sequence and average results. Results for variants with subsampling of surfel-pairs have been additionally averaged over 10 runs with different random subsamplings. We accept a pose hypothesis as true positive, if it is within an angular and a linear distance to the ground truth pose of 15° and 0.2 m, respectively. Otherwise, it is a false positive. A false negative is accounted for if no true positive has been found.

The unvalidated variants already provide high recall rates and high average ranks for the first occurence of a true positive. Subsampling surfel-pairs at the same neighborhood radius does barely harm results. Only on the humanoid slow sequence, using every surfel-pair shows an improvement, while on the chair slow

		w/o graph optimization			with graph optimization		
sequence	loop	min	median	max	min	median	max
humanoid	yes	19.0	68.3	353.3	**2.0**	**14.5**	**63.5**
box	yes	19.9	68.6	182.0	**4.4**	**20.5**	**35.9**
chair	yes	32.5	126.3	408.8	**0.4**	**25.7**	**94.7**
watering can	no	**4.0**	**15.7**	**47.3**	5.1	17.8	51.8
cereal	no	**4.2**	19.7	65.5	7.9	**19.0**	**62.5**

Table 6.5.: Absolute trajectory error in mm obtained by incremental mapping without graph optimization and by our object modeling approach (with graph optimization).

sequence it performs worse than subsampling. This could be explained by the small size and redundant shapes of the humanoid object. Dense sampling supports in disambiguating the few surfel matches. Conversely, surfels on the chair have many redundant matchings for which subsampling seems to be beneficial. Redundancy stems here from the small relative context with respect to the scale of the chair's shape.

For the neighborhood radius factor λ_r there is no clear best choice between $\lambda_r = 8$ and $\lambda_r = 16$. On the humanoid slow and chair slow sequences, the recall rates behave differently with respect to the settings of λ_r. This supports our observation that the parameter depends on the scale of the object shapes.

The run-time of the unvalidated variants clearly is affected by the choice of neighborhood radius factor and subsampling. On all sequences, subsampling the surfel-pairs and using $\lambda_r = 8$ is significantly faster than the other variants.

While validating the pose hypotheses through registration degrades run-time performance, it significantly improves the recall rates to very high values close to 1, even if only the top 5 pose hypotheses are considered. In the majority of cases, it also clearly increases the accuracy of the pose estimate. It keeps the high ranking for the true pose hypothesis.

6.4.5. Object Tracking

Tables 6.7 and 6.8 show the accuracy of our tracking methods on the object tracking dataset, while Table 6.9 contains timing results. As particle filtering and real-time processing involves a source of randomness, we average results over 20 runs.

When all frames are processed without real-time constraints, our particle filter approach exhibits smaller median and maximum ATE in the majority of sequences. We attribute this to the use of a motion model that prevents the registration from overfitting to the available observations. Since we evaluate

Table 6.6.: Average recall for various max. number of top hypotheses, first true positive rank, and run-time in s of different variants of our object detection and pose estimation method.

sequence	variant	avg. recall			rank	accuracy		time
		5	20	50		trans	rot	
humanoid slow	all, $\lambda_r = 16$	0.89	0.90	0.90	**1.23**	0.087	0.112	0.48
	sub, $\lambda_r = 8$	0.86	0.90	0.91	1.98	0.085	0.120	**0.26**
	sub, $\lambda_r = 16$	0.85	0.87	0.88	1.58	0.086	0.109	0.42
	sub, $\lambda_r = \infty$	0.85	0.87	0.87	1.57	0.087	0.110	0.59
	sub, $\lambda_r = 8$, val.	**0.96**	0.99	0.99	1.37	**0.042**	0.046	1.18
	sub, $\lambda_r = 16$, val.	**0.96**	**0.998**	**0.999**	1.56	**0.042**	**0.045**	1.39
box slow	all, $\lambda_r = 16$	0.99	0.99	0.99	**1.02**	0.052	0.049	6.32
	sub, $\lambda_r = 8$	0.94	0.94	0.94	1.07	0.070	0.059	**0.86**
	sub, $\lambda_r = 16$	0.99	0.99	0.99	1.03	0.053	**0.047**	1.73
	sub, $\lambda_r = \infty$	0.87	0.87	0.87	1.05	0.068	0.057	2.18
	sub, $\lambda_r = 8$, val.	0.97	0.98	0.98	1.19	**0.049**	0.049	1.92
	sub, $\lambda_r = 16$, val.	**0.998**	**0.998**	**0.998**	1.10	**0.049**	0.049	2.47
chair slow	all, $\lambda_r = 16$	0.78	0.78	0.78	1.08	0.087	0.064	1.89
	sub, $\lambda_r = 8$	0.79	0.79	0.80	1.16	0.112	0.086	**0.28**
	sub, $\lambda_r = 16$	0.82	0.82	0.82	**1.07**	0.094	0.067	0.51
	sub, $\lambda_r = \infty$	0.91	0.91	0.91	1.17	0.091	0.063	0.72
	sub, $\lambda_r = 8$, val.	**0.96**	**0.97**	**0.97**	1.15	**0.044**	**0.043**	1.51
	sub, $\lambda_r = 16$, val.	0.94	0.94	0.95	1.18	**0.044**	0.044	1.69
cereal 2	all, $\lambda_r = 16$	0.97	0.97	0.97	**1.08**	0.051	0.107	0.26
	sub, $\lambda_r = 8$	0.96	0.97	0.97	1.15	0.052	0.108	**0.15**
	sub, $\lambda_r = 16$	0.97	0.97	0.97	**1.08**	0.051	0.107	0.26
	sub, $\lambda_r = \infty$	0.97	0.97	0.97	**1.08**	0.051	0.107	0.41
	sub, $\lambda_r = 8$, val.	**0.991**	**0.997**	**0.997**	1.30	**0.049**	0.091	0.41
	sub, $\lambda_r = 16$, val.	0.99	0.995	0.995	1.35	**0.049**	**0.090**	0.54
watering can 2	all, $\lambda_r = 16$	0.99	0.99	0.99	**1.004**	0.048	0.055	0.41
	sub, $\lambda_r = 8$	0.98	0.98	0.98	1.02	0.060	0.064	**0.21**
	sub, $\lambda_r = 16$	0.99	0.99	0.99	**1.004**	0.048	0.055	0.40
	sub, $\lambda_r = \infty$	0.99	0.99	0.99	1.006	0.049	0.057	0.58
	sub, $\lambda_r = 8$, val.	**1.00**	**1.00**	**1.00**	1.007	0.027	0.035	0.91
	sub, $\lambda_r = 16$, val.	**1.00**	**1.00**	**1.00**	1.009	**0.026**	**0.034**	0.99

pure tracking performance without reinitialization, the watering can 1 sequence could not be processed by both methods, as it contains a full occlusion. Remarkably, the particle filter is able to track on the cereal 1 sequence despite partial occlusions and only one plane of the box-shaped object being visible at times. Tracking-by-optimization fails on this sequence.

Under real-time constraints, the improved robustness of the particle filter com-

	incremental registration			particle filter		
sequence	min	median	max	min	median	max
humanoid slow	**1.4**	20.3	235.0	1.8	**17.3**	**39.7**
humanoid medium	**1.7**	23.0	124.8	**1.7**	**18.7**	**85.4**
humanoid fast	**1.4**	23.4	339.2	1.6	**20.3**	**123.6**
box slow	4.2	22.7	78.1	**3.7**	**21.1**	**49.1**
box medium	13.2	43.6	642.7	**7.0**	**33.3**	**191.1**
box fast	**0.4**	22.5	158.4	3.4	**18.2**	**73.7**
chair slow	2.5	**16.5**	**74.2**	**0.9**	23.6	93.9
chair medium	3.0	19.7	**66.3**	**1.8**	**16.8**	69.9
chair fast	**1.8**	**23.6**	**114.7**	2.9	33.0	224.0
cereal 1	–	–	–	**1.8**	**27.1**	**278.4**
cereal 2	2.5	24.2	523.6	**1.3**	**20.8**	**138.2**
watering can 1 (full occlusion)	–	–	–	–	–	–
watering can 2	1.1	17.9	161.2	**0.7**	**17.4**	**97.1**

Table 6.7.: Absolute trajectory error in mm obtained for object tracking (all frames processed) by incremental registration and particle filtering (without reinitialization).

	incremental registration			particle filter		
sequence	min	median	max	min	median	max
humanoid slow	**1.4**	20.0	270.6	1.6	**17.5**	**87.3**
humanoid medium	1.4	23.1	343.1	**1.0**	**19.8**	**288.5**
humanoid fast	2.0	24.7	742.2	**0.2**	**21.6**	**125.2**
box slow	1.8	26.8	100.0	**0.9**	**21.5**	**50.9**
box medium	–	–	–	**2.9**	**42.7**	**203.9**
box fast	**0.9**	25.9	240.0	1.1	**19.9**	**78.6**
chair slow	1.2	**17.0**	**73.3**	**0.6**	22.2	187.8
chair medium	0.9	19.5	**77.0**	**0.8**	**18.4**	99.0
chair fast	**1.4**	**25.0**	**182.5**	2.5	33.6	250.4
cereal 1	–	–	–	**0.5**	**28.6**	**365.2**
cereal 2	–	–	–	**0.4**	**20.9**	**142.4**
watering can 1 (full occlusion)	–	–	–	–	–	–
watering can 2	**0.1**	**18.0**	276.7	0.2	18.5	**149.2**

Table 6.8.: Absolute trajectory error in mm obtained for object tracking (real-time processing) by incremental registration and particle filtering (without reinitialization).

pared to tracking-by-optimization becomes even more apparent. Again, particle filtering provides better accuracy in most sequences. Tracking-by-optimization now additionally fails in some of the 20 runs on the box medium and cereal 2 sequences. Our particle filter approach handles all 20 runs robustly.

sequence	incremental registration		particle filter	
	avg. time (all frames)	frames used (real-time)	avg. time (all frames)	frames used (real-time)
humanoid slow	**28.4**	**85.5**	35.5	65.2
humanoid medium	**27.5**	**86.6**	35.8	66.0
humanoid fast	**29.0**	**84.7**	36.8	68.1
box slow	**35.8**	**63.8**	58.0	48.8
box medium	**36.0**	–	56.4	**54.2**
box fast	**31.0**	**80.8**	48.2	53.1
chair slow	**39.3**	**54.7**	65.1	42.9
chair medium	**39.7**	**56.1**	64.7	42.8
chair fast	**39.5**	**59.2**	62.6	45.4
cereal 1	–	–	**21.8**	**98.4**
cereal 2	**25.0**	–	28.8	**92.9**
watering can 2	**21.8**	**97.9**	24.8	95.5

Table 6.9.: Average timing in ms and frames used in % in real-time mode for object tracking by incremental registration and particle filtering (without reinitialization).

Tracking-by-optimization in average processes frames at 32.1 ms, i.e., 31.2 Hz. Throughout the sequences, the average processing rate is close to the 30 Hz image acquisition rate of the camera (ca. 33.3 ms). If an image arrives before the processing is finished, frames need to be dropped eventually. Particle filtering introduces run-time overhead, but still is very efficient at 44.9 ms in average. It demonstrates robustness to the dropping of frames.

6.4.6. Joint Object Detection, Pose Estimation, and Tracking

In Fig. 6.14, we show results of joint detection and tracking in global object localization experiments. We start tracking in each of 1,000 frames of the sequences, for which we initialize tracking with our object detection method. We used a different random subsampling of the surfel-pairs in each initialization. Instead of explicitly validating the pose hypotheses after detection, the particle filter validates the hypotheses over multiple frames. Precision and recall are here determined by measuring the angular and linear distance of the particles to the ground truth pose.

When all frames are processed, precision converges with the recall rates to high values. While the box seems to be a simple shape at first, its planarity at rectangular angles and high symmetry about the object center pose difficulties to our approach. Its rectangular shape often makes only one or two sides of the box being visible for extended periods of time. Due to motion blur, only little texture cues are available to resolve the symmetry. Hence, the tracker converges

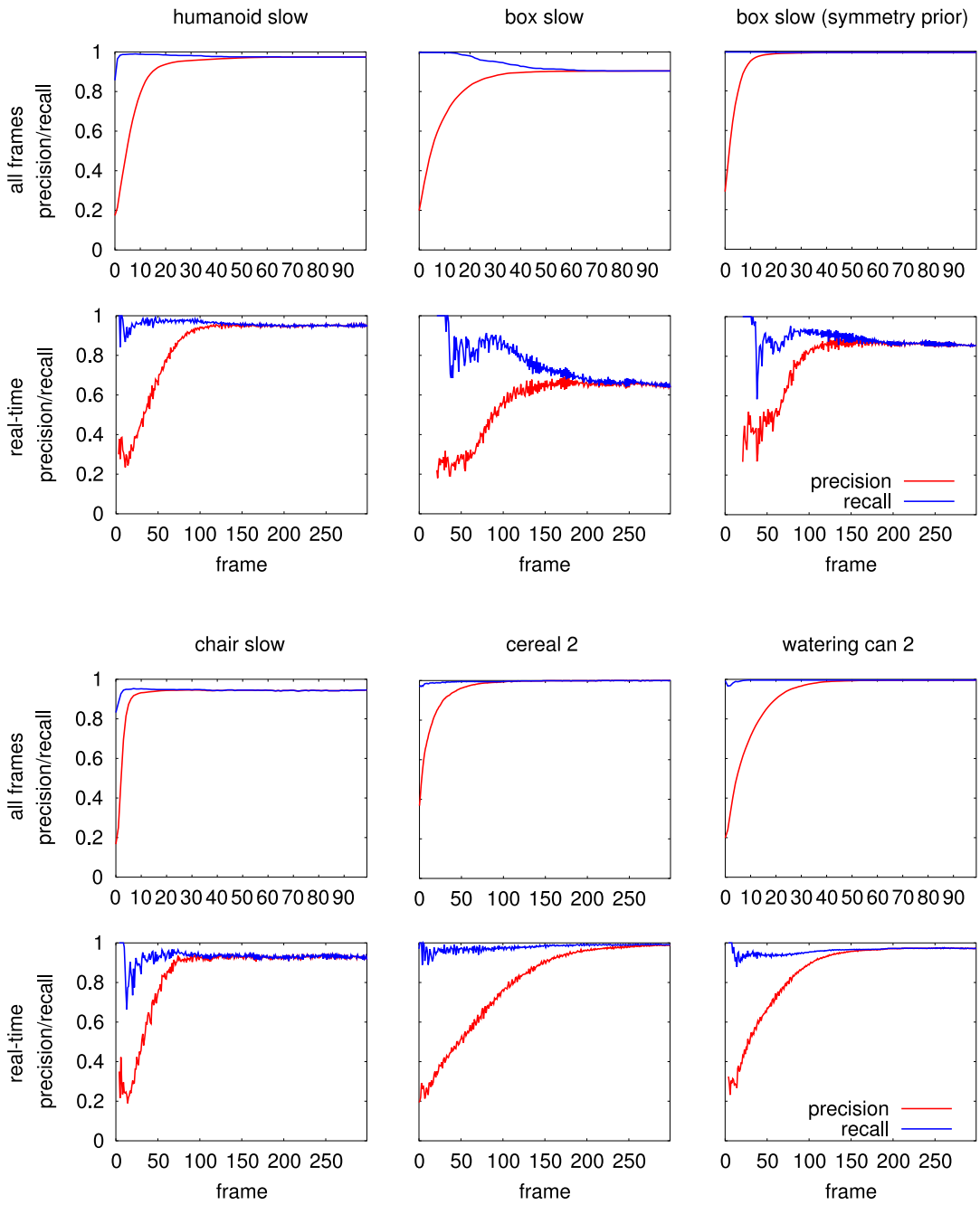

Figure 6.14.: Evolution of precision and recall during global localization on the object tracking sequences.

to the wrong symmetric pose hypothesis in some cases. This problem can be counteracted by making a symmetry-breaking prior available. To demonstrate this, we implemented this prior by only accepting detected poses within 144° of the true pose. Our approach then converges with high precision and recall rates

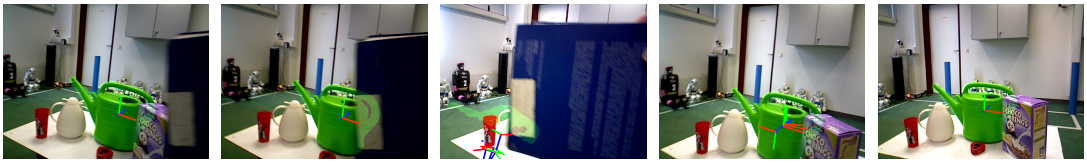

Figure 6.15.: Joint object detection, pose estimation, and tracking on the watering can 1 sequence. The track is lost due to a full occlusion (3rd image). Our approach detects this event and reinitializes the tracker through object detection (3rd and 4th image) until the filter is on track again (right).

close to values of 1 (see top-right Fig. 6.14). For some of the sequences a small increase in recall is visible over time. This is caused by pose hypotheses that require multiple tracking iterations to align well with the object.

Processing the sequences in real-time only slightly decreases the performance of our method on most sequences. Performance on the box slow sequence is more strongly affected, caused by the shape of the object as discussed above.

Fig. 6.15 demonstrates reinitialization on the watering can 1 sequence. As soon as large parts of the object are occluded, the tracker is reinitialized with our object detection method. Reinitialization is repeated until the tracker maintains a likely estimate as before the loss of the track.

6.4.7. Public Demonstration

The object tracking approach in Sec. 6.3.2 has been demonstrated publicy as a perception component of mobile manipulation robots at several RoboCup@Home competitions (Stückler et al., 2012; Stückler et al., 2012b, 2013, 2014)[2]. In the final of the RoboCup@Home competition 2011 in Istanbul, Turkey, service robot Cosero (team NimbRo@Home) showcased the cooperative carrying of a table with a human and the cooking of an omelett (Stückler et al., 2012; Stückler et al., 2012b) (see Fig. 6.16). For carrying the table, it tracked a MRSMap model of the table to perceive its lifting and lowering by the human through the estimated pitch rotation. In the omelett-cooking demonstration, the robot approached a cooking plate which it perceived through tracking a MRSMap model. After fetching the bottle of omelett mixture and opening it, Cosero moved in front of the cooking plate and poured the omelett-mixture into the pan on the plate. The demonstration was well received by the jury that consisted of representatives from science, industry, and media, and the executive committee of the RoboCup@Home league. It was an important contribution to winning.

[2]Videos of the performance of the robots can be found at: http://www.youtube.com/watch?v=nG0mJiODrYw, v=q041IvZ_FVU, v=tUhuHIbbEBA, and v=I1kN1bAeeB0

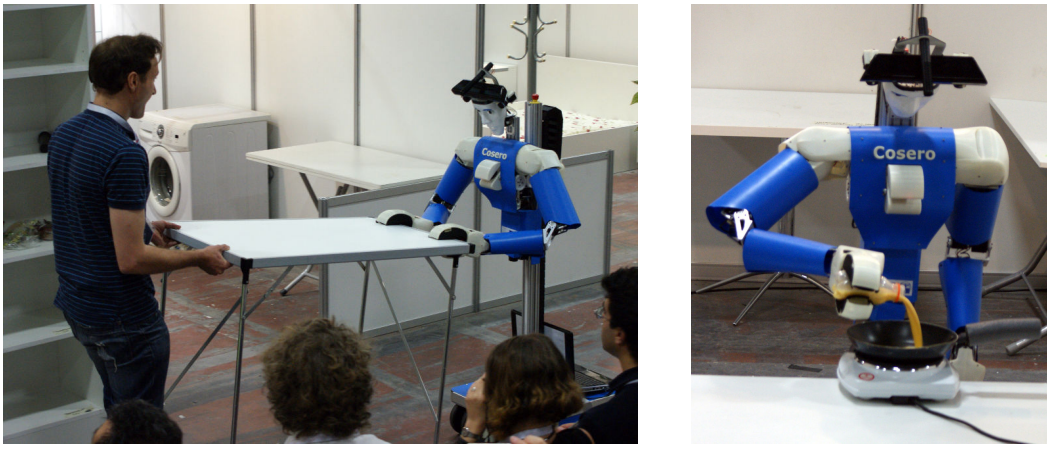

Figure 6.16.: Public demonstrations of object tracking for visual servoing in mobile manipulation tasks. Left: Cosero tracks the table for perceiving the lowering and lifting of the table while it cooperatively carries the table with a human at RoboCup 2011. Right: For switching the cooking plate on and pouring omelett-mixture into the pan, it tracked the pose of the cooking plate (RoboCup 2011).

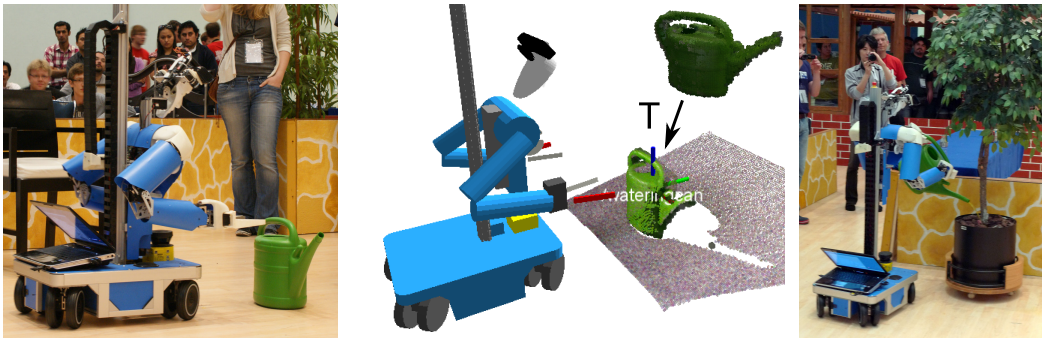

Figure 6.17.: Public demonstrations of object tracking for visual servoing in mobile manipulation tasks. To grasp the watering can, Cosero approaches the can towards a predefined relative pose using our object tracking method.

In 2012, Cosero pushed a chair to its designated location and watered a plant with a watering can (Stückler et al., 2013) (see Figs. 6.17 and 6.18). For both demonstrations, Cosero approached the objects using our tracking approach, and grasped the objects at predefined grasp poses. The demonstrations were important parts of the overall performance of the system that won the 2012 and 2013 RoboCup@Home German Open and the 2012 World Championship in Mexico.

For winning the 2013 RoboCup@Home competition in Eindhoven, Nether-

Figure 6.18.: Public demonstrations of object tracking for visual servoing in mobile manipulation tasks. Left/center: Cosero approaches a chair through tracking, grasps it, and pushes it to a desired location at RoboCup 2012 and 2013. Right: Cosero positions itself in front of a barbecue using our tracking approach at RoboCup 2013.

lands, object tracking also was a fundamental capability of Cosero (Stückler et al., 2014) (see Fig. 6.18). In the Demo Challenge it pushed a chair to its place. In the finals, it used a pair of tongs to pick and place sausages on a barbecue. It perceived and approached the barbecue through tracking.

6.5. Related Work

6.5.1. SLAM with RGB-D Sensors

Early work on SLAM in robotics has focused on acquiring 2D maps using range sensors such as laser scanners and sonars (e.g. (Grisetti et al., 2007)). Over the last decades, some approaches have been proposed that estimate the 6 DoF trajectory of a robot and a 3D map by means of 3D laser scan registration (Nuechter et al., 2005). In computer vision, many approaches to SfM are based on the extraction and matching of keypoints between images. Stereo vision is frequently used to directly obtain depth measurements for keypoints (Se et al., 2001; Nister et al., 2004; Konolige et al., 2010). Efficient RANSAC methods can then be applied to estimate the motion of the camera rig. This approach similarly applies if depth measurements are available from time-of-flight or structured-light RGB-D cameras (Droeschel et al., 2009; Huang et al., 2011).

MonoSLAM (Davison et al., 2007), based on Extended Kalman Filters, was one of the first methods that demonstrated feature-based online SLAM in real-time with a monocular camera. Klein and Murray (2007) proposed a real-time capable BA method within small workspaces. Current work on SfM in computer

vision also includes real-time dense surface reconstruction from monocular videos (Stuehmer et al., 2010; Newcombe et al., 2011b). Newcombe et al. (2011b) proposed DTAM, an impressive method for dense tracking and mapping of small workspaces that is real-time capable on GPU. It acquires dense models of key frames which could be globally aligned into a dense model of the scene using view-based dense SLAM methods such as our approach.

Closely related to our setting is KinectFusion, proposed by Newcombe et al. (2011a). They incrementally register depth images to a map that is aggregated from previous images and demonstrate remarkable performance for small workspaces. The approach is applied for augmented reality user interfaces and supports the tracking of the pose of objects and the camera in real-time. Since KinectFusion is implemented on GPU, it has—due to memory restrictions—stronger workspace limitation than CPU-based implementations like ours. KinectFusion represents the mapped surface with signed distance functions(SDFs) (Curless and Levoy, 1996), and estimates the pose of the camera with respect to the map by a variant of ICP. A depth image is generated from the SDF map and the current pose, and registered with the measured image. Bylow et al. (2013) propose a different approach for camera tracking and evaluate several weighting functions for the SDF representation. In order to scale to larger workspaces, KinectFusion has been extended using moving volumes (Whelan et al., 2012; Roth and Vona, 2012). Due to their incremental nature, these approaches still accumulate minor drift in the map estimate over time (Roth and Vona, 2012) when the camera is swept into previously unseen areas.

This effect can be corrected through loop-closing like in our view-based SLAM approach. For this, local submaps have to be built and eventually to be registered in a submap-based SLAM framework. Our framework supports a compact representation of local submaps and registers individual RGB-D images as well as entire local submaps that summarize many images. We detect loop closures and find a best alignment of key views by jointly optimizing spatial constraints between views. We determine the relative pose between views using our registration method and assess the uncertainty of the pose estimate. We also include spatial constraints between further key views that are not direct neighbors in temporal sequence, and verify the constraints by the quality of the image alignment. Very recently, Kerl et al. (2013) and Steinbruecker et al. (2013) followed similar ideas for key-view based SLAM with SDF representations.

Some approaches have been proposed that also learn maps from depth and RGB-D images in a trajectory optimization framework (May et al., 2009; Henry et al., 2012; Engelhard et al., 2011). May et al. (2009) match time-of-flight depth images using ICP and apply global relaxation over all images to build a consistent 3D map. Henry et al. (2012) extract textured surface patches, register them using the ICP algorithm to the model, and apply graph optimization to obtain an accurate map. Our approach provides shape-texture information in a compact representation that supports pose tracking from a wide range of

distances, since the model is represented at multiple scales. Endres et al. (2012) match RGB interest points between RGB-D images to obtain spatial relations for pose graph SLAM. Our registration method incorporates shape and texture seamlessly and is also applicable to textureless shapes.

The modeling of the geometry of objects from multiple views is a traditional research topic in robotics and computer graphics. A diverse set of applications exists for such explicit geometric map representations like, for instance, object recognition or manipulation planning.

One early work of Chen and Medioni (1992) registers several range images using an iterative least squares method. In order to acquire object models, the authors propose to take four to eight views onto the object. Each view is then registered to a map that is aggregated from the precedingly registered views. If the content and sequence of scans is chosen carefully to include significant overlap with the already acquired map, this procedure accumulates less error than pairwise registration of successive views. Weise et al. (2011) match surface patches between range images and align them globally to reconstruct 3D object models. Krainin et al. (2011) learn models of objects with an approach similar to (Henry et al., 2012). Schnabel et al. (2008) represent objects by graphs of geometric shape primitives. Our map representation includes shape and texture seamlessly and inherently supports tracking from a wide range of distances due to its multi-scale structure.

6.5.2. Object Detection and 6-DoF Pose Estimation

Methods for object detection and 6-DoF pose estimation can be distinguished into methods based on voting, RANSAC, and templates. The generalized Hough transform (Ballard, 1981) underlies voting-based methods. In early work, efficient implementations have been proposed using hash tables (Lamdan and Wolfson, 1988). These methods cast votes on the 6-DoF pose of 3D objects from tuples of interest points or edges in grayscale or range images and have been demonstrated for polyhedral objects. The SHOT (Tombari et al., 2010) descriptor and its color extension C-SHOT (Tombari et al., 2011) define a unique 3D reference frame for interest points in point clouds and RGB-D images. This allows for casting a vote on the object pose from only a single interest point match to an object model. However, extracting stable reference frames at interest points requires well sampled surfaces. Another recent approach by Drost et al. (2010) uses pairs of surfels that also define a unique reference frame. The surfel-pairs are described by the relation of the surface normals and the position difference between the surfel points. The descriptors extracted from the object model are hashed for efficient retrieval of surfel-pair matches with the scene. The approach has been extended to incorporate visibility-context (Kim and Medioni, 2011), contours (Choi et al., 2012), and RGB information (Choi and Christensen, 2012a). We also use color and extend the method with local

multi-resolution processing for improved run-time efficiency. Furthermore, we disambiguate pose hypotheses over time in a particle filter framework.

Similar concepts like in voting approaches can often be used within RANSAC. The seminal work by Lowe (2004) proposes an interest point detector and descriptor coined scale-invariant feature transform (SIFT), and applies it in efficient RANSAC for estimating object pose. The approach is robust to partial occlusions and view-point changes, but requires well textured objects. Scalable recognition of a multitude of object instances using SIFT is demonstrated in the MOPED framework (Martinez et al., 2010). Schnabel et al. (2008) estimate the pose of objects from a representation by graphs of geometric shape primitives using RANSAC. The pose estimate retrieved from matches of surfel-pairs between scene and model can also be used to create pose hypotheses in a RANSAC framework (Papazov et al., 2012).

Template-matching represents the object by templates that are correlated with the scene at all possible locations. To estimate the 6-DoF pose of the object, several templates are used for an object from different view points. Examples of early approaches match contour templates of 3D objects in RGB images using the Hausdorff (Olson and Huttenlocher, 1997) or Chamfer distance (Gavrila and Philomin, 1999). In the same line of research, Hinterstoisser et al. (2012) proposes LINE-2D and its RGB-D extension LINE-MOD. Their templates use only the most dominant gradients to represent the objects. If depth is available, they seamlessly integrate surface normals with 2D image gradient matching. The LINE algorithms are efficiently implemented using SIMD instructions of modern CPU architectures to operate in real-time. In contrast to the LINE method, our approach detects multi-view 3D models of objects, such that the management of multiple templates per object is not necessary. The VFH (Rusu et al., 2010) and CVFH (Aldoma et al., 2011) methods represent point cloud segments by histograms that quantify shape and view point. Lai et al. (2011) retrieve object category, object view, and continuous 6-DoF pose in a multi-stage classification approach for RGB-D image segments. Both approaches, however, require a presegmentation of the point cloud or RGB-D image, respectively.

6.5.3. Object Tracking

A vast set of object tracking methods in computer vision estimate the image location and bounding box of moving objects in RGB images. In the following, we focus on tracking approaches that estimate 6-DoF pose. Tracking approaches can be classified into tracking-by-optimization and tracking-by-detection.

6.5.3.1. Tracking-by-Optimization

In tracking-by-optimization, the pose estimate from the previous image is used to initialize pose estimation for the current image. Frequently, non-linear least

squares minimization is employed. The famous early method by Harris (1993) aligned 3D edge models of objects with the current model. Since then, edge-based methods have been revisited many times. The approaches by Drummond and Cipolla (2002) and Comport et al. (2004) track edge-models using iteratively re-weighted least squares (IRLS). Texture can also be combined with edge-based approaches (Vacchetti et al., 2004) to achieve tracking with textured as well as textureless objects. We propose tracking-by-optimization of dense 3D object models represented by MRSMaps. Our registration method leverages shape as well as texture cues for accurate tracking. The multi-resolution structure of MRSMaps allows for a wide range of distances to the tracked object. We further improve the robustness by embedding our registration method in a particle filtering framework.

Recursive Bayesian filtering is also popular in tracking-by-optimization. In contrast to Kalman filter approaches, particle filters can be applied for non-linear state-transition and observation models, and are not restricted to Gaussian noise models. The state density estimate is also not single mode, but could in principle represent any multi-modal distribution. This makes particle filters more robust for pose ambiguities, especially in the early phases of tracking when initialization provided multiple pose hypotheses, or if the measurements do not constrain poses to a single mode. Edge-models have been tracked within a particle filter by Klein and Murray (2006). To achieve real-time tracking, an implementation of the evaluation of the observation likelihood on GPU is required. Choi and Christensen (2012b) track multiple edge-based templates of an object with a particle filter. We obtain a highly accurate yet robust method that tracks dense 3D object models in real-time on a CPU. In the sampling step, we sample the particles from a proposal distribution that improves the state-transition model through alignment of the current image with the object.

6.5.3.2. Tracking-by-Detection

Tracking-by-detection applies a detection approach in each image to find the pose of the object. A key ingredient is the real-time detection and pose estimation from a single image without a prior guess. The LINE method as discussed above (Hinterstoisser et al., 2012) can hence be seen as one instance of tracking-by-detection methods. Lepetit and Fua (2006) develop an efficient interest point detector, descriptor, and matching algorithm based on RFs. This makes pose estimation from interest point matches efficient for real-time tracking.

In general, tracking-by-optimization incorporates a strong prior for determining the object's pose in the current frame, and yields better temporal coherence of the estimated trajectory. Tracking is often faster and more accurate than detection. On the other hand, strong priors can also be violated, for instance, at rapid object motion, occlusions, or if the object moves out of view.

6.5.4. Joint Object Detection, Pose Estimation, and Tracking

In order to overcome the limits of either tracking-by-optimization or tracking-by-detection, joint object detection and tracking aims at providing well and fast initialization of optimization-based tracking. Tracking is also made more robust when failures are detected and the track is reinitialized using a fast detection method. As pointed out by Lepetit and Fua (2005) and Uchiyama and Marchand (2012), joint object detection and tracking is an active research topic.

For initializing particle filter based tracking of edge templates, Choi and Christensen (2011) recognize object pose from SURF (Bay et al., 2006) interest point matches with a set of key views stored for the object. In (Choi and Christensen, 2012b), the approach has been extended to detect the edge-templates directly through template matching. Our approach tracks a dense multi-view 3D model of the object that represents the object more compactly than by a set of templates from discrete view points. Its run-time efficiency facilitates real-time operation on a CPU.

6.6. Summary

We developed means for modeling static indoor environments and rigid objects with RGB-D cameras. The models are acquired from RGB-D image sequences in which the camera moves through the scene or views the object from multiple view points. Our approach extracts key views represented as MRSMaps from the sequences. We use our registration method to keep track of the camera motion. It provides spatial constraints between the key views whose view poses are optimized through pose-graph SLAM. We not only consider the registration of key views in temporal sequence, but also find further constraints between key views. In each frame, a new constraint is examined and eventually added to the pose graph, if the key-view matching could be validated in terms of matching likelihood. The key views are aligned in one MRSMap to obtain a multi-view 3D model. Our SLAM approach supports real-time operation on a CPU which we demonstrate on sequences of the RGB-D benchmark dataset. It outperforms a sparse interest-point-based approach to RGB-D SLAM on several sequences. On sequences with long trajectory loops, SLAM is clearly superior to drift-prone incremental registration.

Once a MRSMap model of an object is available, we use the model for perceiving the object in live RGB-D images. We propose an object detection and 6-DoF pose estimation method that efficiently finds a coarse initial pose estimate at high recall rates. It is based on surfel-pair voting and utilizes multiple resolutions for efficiency. For each surfel, pairings with other surfels are established in a local neighborhood that depends on the surfel's resolution. Our experiments indicate, that the scale of these neighborhoods is also related to the shape of the

object and must be chosen appropriately.

We utilize our rigid registration method to track the 6-DoF pose of the objects in real-time. The registration is integrated in a particle filter for improved proposals. By this, we combine the robustness of maintaining multiple hypotheses during tracking with the accuracy of tracking-by-optimization. We evaluate accuracy and run-time of our tracking approach and demonstrate real-time tracking at high accuracy.

Tracking is initialized by our object detection method in a particle filter framework for joint object detection, pose estimation, and tracking. If tracking cannot be resumed, e.g., because of occlusions or extreme camera motions, we detect this event and reinitialize the tracker.

We demonstrated object tracking through registration at several public occasions at RoboCup@Home competitions. It was a building block for many mobile manipulation demonstrations. The robot performances were well received and have been important contributions to winning in 2011, 2012, and 2013.

In our current SLAM method, we search for loop-closures among key views with similar view poses. For very long trajectory loops with strong drift, this method may not be able to find a loop-closing constraint. A complementing approach could define similarity in key views through signatures that concisely describe the key view content.

The use of contours could further improve object detection and tracking. Efficient means would be required to extract expected contours from the model during the pose voting and registration processes. One possible approach is to project the model into the image plane for finding the contours. Another possibility is to consider those surfels as lying on contours, whose normals are perpendicular to the view direction.

A further line of research would be to transfer our approaches to object modeling, detection, and tracking with multi-resolution SDF representations such as the one proposed by Steinbruecker et al. (2013). Since SDFs are implicit surface representations, it can be costly to extract explicit geometry, e.g., in the form of a triangle mesh by algorithms such as fast marching cubes (Newman and Yi, 2006). This is less an issue for a multi-view SDF model of an object, as the explicit geometry could be extracted once in a preprocessing stage. The live camera image could be represented in another SDF which would require efficient methods to extract surface positions and normals. It is also possible to extract surface normals efficiently from the RGB-D image with different approaches, e.g., using integral images (Holz et al., 2011). However, representation of model and current image would then not share the same properties as with MRSMaps.

7. Non-Rigid Scene and Object Perception

In this chapter, we go beyond the assumption of rigid scenes and objects for perception and modeling. We first present an approach to learn dense models of the moving objects in a scene (see Fig. 7.1). At its core is our efficient rigid multi-body registration method (Ch. 4). We exploit motion as a fundamental grouping cue. Our approach not only models distinct parts in a scene. It also finds hierarchical part relations between the objects by observing them split and merge over time. This way, a robot may discover the hierarchical decomposition of the environment into objects in an unsupervised way.

Our second non-rigid perception technique matches object instances that vary in shape through continuous deformations. We transfer object manipulation skills defined for an example instance to new instances using deformable registration (Ch. 5).

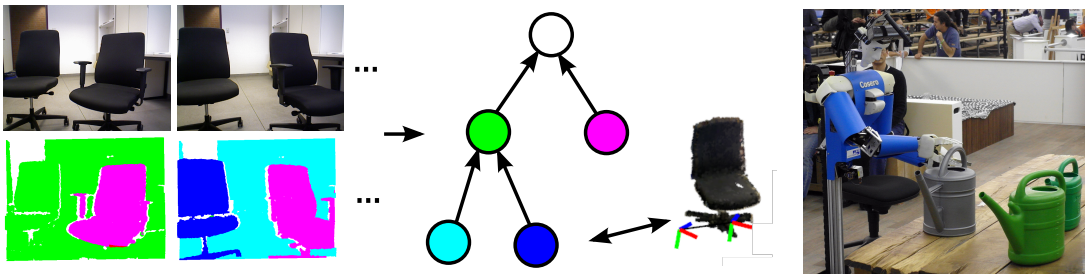

Figure 7.1.: Left: By integrating motion segmentation with SLAM, we discover objects and hierarchical part relations. Right: We transfer object manipulation skills through shape matching.

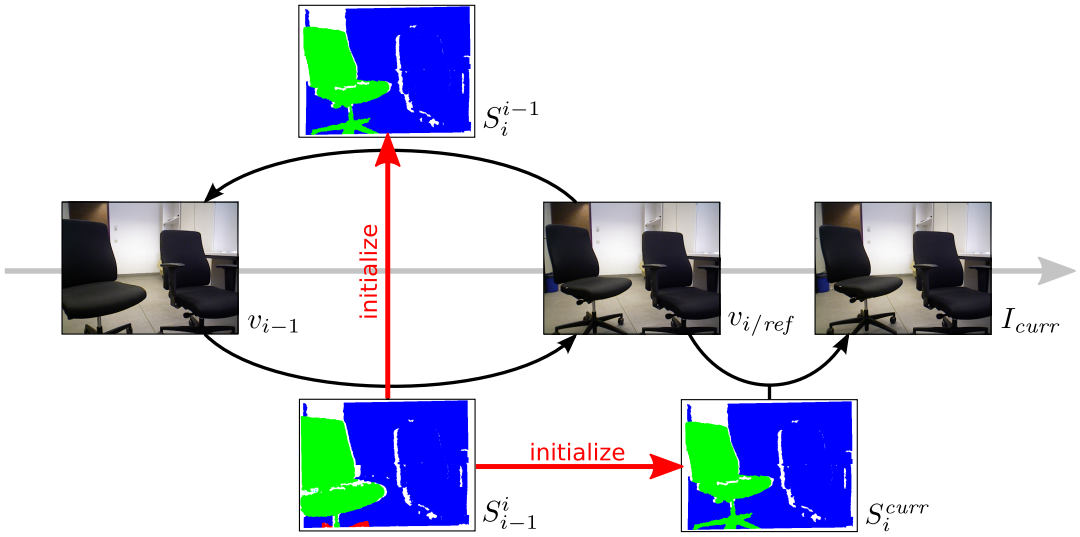

Figure 7.2.: We sequentially track the segmentation of the current image I_{curr} towards a reference key view v_{ref}. After sufficient motion, we include a new key view v_i for the current image. Its segmentations S_i^{i-1} towards its previous reference key view v_{i-1} and S_i^{curr} towards the current image are initialized from the tracked segmentation S_{i-1}^i.

7.1. Discovery and Dense Modeling of Object Hierarchies in Dynamic Scenes

Many SLAM approaches, as also our method in Sec. 6.2, make the assumption that the observed scene remains static during the mapping process. In this section, we release this assumption: We develop SLAM in dynamic environments in which we assume the moving parts to be rigid. We exploit motion as a fundamental grouping cue that allows an agent to learn about the decomposition of scenes into objects and parts.

We extend our key-view-based SLAM approach (Sec. 6.2) towards simultaneous motion segmentation, localization, and mapping (SMOSLAM). We still extract key views from RGB-D video. Key views are now related using rigid multi-body registration (Ch. 4) to discover the distinct moving parts between pairs of key views. The found segments are attributed to individual objects. Our approach examines the merging and splitting of segments, from which it infers part and equivalence relations of objects. Concurrently, each object maintains and optimizes an individual view pose graph for its segments. We overlay the segments from their estimated view poses into dense object models.

7.1.1. Discovery of Objects and Relations in RGB-D Video

We process RGB-D video sequentially. In order to localize the sensor with respect to the moving parts in the scene, we register the current RGB-D image towards a reference key view (see Fig. 7.2). We apply our rigid multi-body registration method to segment the reference key view with respect to the current image, and concurrently estimate the relative motion between the segments.

7.1.1.1. Key Views

The initial reference key view is set to the first image. We track the segmentation S_{ref}^{curr} of a reference key view v_{ref} towards the current image I_{curr} using our online EM-approach (Sec. 4.2.5). After sufficient motion of one of the segments, we create a new key view v_i from the current image and make it the new reference key view. We also create a new key view, if the motion of the segments – after a significant move – ceased. This event is detected from the magnitude in rotational and translational velocity which is determined from the motion estimates for a few most recent images.

7.1.1.2. Sequential Key View Segmentation

As illustrated in Fig. 7.2, we establish several segmentations between key views. When a new key view v_i is included, we already have a motion segmentation S_{i-1}^{i} between the new key view and its reference key view v_{i-1} available through tracking. As will become apparent shortly, we also require the segmentation S_{i}^{i-1} in the opposite direction between the key views for the establishment of object relations. We initialize this backward segmentation from the result of the tracked forward segmentation. Few EM iterations suffice to let the segmentation converge from this initialization. The new key view becomes the reference for tracking towards the current image in the sequence. Its segmentation S_{i}^{curr} is also initialized with the result of the previous tracking segmentation S_{i-1}^{i}.

For the initialization, segmentation transfer proceeds in two ways. If source and target segmentation share the same segmented image, we simply set the segmentation of the target equal to the source. If the segmentations are opposite, i.e., source and target segment the same images in opposite directions, we transfer the labeling: Each labeled image site in the segmented image of the source is associated with a site in the segmented image of the target. It propagates its label to its associated image site. To compensate for the different local multi-resolution structures of both images, we further distribute this labeling to unlabeled successors in the octree. We also set the motion estimates of the initialized segments to the inverses of their counterparts.

As we segment between images, the observed scene content will not completely overlap due to the limited field-of-view of the sensor and due to occlusions. In

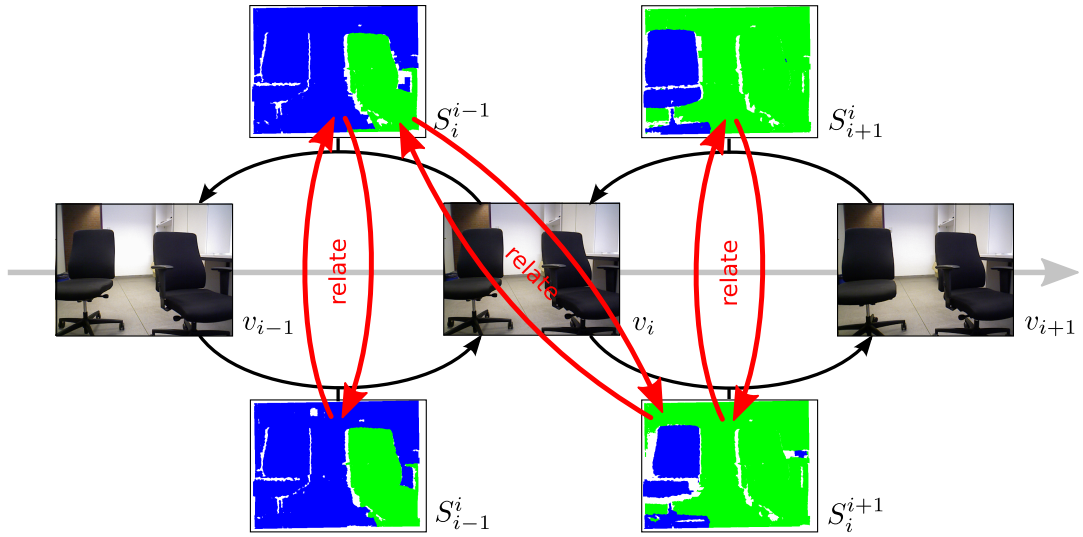

Figure 7.3.: We relate motion segmentations between pairs of key views. The related pairs either segment key views in opposite directions (e.g., S_{i-1}^i and S_i^{i-1}), or segment the same image (e.g., S_i^{i-1} and S_i^{i+1}).

Sec. 4.2.6.1, we propose to handle this by memorizing the observation likelihood of image sites that would transform beyond the field-of-view or that are occluded. This information is only available through tracking. We thus also transfer memorized observations between the segmentations.

7.1.1.3. Identifying Relations between Segments and Objects

Our goal is to assign motion segments to objects for dense modeling, and to deduce a decomposition of the objects into parts by observing the objects split and merge. Each motion segmentation contains a set of segments for which we create objects. We relate segments between different motion segmentations to

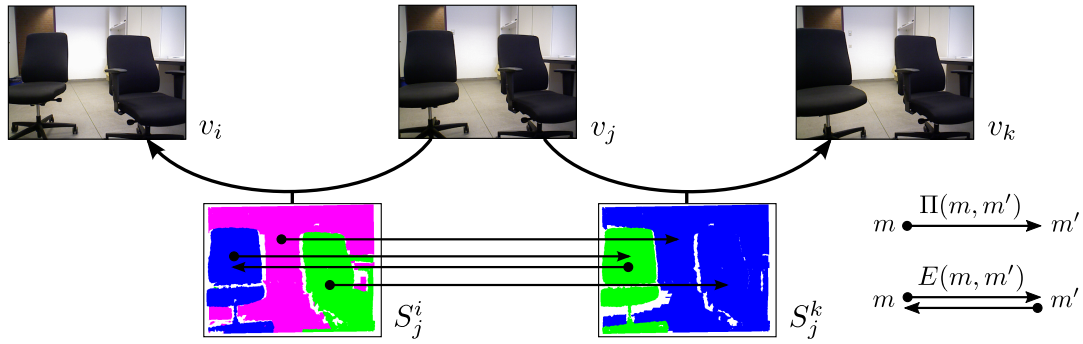

Figure 7.4.: We determine part $\Pi(m,m')$ and equivalence relations $E(m,m')$ between segments m,m' from their overlap.

determine, if the segments are either part of one another, or if they equivalently observe the same object. These segment relations in turn provide knowledge about part and equivalence relations between objects.

Relations between Segments: As a first step, we find part and equivalence relations between segments of different segmentations. We relate segments by their overlap in two ways. First, both segmentations $S := S_a^b$, $S' := S_a^c$ may share the same segmented image. We denote such a pair of segmentations as adjacent. We determine the overlap

$$\rho(m_{S,k}, m_{S',k'}) := \frac{\left|\left\{i \in \{1,\dots,N\} : y_{S,i} = k \wedge y_{S',i} = k'\right\}\right|}{\left|\left\{i \in \{1,\dots,N\} : y_{S,i} = k\right\}\right|} \tag{7.1}$$

of source segments $m_{S,k} \in \mathcal{M}_S$ with target segments $m_{S',k'} \in \mathcal{M}_{S'}$ by directly counting matching labelings of image sites in the segmented images. We denote the labeling of image sites $i \in \{1,\dots,N\}$ in source and target segmentation as $y_{S,i} \in Y_S$ and $y_{S',i} \in Y_{S'}$, respectively. The overlap measure is directional and quantifies the degree of inclusion of source segments in target segments. Hence, we relate segmentations in both directions.

Opposite segmentations S_a^b, S_b^a between pairs of images can also be evaluated for overlap. To count matches, the label of each image site in the segmented image of the source is compared with the label of its associated site in the target segmentation.

We take care of occlusions and outliers and discard them for the overlap measure. Occlusions occur at image sites that would move behind another image site in the connected image and would hence not be visible. The segmentation at such sites is not well supported by observations and governed by context.

We process RGB-D video sequentially and measure the overlap of segments between adjacent and opposite segmentations (see Fig. 7.3). Adjacent segmentations S_i^{i-1}, S_i^{i+1} connect consecutive key views v_{i-1}, v_i, and v_{i+1} through a center key view v_i. The relation of opposite segmentations S_i^{i+1}, S_{i+1}^i connects consecutive adjacent relations throughout the key view sequence.

We estimate part relations between segments from their overlap (see Fig. 7.4). A segment m is part of segment m', if it overlaps m' by a specific amount χ_ρ, i.e.,

$$F_0 : \ \rho(m, m') \geq \chi_\rho \implies \Pi(m, m'). \tag{7.2}$$

Two segments m and m' observe a physical entity equivalently, if they are part of each other,

$$F_1 : \ \forall m \forall m' : \ \Pi(m, m') \wedge \Pi(m', m) \iff E(m, m'). \tag{7.3}$$

Obviously, $E(m', m)$ also holds by symmetry.

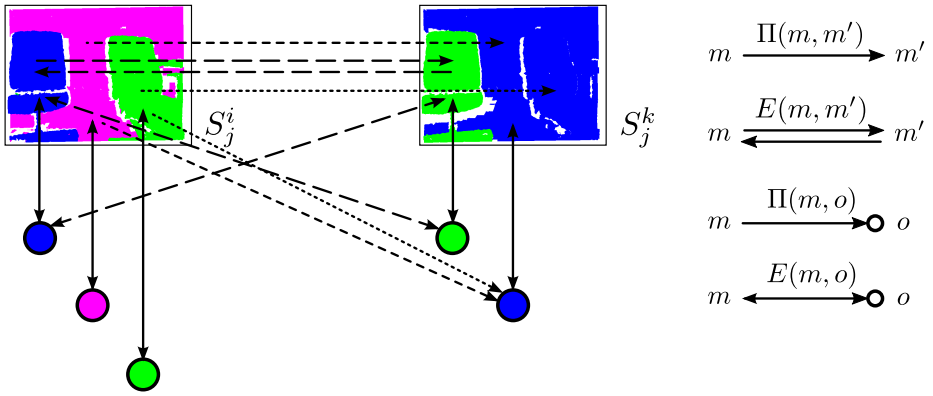

Figure 7.5.: Each segment m is assigned an object o, which the segment is part of ($\Pi(m,o)$) and equivalent to ($E(m,o)$). Segment relations induce further part and equivalence relations to objects. Induced segment-object relations (dashed) and their origin relations between segments are depicted by common dash styles.

When a new segmentation is established, we find all new unrelated pairs of segmentations and determine part and equivalence relations. We establish new part relations between segments, then examine the new part relation for further equivalence relations between segments.

Relations between Segments and Objects: Each segment m creates its own object $o = c(m) \in \mathcal{O}$. A segment is part of and equivalent to its object,

$$F_2: \ \forall m \forall o: \ o = c(m) \implies \Pi(m,o) \wedge E(m,o). \tag{7.4}$$

Segments m are also part of an object o, if they are part of another segment m that is itself part of o:

$$F_3: \ \forall m \forall m' \forall o: \ \Pi(m,m') \wedge \Pi(m',o) \implies \Pi(m,o). \tag{7.5}$$

Analogeously, segment m is equivalent to an object o through equivalence with a segment m that is equivalent to o:

$$F_4: \ \forall m \forall m' \forall o: \ E(m,m') \wedge E(m',o) \implies E(m,o). \tag{7.6}$$

Fig. 7.5 illustrates how segment-object relations are induced by segment-segment relations.

When new objects o or relations between segments m and m' are added, we examine if they induce novel segment-object relations by inspecting other segment-object relations that involve segments m, m', or object o. Furthermore, if a relation between segment m and object o is included, it may induce additional segment-object relations which are searched for by inspecting part and equivalence relations between other segments and m.

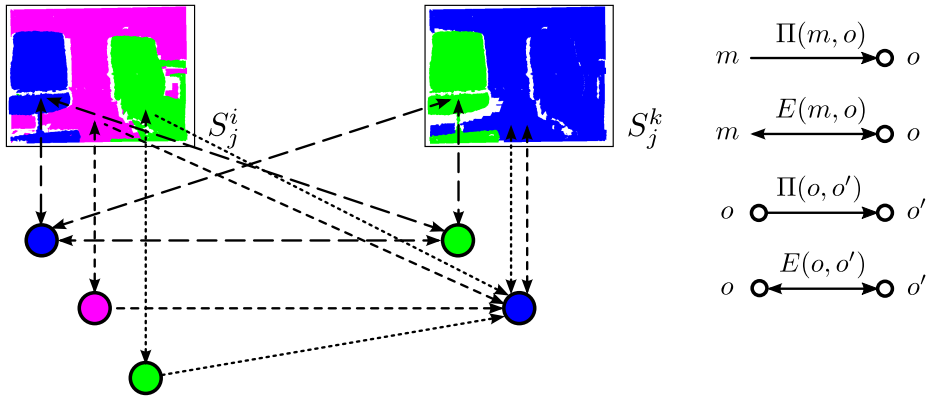

Figure 7.6.: We infer part and equivalence relations between objects from segment-object relations. Induced segment-object relations and their origin relations are depicted by common dash styles.

Relations between Objects: From relations between segments and objects, we can further conclude part and equivalence relations between objects (see Fig. 7.6). If a segment m is part of two objects o and o', but only is equivalent to o', then o must be part of o', i.e.,

$$F_5: \ \forall m \forall o \forall o': \ \Pi(m,o) \wedge \neg E(m,o) \wedge E(m,o') \implies \Pi(o,o'). \tag{7.7}$$

If the segment is in an equivalence relation to both objects, the objects are representing the same physical entity, i.e.,

$$F_6: \ \forall m \forall o \forall o': \ E(m,o) \wedge E(m,o') \implies E(o,o'). \tag{7.8}$$

By symmetry, also $E(o',o)$ holds.

The procedure to establish relations between objects is to consider only novel relations between segments m and objects o or o'. For a new part relation between segment m and object o, we search for equivalence relations of segment m with other objects o'. If an equivalence relation between segment m and object o is induced, we find all other objects o' which m is equivalent to in order to include equivalence of o and o'.

Object Pruning: Including objects for each segment in every motion segmentation generates many redundant, equivalent objects. We spare computation time and merge objects that we infer to be equivalent. As our inference process generates relations between segments and other objects equivalently for both objects, we can simply discard the newer object and all its relations with segments.

7.1.1.4. Probabilistic Reasoning on Segment and Object Relations

To cope better with imperfect segmentations and uncertain overlap decisions, we perform probabilistic reasoning about segment and object relations. The

relations identified in Sec. 7.1.1.3 are formulated in first-order logic and form a knowledge base KB. We use Markov logic networks(MLNs) to transform the set of hard constraints in first-order logic into a probabilistic interpretation. See Richardson and Domingos (2006) for an introduction to first-order logic and MLNs.

In terminology of first-order logic, each existing segment and object is a constant in a finite set C. Generically, we refer to segments and objects through variables m and o. Part and equivalence relations are expressed by predicates r on variables and constants. The function $o = C(m)$ assigns each object to its creating segment. Eqs. (7.2),(7.3),(7.4)(7.5),(7.6),(7.7), and (7.8) define a set of formulae $\mathcal{F} = \{F_i\}_{i=0}^{6}$ over predicates and functions on segments and objects. Each predicate and formula is grounded by inserting existing segments and objects for the variables. A possible world assigns a truth value to each ground atom. Eq. (7.2) is a special case of grounded formula that we will interpret as uncertain evidence of a grounded predicate that expresses a part-relation between segments. As we are only interested in beliefs on ground predicates within our KB, inference is feasible by only considering those groundings of formulae that involve segments, objects, and predicates that are identified through the process in Sec. 7.1.1.3.

We define the MLN L on the formulae $\mathcal{F}$. Each fomula $F \in \mathcal{F}$ is associated a weight w_F that expresses the importance of the formula. With the existing segments and objects C, the MLN determines a MN $M_{L,C}$ (Sec. 4.1.2.1). Each ground predicate $\check{r}$ in our KB is assigned a binary random variable x_r whose value is 1 if it is true, and 0 otherwise. For each grounded formula $\check{F}$, the MN contains a potential $\varphi_F(x_{r_1}, \ldots, x_{r_R})$ on the R ground predicates in $\check{F}$. Formulaes of types F_1 to F_6 have a value of 1 if the formula is true, and 0 otherwise. For formula F_0 we express uncertainty through the degree of overlap: The relation $\Pi(m_{s,k}, m_{t,l})$ is true with probability

$$p\left(\Pi(m_{s,k}, m_{t,l})\right) = \begin{cases} \frac{\rho(m_{s,k}, m_{t,l}) - \rho_0}{1 - \rho_0} & \text{if } \rho(m_{s,k}, m_{t,l}) \geq \rho_0 \\ 0 & \text{otherwise} \end{cases} \tag{7.9}$$

in dependency on the overlap of the segments with a zero probability threshold ρ_0. This yields the joint probability

$$p(x) = \frac{1}{Z} \prod_{\check{F} \in KB} \varphi_F(x_{r_1}, \ldots, x_{r_R})^{w_F} \tag{7.10}$$

of possible worlds x in our KB.

We perform inference on this MN using sum-product LBP (Sec. 4.1.2.4). Relations are regarded as valid, if their belief is above a threshold.

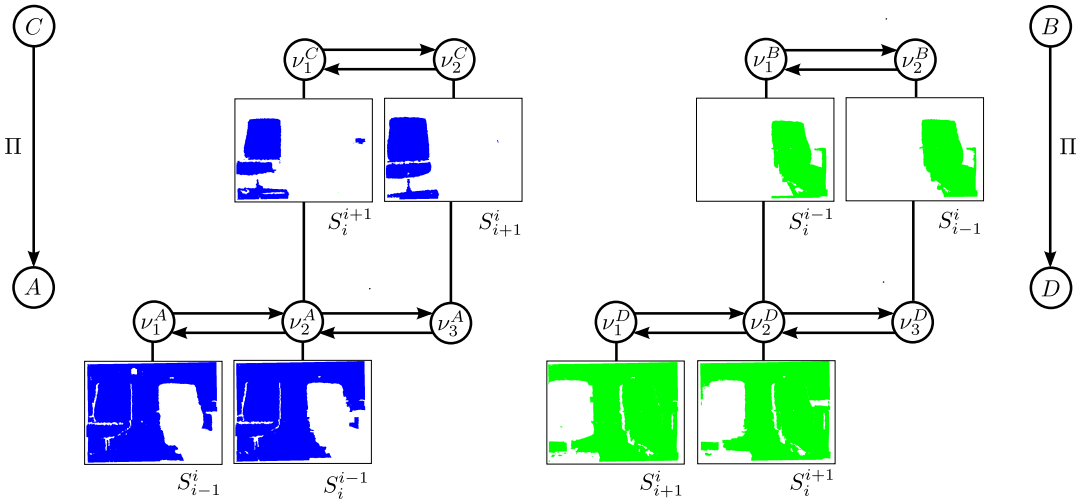

Figure 7.7.: For each object o, we maintain a graph of view poses ν_k^o of the segments that are part of the object. The segmentation S_i^j of key views i and j provides relative motion estimates between segments, which we include as spatial constraints between the segment view poses.

7.1.2. Simultaneous Localization and Mapping of Singularized Objects

Segments are dense RGB-D measurements of objects that move distinguishable between two key views. With our reasoning approach (Sec. 7.1.1), we link the segments of different key-view segmentations and identify which objects each segment is part of. The segments view the objects from specific view poses. The motion estimates of the segments are relative constraints between the segment view poses. We estimate the view poses by pose-graph optimization analogously to our SLAM approach in Sec. 6.2.

Each object o is created by a segment m_o whose view pose $\nu(m_o) \in \mathcal{V}_o$ defines the reference frame of the object SLAM graph. To establish a view pose graph for all segments that belong to an object, we examine valid part-relations $\Pi(m', o)$ of segments m' with the object. These relations are induced by formulae of type F_3 (Eq. (7.5)) which link segment m' to object o through other segments m.

If both segments share the same segmented key view, i.e., the segmentations are adjacent, no motion could occur between the segments. Hence, the segments reference the same view pose $\nu = \nu(m) = \nu(m') \in \mathcal{V}$.

Otherwise, the motion estimate θ_m of segment m is a relative view pose observation $x_m^{m'} := \theta_m$ between the segmented key views. We include the observation with its uncertainty estimate $\Sigma(x_m^{m'})$ as spatial constraint $e_{\nu(m),\nu(m')} \in \mathcal{E}_o$ between the segment view poses in the pose graph $G_o = (\mathcal{V}_o, \mathcal{E}_o)$ of object o. We

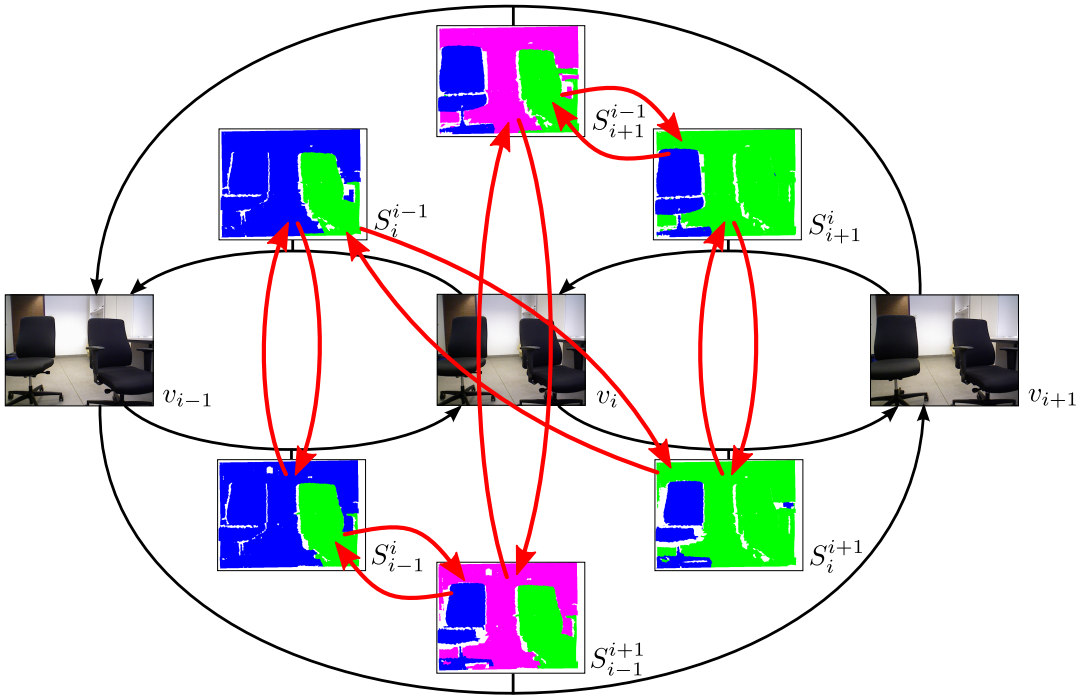

Figure 7.8.: We segment motion between pairs of key views that are not in direct temporal sequence. By this, we discover further relations between segments and objects, and include additional spatial constraints in the object SLAM graphs (black arrows: segmented key view pairs; red arrows: related segmentations).

obtain the uncertainty estimate of the segment with our approach in Sec. 3.2.2.3.

Multiple segments m_i within one segmentation can be part of the same object. In this case, we also maintain just a single node $\forall i : \nu = \nu(m_i)$ for the segments. If the segments have pose observations x_i towards the same node ν', we merge the observations into a single constraint $e_{\nu,\nu'}$ with pose covariance $\Sigma = \left(\sum_i \Sigma_i^{-1}\right)^{-1}$ and mean $x = \Sigma\left(\sum_i \Sigma_i^{-1} x_i\right)$, which follows from the product of the normal distributed pose observations.

As in Sec. 6.2, the resulting object pose graph is optimized using the LM method within the g^{2}o framework (Kuemmerle et al., 2011).

7.1.3. Out-Of-Sequence Relations

Our approach discovers objects and relations by the motion that is observable between the generated key views. So far, we only considered the segmentation and relation of key views that are direct neighbors in the temporal key view sequence. Key views can be segmented yet from larger relative pose distances. This increases the capability of our algorithm to observe objects split and merge,

and adds new spatial constraints into the object SLAM graphs.

We search for segmentations of key views v_i, v_j for which the segmentation S_i^j has not been incorporated yet. To only consider reasonable matches between key views with large image overlap, we also require the relative motion between the segments to be sufficiently small. We retrieve these relative motions from view pose estimates maintained within the object SLAM graphs. As our new segmentation is directional from key view v_i to v_j, we determine for each segment m_i in key view v_i, if it is connected with any segment in key view v_j by a view pose node and if the relative pose estimate between the nodes for the segments is small in at least one of the object SLAM graphs.

For temporally distant image pairs, our segmentation approach requires an initial guess. We determine this guess from established segmentations with a common intermediate key view v_k. Our algorithm first identifies if there is a k such that, in the object SLAM graphs, the segments of S_i^k are connected with segments in S_j^k. If such a k exists, we propagate the segmentations in the order $S_j^k \to S_k^j \to S_k^i \to S_i^k$ to arrive at a guess $\widetilde{S}_i^j$. The segmentation S_i^j is found within a few EM iterations from this guess.

In order to propagate segmentations from S to S', we need to distinguish between adjacent and opposite segmentation pairs. Adjacent pairs $S := S_k^j$, $S' := S_k^i$ segment the same key view, while opposite pairs are of the form $S := S_j^k$, $S' := S_k^j$. In both cases, we relate segments m_S with segments $m_{S'}$ through image overlap as in Sec. 7.1.1.3. If a segment m_S is only part of a segment in $m_{S'}$ but not equivalent with it, we split segment $m_{S'}$ into two segments: One that equivalently overlaps with m_S and a new segment $m'_{S'}$ that explains the remaining part of $m_{S'}$. In the opposite case, the motion estimates of segments $\widetilde{m}_{S'}$ are set to the inverse of the estimates of segments m_S that have best overlap from S' to S. For adjacent pairs, we set the motion estimate $\widetilde{\theta}_{S',m} = \theta_{S',m} \oplus \left(\theta_{S,m}\right)^{-1}$ to the combined estimate of $m_{S'}$ and the segment m_S with best overlap.

The new segmentation S_i^j is related with all existing adjacent and opposite segmentations to find new segment relations (Sec. 7.1.1.3). We add induced relations and formulae (Sec. 7.1.1.3) to our knowledge base KB, and update the object SLAM graphs with the new segments and motion estimates (Sec. 7.1.2).

7.1.4. Dense Models of Singularized Objects

We extract dense object models from the object SLAM graphs. First, we obtain probabilistic interpretations of each segment by performing mean-field iterations for the CRF segmentation (Sec. 4.2.3) until convergence. The soft-classified segments are fused in the common reference frame of the object pose graph using a log-odds filter. For segments that are only valid part of an object but not in an equivalence relation, we only consider positive observations. Images

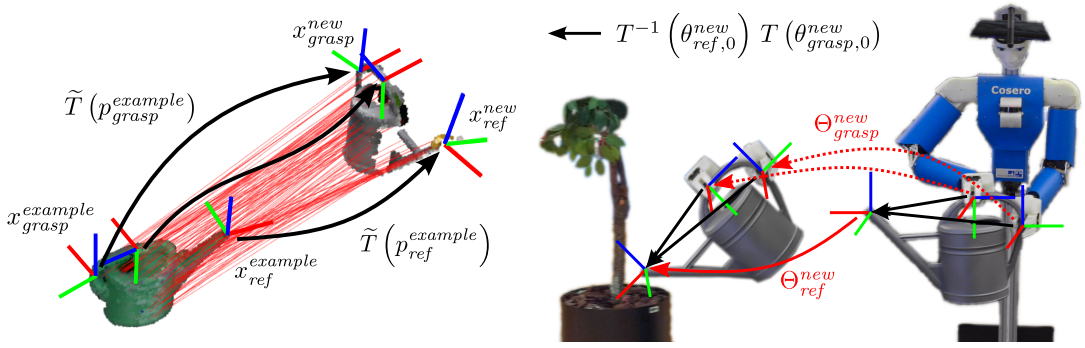

Figure 7.9.: We transfer grasp poses, tool end-effector frames (left, displayed as coordinate frames), and motion trajectories (right, red arrows) between different instances of the same class of objects. The local pose transformation at the grasps between the object instances is estimated from the displacement field between the object shapes.

with equivalent segments also provide negative observations at image sites that are most probable to belong to a different segment. We integrate this negative evidence to improve the segmentation of the object.

7.2. Shape Matching for Object Manipulation Skill Transfer

Objects with the same function often share a common topology of functional parts such as handles and tool-tips (Tenorth et al., 2013). We propose to interpret shape correspondences as correspondences between the functional parts. We utilize these correspondences for object manipulation skill transfer.

In many object manipulation scenarios, controllers can be specified through grasp poses and 6-DoF trajectories relative to the functional parts of an object. With known correspondences of the functional parts, these grasps and motions are transferable to other object instances.

In Ch. 5, we propose an efficient deformable registration method that provides a dense displacement field between object shapes observed in RGB-D images. From the displacements, local transformations can be estimated between points on the object surfaces. We apply these local transformations to transfer grasps and motion trajectories between the objects, which are defined relative to the objects and their functional parts (illustrated in Fig. 7.9).

7.2.1. Grasp Transfer

We define a grasp as a 6-DoF end-effector pose $x_{grasp}^{example}$ relative to a reference frame of the example object. When a new instance with different shape is given, we estimate a displacement field between both shapes using deformable registration (Ch. 5). The grasp pose is transformed onto the new object to a pose x_{grasp}^{new} using the displacement field.

For the registration, we assume that the new object instance is segmented from its surrounding, e.g., using a plane segmentation approach (Holz et al., 2011). We represent the RGB-D image segment in a MRSMap. The orientation of the new instance needs to coarsely match with the example object. As an intialization step for the registration, the MRSMaps are brought into coarse pose alignment by moving their spatial means onto each other. Deformable registration between the MRSMaps then yields a displacement field v.

Since the new object is only partially visible, we register the smaller map of the new object onto the multi-view model of the example object. I.e., in the formalism of our deformable registration, the new object is the model and the example object the scene. The method in Sec. 5.3.2 is the appropriate choice to estimate the local deformation $\widetilde{T}\left(p_{grasp}^{example}\right)$ from the example object to the new object, where $p_{grasp}^{example}$ is the position of the grasp on the example object. The grasp pose on the new object is

$$T\left(x_{grasp}^{new}\right) = \widetilde{T}\left(p_{grasp}^{example}\right)\ T\left(x_{grasp}^{example}\right). \tag{7.11}$$

7.2.2. Motion Transfer

We express the usage of an object through the motion of a reference frame of the object. If the object is a tool that affects another object, it is often useful to define this reference frame at the tool's end-effector. We transfer the reference frame to a new object through deformable registration, and execute the same motion with this frame as for the example object. The reference frame is a pose $x_{ref}^{example}$ on the example object. Its counterpart x_{ref}^{new} on the new object is found through local deformation

$$T\left(x_{ref}^{new}\right) = \widetilde{T}\left(p_{ref}^{example}\right)\ T\left(x_{ref}^{example}\right) \tag{7.12}$$

The local transformation $\widetilde{T}\left(p_{ref}^{example}\right)$ is determined at the reference frame's example position $p_{ref}^{example}$.

The example motion of the reference frame is given as a trajectory $\Theta_{ref}^{example} = \left(\theta_{ref,0}^{example}, \ldots, \theta_{ref,T}^{example}\right)$ which typically starts at the current pose of the reference frame. If the object is used as a tool on an affected object, the end of the trajectory is constrained through the affected object. The motion can be parametrized in dependence on the pose of the affected object.

We make the trajectory relative to the start pose, i.e.,

$$\widehat{\Theta}_{ref}^{example} = \left(\widehat{\theta}_{ref,0}^{example}, \ldots, \widehat{\theta}_{ref,T}^{example}\right), \tag{7.13}$$

with $T\left(\widehat{\theta}_{ref,t}^{example}\right) = T^{-1}\left(\theta_{ref,0}^{example}\right) T\left(\theta_{ref,t}^{example}\right)$. The corresponding trajectory for the new object is then $\Theta_{ref}^{new} = \left(\theta_{ref,t}^{new}\right)_{t=0}^{T}$ where

$$T\left(\theta_{ref,t}^{new}\right) = T\left(\theta_{ref,0}^{new}\right) T\left(\widehat{\theta}_{ref,t}^{example}\right). \tag{7.14}$$

The start pose of the reference frame for the new object can be found from the local deformation from example to new object,

$$T\left(\theta_{ref,0}^{new}\right) = \widetilde{T}\left(p_{ref}^{example}\right)\ T\left(\theta_{ref,0}^{example}\right). \tag{7.15}$$

We choose the start pose of the trajectory for the transformation, as it is close to the object surface. By this, we intend that the displacement field estimate at the reference pose is well supported by data evidence.

If multiple motions of a rigid object are concatenated in a sequence, it is not necessary to perceive the deformation at each start of a motion. E.g., if we assume the grasps to be fixed during all motions, we can initially store the reference frames used relative to the grasp poses on the new object, and recover the reference frames from the current grasp poses at the beginning of each motion.

The robot does not directly move the reference frame, but generates object motion with its end-effectors that act on the object through the grasp poses. To generate the desired reference frame motion, the robot end-effectors that grasp the object are thus moving on a trajectory $\Theta_{grasp}^{new} = \left(\theta_{grasp,t}^{new}\right)_{t=0}^{T}$ that is constrained relative to the reference frame. We assume rigidness of the object instances such that the relative pose of the grasp towards the reference frame remains constant, i.e., for all t and t',

$$T^{-1}\left(\theta_{ref,t}^{new}\right)\ T\left(\theta_{grasp,t}^{new}\right) = T^{-1}\left(\theta_{ref,t'}^{new}\right)\ T\left(\theta_{grasp,t'}^{new}\right). \tag{7.16}$$

This allows for writing

$$T\left(\theta_{grasp,t}^{new}\right) = T\left(\theta_{ref,0}^{new}\right) T\left(\widehat{\theta}_{ref,t}^{example}\right)\ T^{-1}\left(\theta_{ref,0}^{new}\right)\ T\left(\theta_{grasp,0}^{new}\right). \tag{7.17}$$

Also the start pose of the grasp is given through the local deformation from example to new object,

$$T\left(\theta_{grasp,0}^{new}\right) = \widetilde{T}\left(p_{grasp}^{example}\right)\ T\left(\theta_{grasp,0}^{example}\right). \tag{7.18}$$

Clearly, our approach assumes the object instances themselves to be rigid, and cannot consider dynamics or complex causalities involved in the execution of a task. Releasing these restrictions is a potential path for future research.

7.3. Experiments

7.3.1. Hierarchical Object Discovery and Dense Modelling

We demonstrate and evaluate our approach to hierarchical object discovery and dense modelling on two RGB-D sequences. The first sequence contains two independently moving chairs in a static camera setup. The second sequence displays a container with drawers in which the camera is moving througout the sequence. The container is moved with respect to the static scene background, before one of the drawers is pulled open.

Both sequences have been recorded with an Asus Xtion Pro Live RGB-D camera. Ground truth motion could not be captured with a motion capture system, as the optical markers would have been occluded during the recordings. For reference, the accuracy of our motion segmentation and SLAM methods have been assessed in chapters 4 and 6. For the MRSMaps we use a distance dependency factor of $\lambda_\rho = 0.014$ at a maximum resolution of $0.025\,m$. All formulae but F_0 have been weighted by $w_F = 1$. For F_0 we used a weight of 10 to increase the influence of these evidence relations. The lower bound for the overlap was chosen as $\rho_0 = 0.5$. Finally, we accept relations as valid, if their belief exceeds a threshold of 0.8.

7.3.1.1. Chairs Sequence

Fig. 7.10 shows the sequence of the 14 key views extracted in the chairs sequence. In addition, we show the 34 segmentations made between pairs of key views. It can be seen, that many out-of-sequence segmentations between key view pairs are established. They occur most frequently, where one chair stops moving while the other chair is pushed.

All valid relations between segments and objects found by our approach are shown in Fig. 7.11. At the end of the sequence, the MLN consists of 7,466 formulae. To keep the graph-structure comprehensible, we do not display relations with a belief below a threshold of 0.8. In the graph, the 5 objects cluster those segments that are in equivalence relations with each object. Many relations between segments are incorporated by relating out-of-sequence segmentations, which are visible as smaller loops in the segment relations.

Figs. 7.12 and 7.13 show representative SLAM graphs of two of the found objects. Out-of-sequence relations also produce loops in the pose graphs. Each view pose is attributed multiple segmentations of the same key view towards different other key views. While the right chair in Fig. 7.12 is only seen by equivalent segments, the pose graph of the object that subsumes left chair and background is more complex (Fig. 7.13). It not only has view poses for the segments that see the complete object, but also for segments that partially observe it. If a segment in a key view represents only parts of the object, it appears

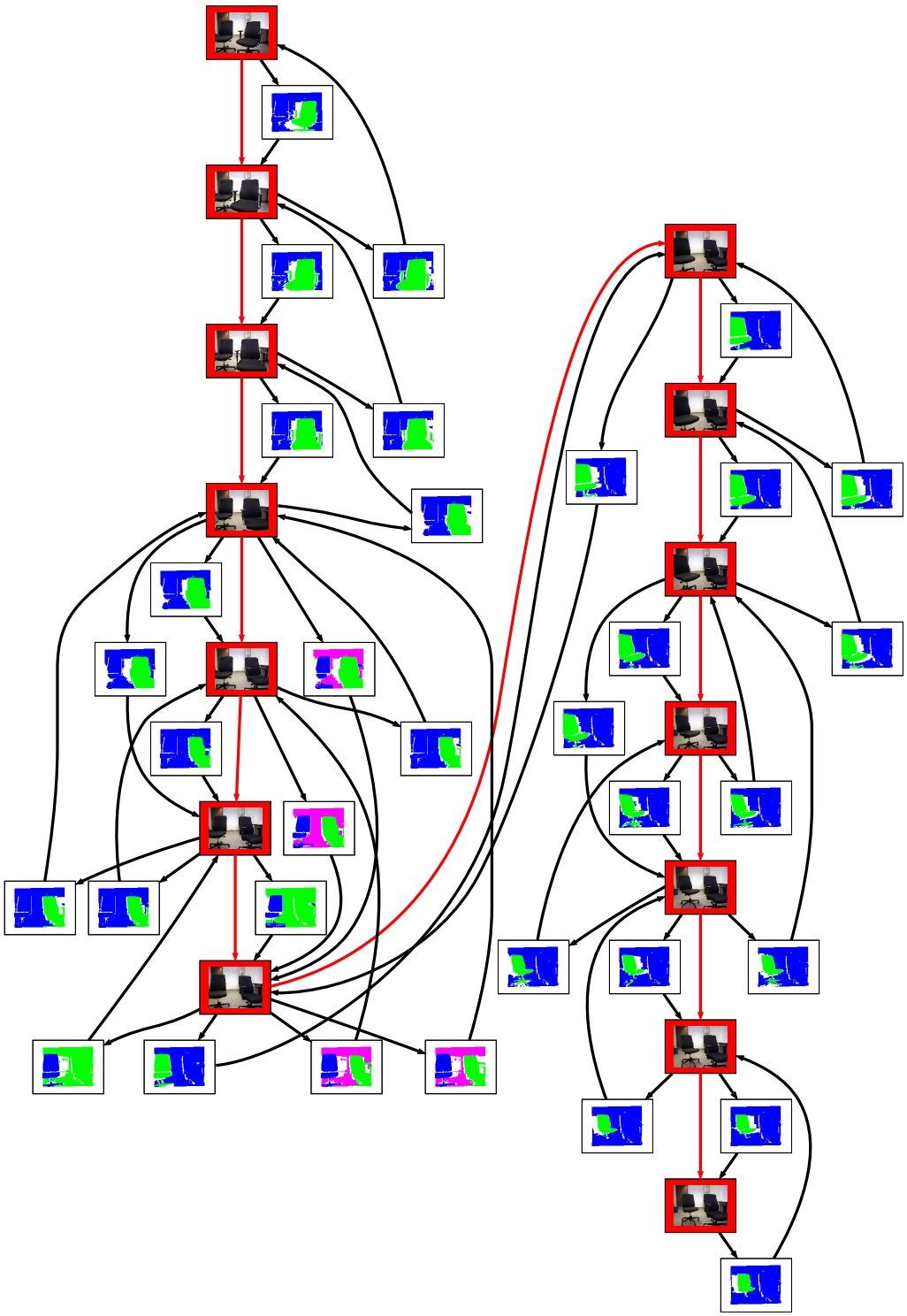

Figure 7.10.: Extracted key views and segmentations on the chairs sequence. Red arrows depict the temporal sequence of the key views. Black arrows point from segmented to connected key view.

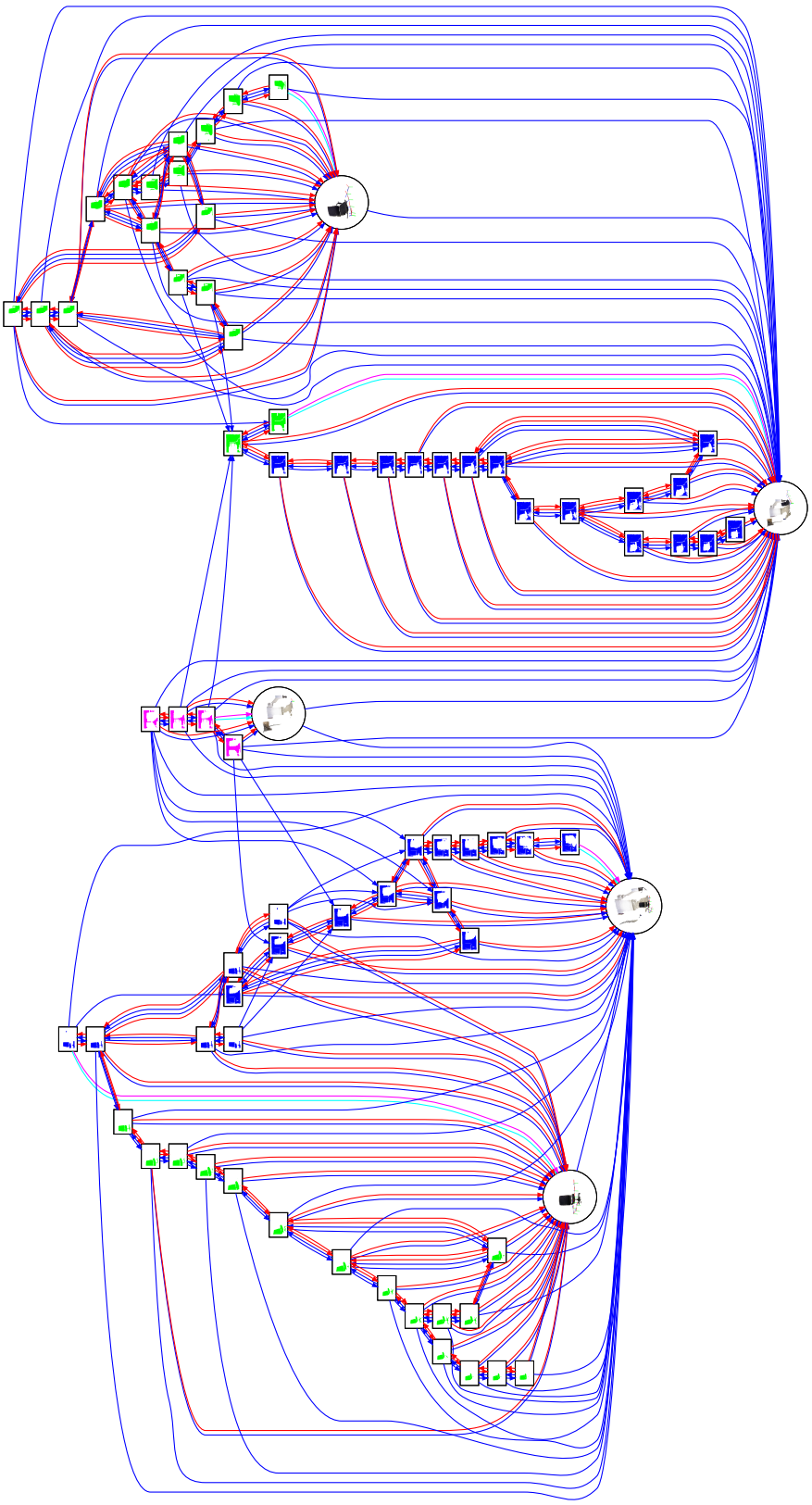

Figure 7.11.: Graph of valid relations on the chairs sequence. Blue/cyan: part-relation, red/magenta: equivalence relation, cyan/magenta: segment-object evidence relation.

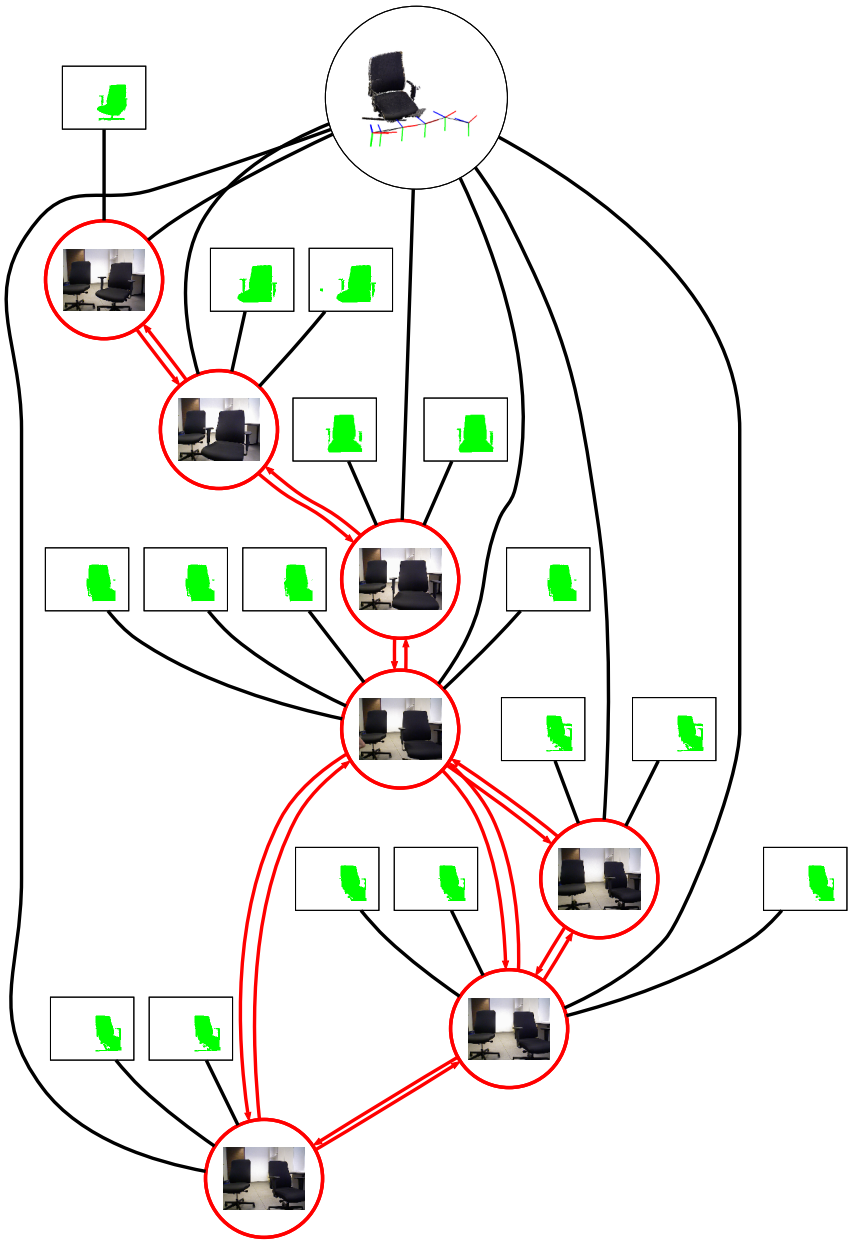

Figure 7.12.: SLAM graph of one object (black circle) on the chairs sequence. The view poses are shown as red circles, their interior displays the key view corresponding to the view pose. Spatial constraints in the pose graph are shown as red edges.

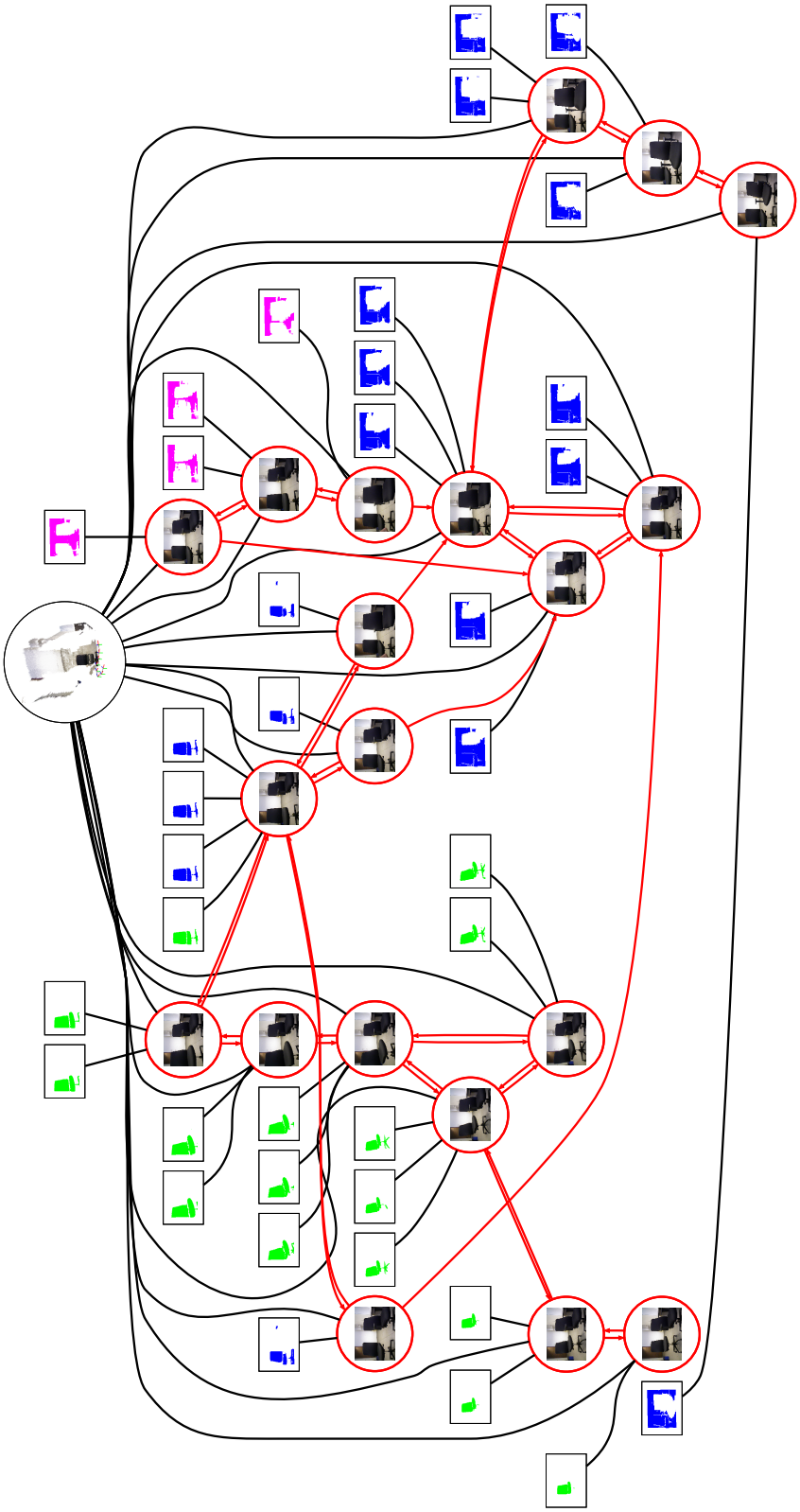

Figure 7.13.: SLAM graph of one object (black circle) on the chairs sequence. The view poses are shown as red circles, their interior displays the key view corresponding to the view pose. Spatial constraints in the pose graph are shown as red edges.

in a different view pose node than segments within the same key view of the complete object. This is necessary, since the parts move differently between the segmented key views and create different spatial constraints.

The resulting objects and the hierarchical relations between them are shown in Fig. 7.14. Our method finds left and right chair as well as the background segment. It also includes two objects that are composed of the background and either one of the chairs. We display the objects by overlaying the RGB-D measurements of their segments from their estimated view poses. We use segments that are in both part and equivalence relations with the objects.

The hierarchy reflects which segment splits and merges have been observed. Between key views, often one chair has been moving with respect to the background and the other chair. Both chairs could also be observed to move simultaneously with respect to the background. Our approach correctly recognizes that the background segment is part of the two objects that combine the background with either one of the chairs.

7.3.1.2. Container Sequence

The container sequence is more difficult than the chairs sequence. The camera is moving during the recording such that naive background subtraction would not be possible. The objects, furthermore, have to be singularized in a three-level hierarchy from drawer to container to background. Finally, large parts of the drawer are occluded while the container is closed in which case only the front panel of the drawer is visible.

Fig. 7.15 shows the 6 key views and 20 segmentations used by our approach. As the motion of the objects is on a smaller scale, key views are related with large temporal gaps between them. Our approach finds 4 objects in the sequence. In addition to the singularized objects background, container, and drawer, it also finds an object that combines background, container, and drawer. This is caused by the sequence of split and merge events: the container is observed static with the background while the drawer is moving. The valid relations inferred are shown in Fig. 7.16. The MLN has 4,460 formulae after the last frame. As in the chairs sequence, segments cluster at the objects with which they are equivalent.

Figs. 7.17 and 7.18 visualize the object SLAM graphs for the background and the drawer objects. For the background, the segments observe the object equivalently. Hence, each key view that is segmented for the background, is included once as a view pose in the pose graph. This is also the case for the drawer for most of the key views. For one key view, however, the whole visible part of the drawer, as well as the front panel alone is segmented. While the segments are determined to not be equivalent by their overlap, the front panel is observed as part of the whole drawer. By this, a spatial constraint from the front panel to the whole drawer is included in the pose graph. Remarkably, although these individual segments are not directly equivalent through overlap,

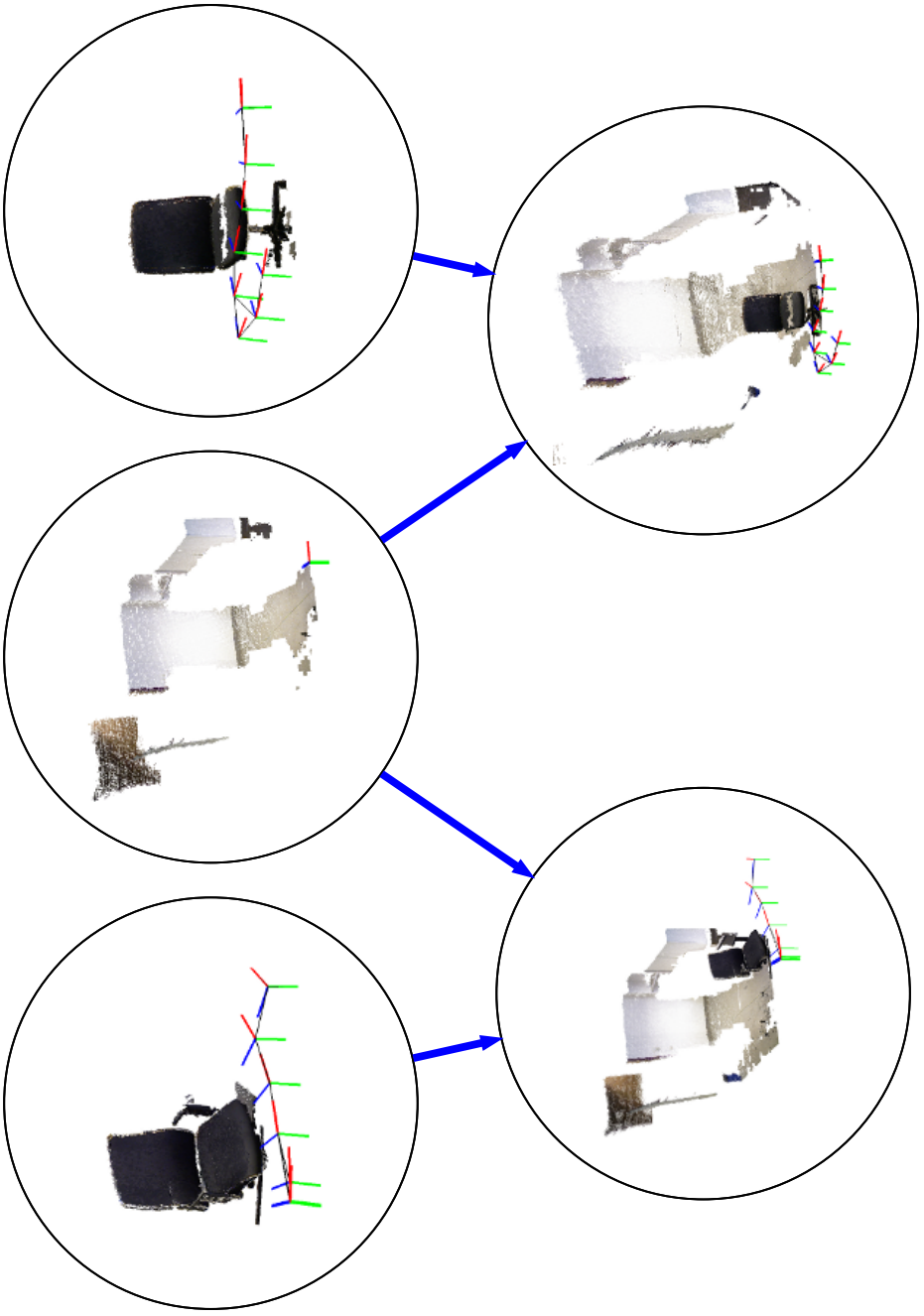

Figure 7.14.: Discovered objects (black circles) and valid part-relations (blue arrows, point from part to containing object) on the chairs sequence.

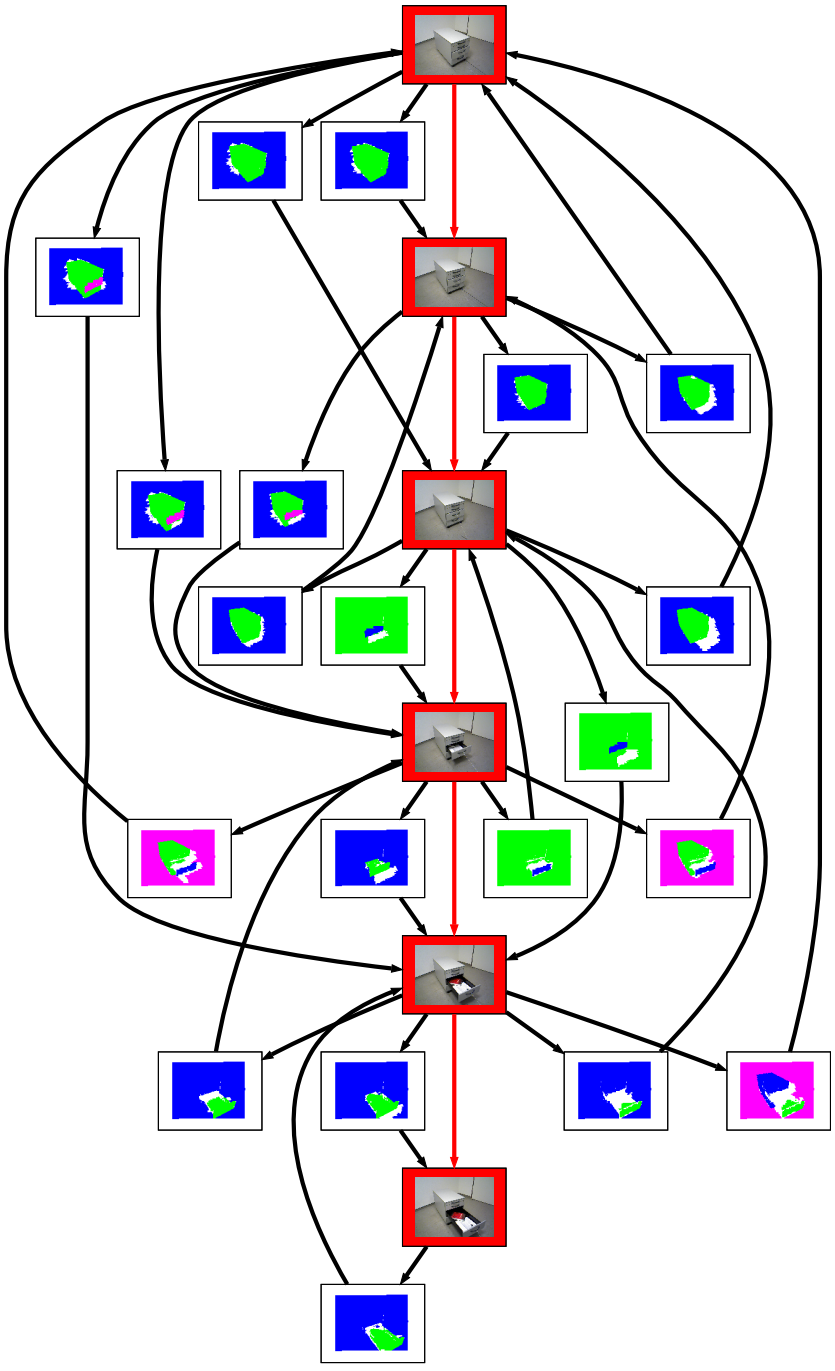

Figure 7.15.: Extracted key views and segmentations on the container sequence. Red arrows depict the temporal sequence of the key views. Black arrows point from segmented to connected key view.

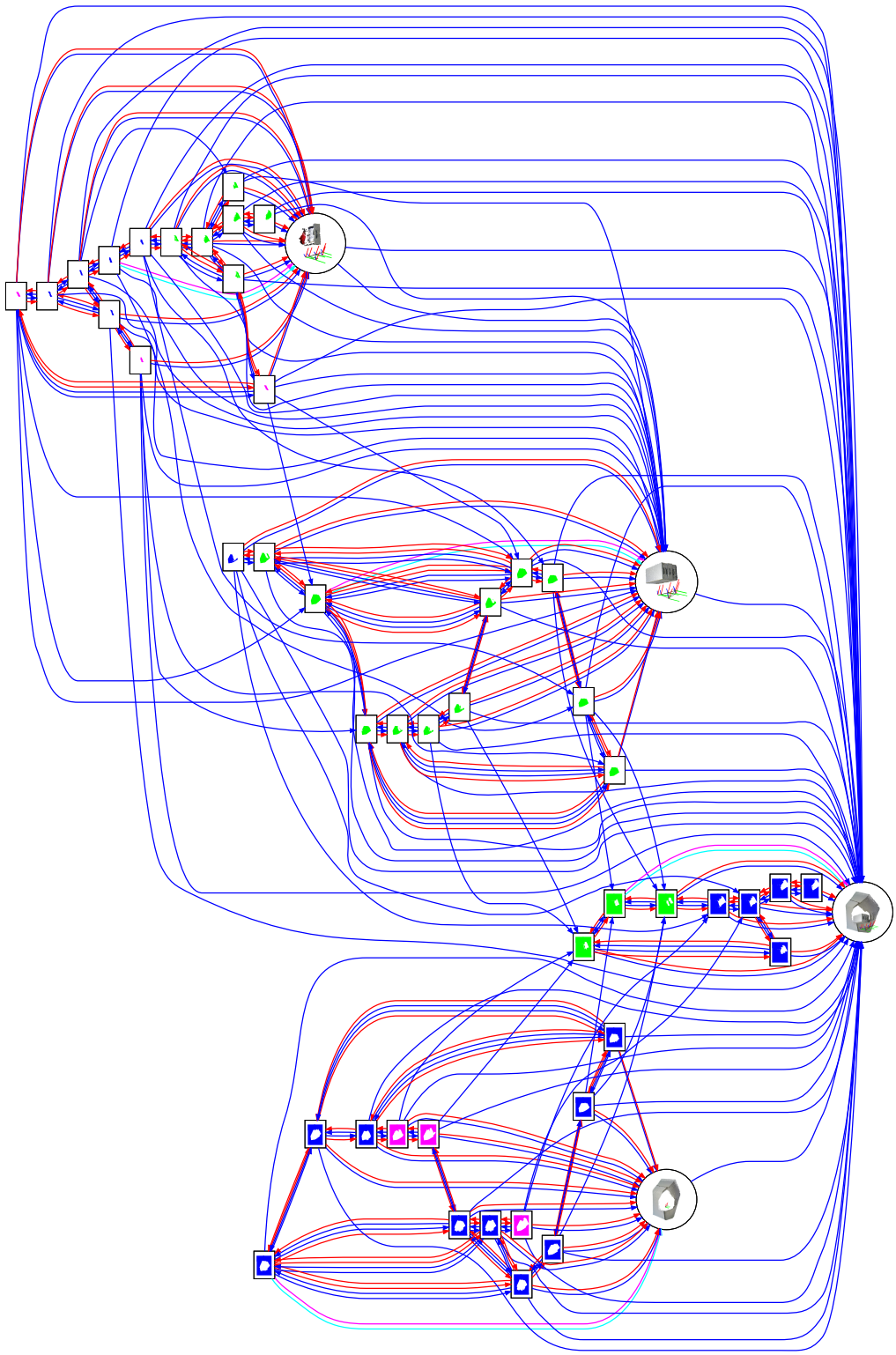

Figure 7.16.: Graph of valid relations on the container sequence. Blue/cyan: part-relation, red/magenta: equivalence relation, cyan/magenta: segment-object evidence relation.

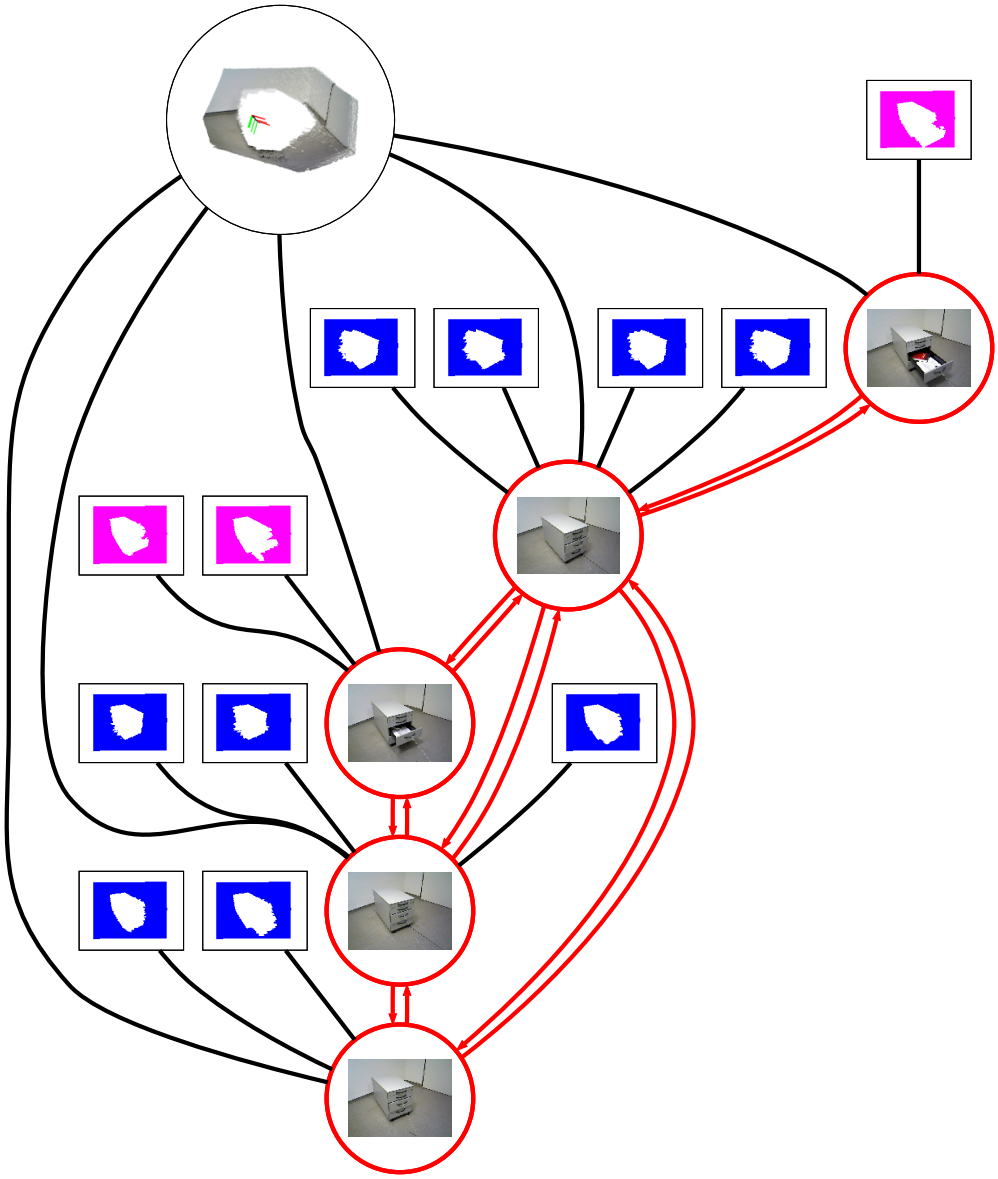

Figure 7.17.: SLAM graph of one object (black circle) on the container sequence. The view poses are shown as red circles, their interior displays the key view corresponding to the view pose. Spatial constraints in the pose graph are shown as red edges.

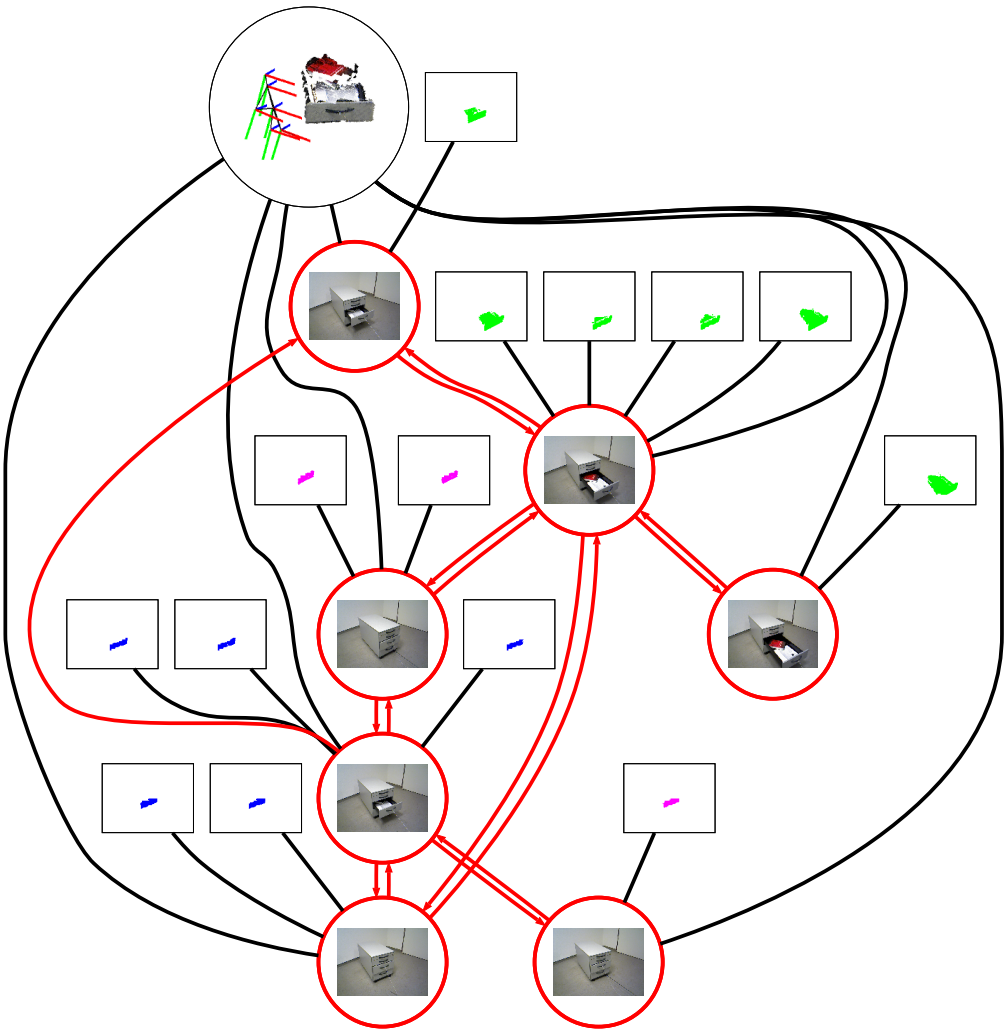

Figure 7.18.: SLAM graph of one object (black circle) on the container sequence. The view poses are shown as red circles, their interior displays the key view corresponding to the view pose. Spatial constraints in the pose graph are shown as red edges.

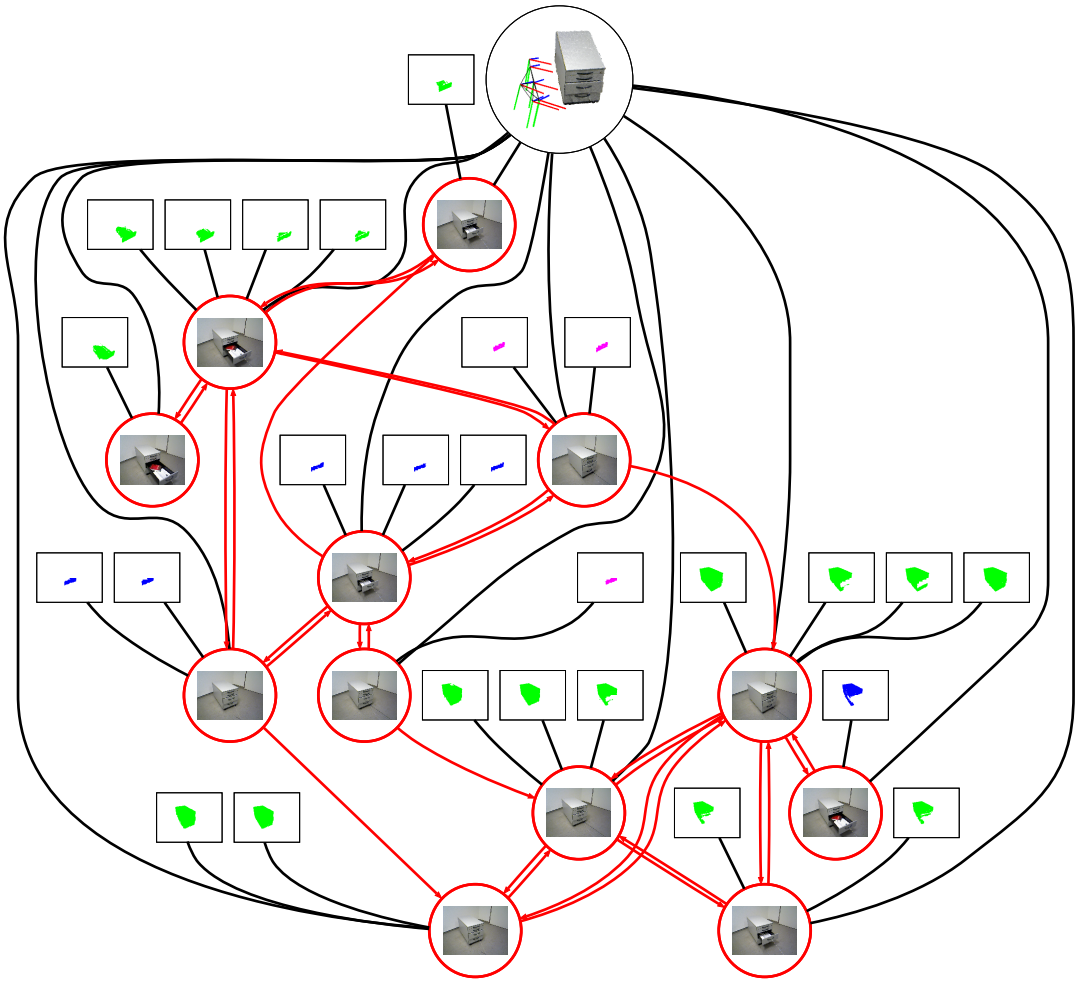

Figure 7.19.: SLAM graph of one object (black circle) on the container sequence. The view poses are shown as red circles, their interior displays the key view corresponding to the view pose. Spatial constraints in the pose graph are shown as red edges.

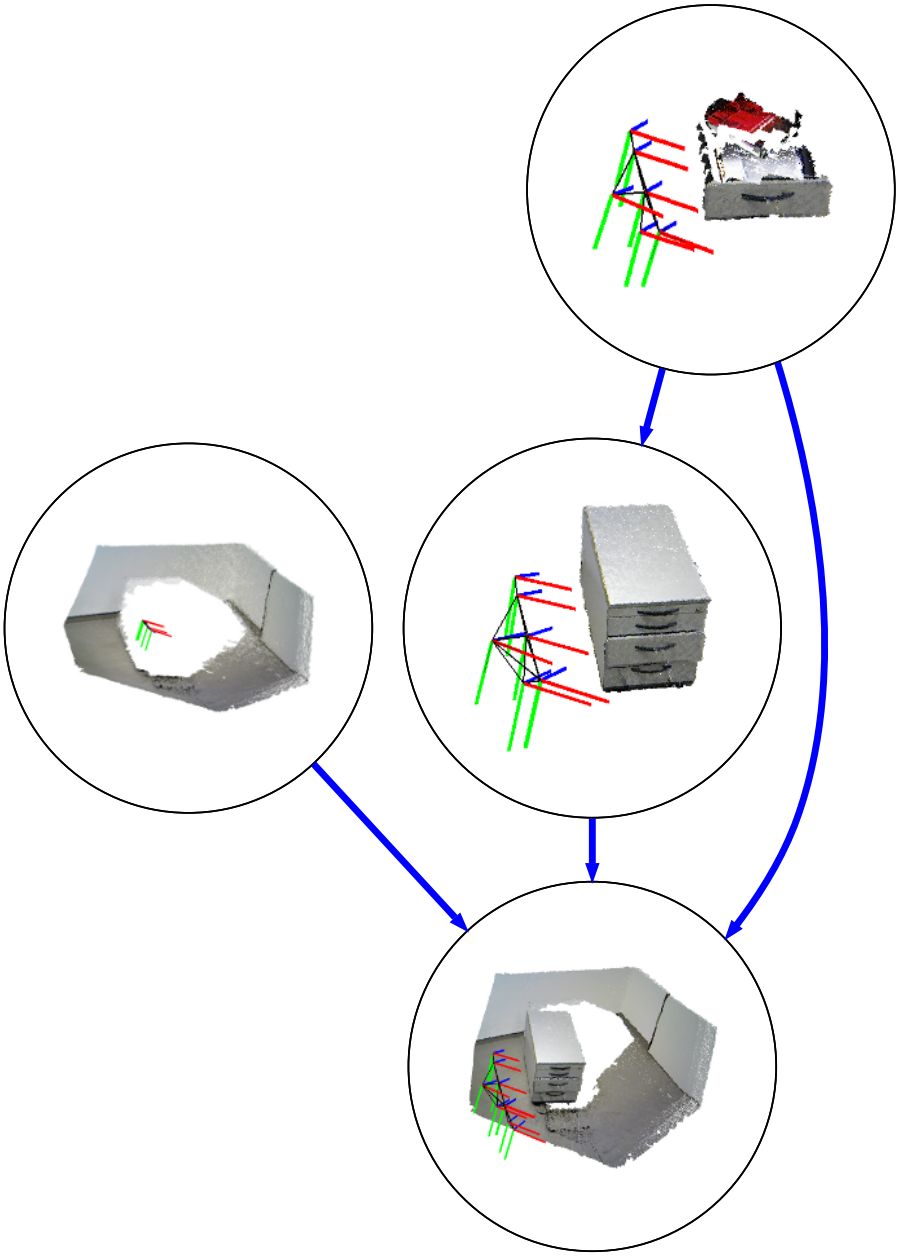

Figure 7.20.: Discovered objects (black circles) and valid part-relations (blue arrows, point from part to containing object) on the container sequence.

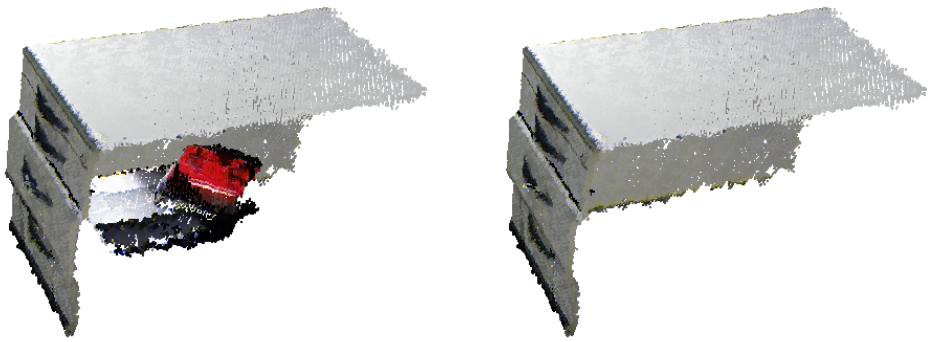

Figure 7.21.: Our method makes the drawer inside the container explicit. The drawer segments are part of the shown container object (left). They are not in equivalence relations with the object (right).

sequence	tracking	key view addition	out-of-sequence relation	belief propagation	pruning	pose graph update	total
chairs	0.139 (0.257)	1.025 (2.042)	0.010 (0.876)	0.127 (6.597)	0.0002 (0.001)	0.001 (0.003)	0.301 (7.626)
container	0.208 (0.298)	0.921 (1.444)	0.017 (1.143)	0.092 (12.268)	0.0002 (0.002)	0.0001 (0.004)	0.329 (13.652)

Table 7.1.: Average (maximum) run-time in seconds for the individual parts of the processing pipeling of our hierarchical object discovery and dense modeling approach.

our probabilistic reasoning approach recognizes the segments as equivalent to the drawer. The front panel overlaps to a large degree in both directions with many other segments. Those segments have strong evidence to be equivalent with the drawer.

Fig. 7.19 shows the object SLAM graph of the container. It is more complex than the graphs of the drawer and the background, as it also includes view poses for segments in part-relations with the container.

The discovered hierarchy between the objects can be seen in Fig. 7.20. Our approach correctly discovers that the drawer is part of the container, which in turn moves separately with respect to the background. All objects are part of the combined object of background, container, and drawer. Fig. 7.21 displays that our approach makes the drawer explicit as a part inside the container.

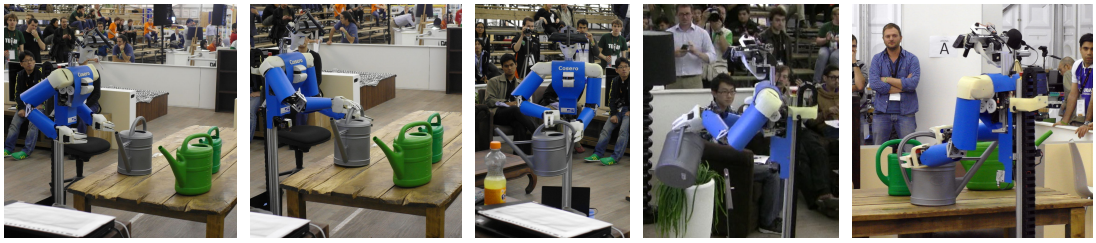

Figure 7.22.: Cognitive service robot Cosero manipulates an unknown watering can during the Open Challenge at RoboCup 2013.

7.3.1.3. Run-Time

The run-time of our approach on both sequences is shown in Table 7.1. Keeping track of the current's image segmentation with respect to the reference key view requires run-time similar to the timing results in Ch. 4. Instantiating a new key view v_t involves two segmentations S_t^{t-1} and S_{t-1}^{t} that are run for several iterations until convergence. In average this takes 1.025 s on the chairs and 0.921 s on the container sequence. In each frame, we search for one new out-of-sequence relation. If a new relation is included, one segmentation has to be determined and the relations between segments and objects need to be updated. This amounts in average to 0.01 s and 0.017 s. The maximum times of 0.876 s and 1.143 s occur if a relation is established. We only search for out-of-sequence relations, if no key view is added for the current image. MLN inference is also efficient in average. It can, however, take many iterations and several seconds to converge if ambiguous evidence needs to be balanced. Pruning objects and object SLAM graphs as well as updating the object SLAM graphs with new relations costs negligible time. In average, the total run-time is governed by the time required for tracking and belief propagation. If new key views or out-of-sequence relations are added, or if relational information is ambiguous, run-time can peak up to a few seconds. In our current approach, both the estimation of new segmentations as well as belief propagation is run until convergence. While such peaks are infrequent and their magnitude is low, in future work instead, peaks could be avoided by distributing the computational load from a single image to multiple subsequent images.

7.3.2. Object Manipulation Skill Transfer

We publicly demonstrated our approach to object manipulation skill transfer during the Open Challenge at RoboCup 2013 in Eindhoven, Netherlands. Our robot Cosero transfered its skills for handling a watering can to another instance of cans. The jury had the choice between two instances of cans that clearly differed in shape from the example can. Fig. 7.22 shows images taken during

the demonstration[1].

We specified bimanual grasp poses and the can's end-effector on the example instance. The demonstration involved several actions in sequence: grasping the can, lifting the can from the table, retracting the can close to the robot body, approaching a plant, watering a plant, and placing the can back on the table. We defined the lifting, retracting, and placing motion relative to the mean pose of the grasps. Approaching and watering the plant have been specified relative to the can's end-effector. In several parts, the robot moves the can also with its mobile base. Cosero successfully performed this demonstration which received high scores from the jury consisting of team leaders. It was one important contribution for winning the 2013 RoboCup@Home competition.

7.4. Related Work

7.4.1. Hierarchical Object Discovery and Dense Modelling

Image segmentation into meaningful objects is an actively researched topic in computer vision (e.g., (Li et al., 2009; Arbelaez et al., 2011; Carreira and Sminchisescu, 2012)). Bottom-up cues for single-image segmentation such as texture (Cremers et al., 2007; Delong et al., 2012) or 3D-shape (Holz and Behnke, 2012; N. Silberman and Fergus, 2012) often do not suffice to find segment borders that coincide with the boundaries of objects. Thus, they are frequently combined with top-down cues to integrate spatial and semantic context (e.g., (Carreira and Sminchisescu, 2012)). Motion is a further important bottom-up cue that can be utilized in image sequences. In contrast to texture and shape, common motion provides unambiguous segmentation hints for the constituent parts of a rigid object.

The mapping of static as well as dynamic parts of environments from a sequence of measurements is investigated in the robotics community. Early work focused on 2D mapping using laser scanners. Anguelov et al. (2002) learned templates and object classes of non-stationary parts of an environment in a two-level hierarchical model. Hähnel et al. (2003) filter dynamic objects and only map the static parts of the environment in 2D. They then extract 3D models of the dynamic parts by stitching the laser measurements. In SLAMMOT (Wang et al., 2004), dynamic objects are detected in 2D laser scans and tracked while SLAM maps the static environment. We integrate 3D motion segmentation in a SLAM framework that also reasons about hierarchical relations between object parts.

Several approaches have been recently proposed that learn 3D articulation models of objects. Sturm et al. (2011) track the 3D rigid-body motion of planar rectangles in RGB-D images and fit articulation models to the segments.

[1] A video can be found at http://www.youtube.com/watch?v=I1kN1bAeeB0

Katz et al. (2012) track RGB image features and retrieve 3D trajectories of the features from depth measurements of an RGB-D sensor. They segment the features into groups with consistent 3D rigid-body motion in a generate-and-test clustering algorithm. Finally, they fit articulation models to the 3D motion trajectories of the segments. Our approach does not explicitly model the articulation of objects, but singularizes objects by their motion in an unsupervised way. It learns dense multi-view models of the objects and hierarchical relations between them. Articulation models could be fitted to the relative motion of a part to its containing segment.

For unsupervised learning of object models, Ruhnke et al. (2009) propose an approach to learn 3D object models from multiple 3D laser scans. To segment the object, they fit planes to the background and remove the background from the 3D scans. We do not make such strong assumptions on the structure of the background. Instead, we track and map the background as one of the objects in the map. The work by Herbst et al. (2011) discovers objects using an RGB-D camera through scene differencing. For this, they recover the camera trajectory using an RGB-D SLAM approach, assuming that the SLAM method is sufficiently robust to the changes in the scene. We formulate our problem to simultaneously estimate the motion of all segments within an image, treating background and foreground segments equally.

7.4.2. Object Manipulation Skill Transfer

Our approach to skill transfer can be seen as a variant of learning from demonstration which is actively researched in the robotics community (Billard et al., 2008). Very recently, Schulman et al. (2013b) also proposed an approach in which motion trajectories are transfered between shape variants of objects. They primarily demonstrate tying knots in rope (Schulman et al., 2013b) and suturing (Schulman et al., 2013a), while they also show examples for folding shirts, picking up plates, and opening a bottle. Their non-rigid registration method is a variant of the thin plate spline robust point matching (TPS-RPM) algorithm. We demonstrate bimanual tool-use, and propose to select tool end-effectors as reference frames for the example trajectory, where it is appropriate. In contrast to their method, we do not assume the estimated displacement field to be valid at any pose on the motion trajectory. Instead, we make example motions relative to reference frames. These reference frames are transformed between example and new object.

7.5. Summary

We proposed two methods for perceiving objects and scenes that release assumptions on rigidness. The first method allows moving objects in a modeled scene

while still assuming the individual objects to be rigid. With our rigid multi-body registration approach, we segment pairs of key views in RGB-D image sequences into the rigidly moving parts and estimate their 6-DoF motion. We acquire dense 3D models of the moving objects by relating observed image segments to the objects, and estimating the view poses of the segments through pose-graph SLAM. We also observe equivalence of objects and part hierarchies from the splitting and merging of segments. The relations between segments and objects are reasoned on within a probabilistic framework to improve robustness for uncertain decisions that would be conflicting in the deterministic case. The part hierarchy and trajectory estimates in the object SLAM graphs could be used for fitting articulation models between parts. In future work, we will investigate the use of additional cues to motion, texture, and shape for segmentation such as co-occurrence or pretrained object classifiers. Releasing the assumption on the rigidness of the parts is also a reasonable next step for future research.

Our second perception approach is tightly coupled with robot control. We transfer example motions for an object instance to new instances of the same object class that differ in shape. We express the motion in terms of grasp poses and a motion relative to a reference frame on the object. Our deformable registration method is used to transfer these poses and frames between the object instances. Future research could involve the consideration of collisions, occlusions, articulated and deforming objects, and physical aspects such as dynamics and elasticity. The latter would also necessitate more complex physical models which are to be perceived with the robot's sensors.

8. Semantic Object-Class Perception

In the previous chapters we proposed methods for representing the geometry and appearance of objects densely in MRSMaps. The models are used for pose estimation and tracking, for parsing scenes for moving parts and hierarchical relations between them, or for establishing correspondence between object shapes in skill transfer. Robots that perform complex tasks in unstructured environments also require the ability to categorize surfaces into semantic classes. Made persistent in a semantic map this knowledge is available for reasoning about tasks and for communication with humans.

We propose a real-time approach to learn semantic maps from a moving RGB-D camera. Our semantic mapping system integrates efficient SLAM with object-class segmentation of RGB-D images. The RGB-D frame is segmented for object classes using random decision forests(RFs) concurrently which is facilitated by a real-time implementation on a GPU. Our image segmentation approach uses depth for scale-invariance and incorporates shape and texture cues seamlessly to provide a probabilistic labeling into object classes. The probabilistic image labeling is fused in 3D within a Bayesian framework given the trajectory estimate of SLAM. By this, segmentation evidence from various view points improves the classification accuracy in the map.

8.1. RGB-D Object-Class Segmentation using Random Decision Forests

8.1.1. Structure of Random Decision Forests

RFs $\mathcal{F}$ are ensembles of K binary decision trees $\mathcal{T}_k$ (Breiman et al., 1984). Each node n in a tree classifies an example by a binary decision on a scalar feature function that quantifies local appearance or shape in the image. In addition,

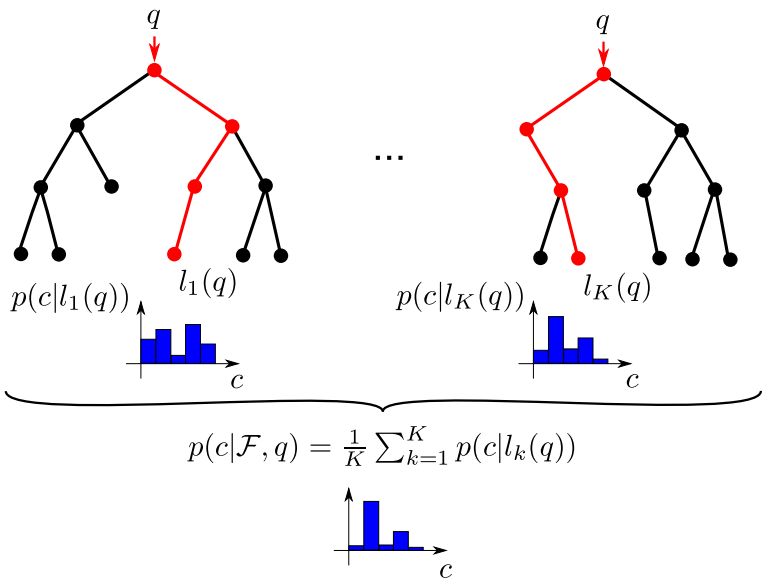

Figure 8.1.: Random decision forests (RFs). Query pixels q are classified in a binary decision cascade in each tree. Nodes in a tree cast binary decisions on the pixels. Query pixels are soft classified by the empirical class probability $p(c \mid l(q))$ of training pixels that arrive at a leaf $l(q)$. The posterior classification probability for the RF is determined by the average over trees.

each node is associated with a distribution $p(c \mid n)$ over class labels $c \in \mathcal{C}$ that arrived at the node during training. Randomness is injected into the classifier by considering only a random subset of the training data for generating a tree and by sampling node functions from only a random subset of the available binary decision functions. In this way, trees are decorrelated and generalization performance increases.

The probabilistic labeling at a query pixel q is determined as the posterior distribution over class labels encoded in the forest (illustrated in Fig. 8.1). In this process, the example pixel is passed down each decision tree $\mathcal{T}_k$, branching at each node according to its binary decision criterion until a leaf node l is reached. The posterior distribution is computed by averaging over the individual distributions at the leaf nodes $l_k(q)$ that the example reaches, i.e.,

$$p(c \mid \mathcal{F}, q) = \frac{1}{K} \sum_{k=1}^{K} p(c \mid l_k(q)).$$

8.1.2. RGB-D Image Features

As scalar feature functions (i.e., features) we determine differences in local regions of depth or color. Dense depth is used to normalize the features for scale

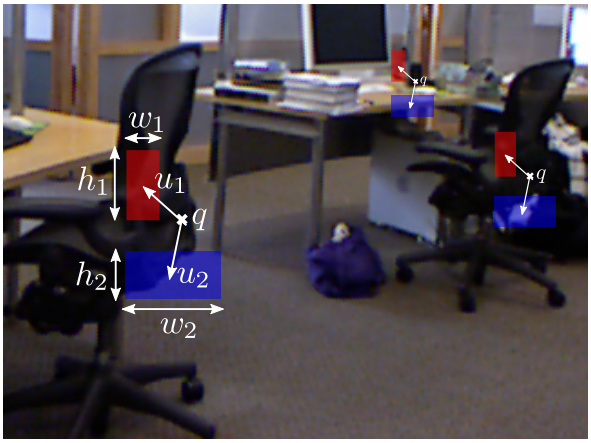

Figure 8.2.: Random decision forest features. Local shape and appearance at a query pixel q is calculated from the difference of average values in two offset regions. We exploit dense depth to normalize for scale changes and scale relative offset locations u_i and region extents w_i, h_i with the inverse of the depth $d(q)$ at the query pixel.

changes (see Fig. 8.2). More formally, we parametrize a feature evaluated at pixel q by

$$f_\theta(q) := \frac{\sum_{p \in R_1(q)} \phi_1(p)}{|R_1(q)|} - \frac{\sum_{p \in R_2(q)} \phi_2(p)}{|R_2(q)|}, \tag{8.1}$$

where $R_j(q) := R\left(q + \frac{u_j}{d(q)}, \frac{w_j}{d(q)}, \frac{h_j}{d(q)}\right)$ is the rectangular image region at the offset u that is normalized in offset position and size by the depth $d(q)$ measured at the query pixel. The features are configured by parameters θ that comprise unnormalized offset positions u_j, region extents w_j, h_j, and image channels ϕ_j. Note, that we restrict comparisons to either two depth regions or between any two regions in color channels, and represent color in the CIE Lab color space. In the depth image, the region size $|R_j(q)|$ counts the number of valid depth readings in the region. If an offset region contains no valid depth measurement or lies beyond the image, the pixel traverses to the right child node. We efficiently implement region features using integral images.

Each node in the decision tree decides on the query pixels with a threshold τ to either pass the pixel further to its left or right child. Individually, each feature gives only small information about the object class at a pixel. Within the cascades in the decision trees, however, the tests describe complex texture and shape patterns which allows for accurate pixel classification.

8.1.3. Training Procedure

We train each of the K decision trees with a subset $\mathcal{D}$ of images from the training dataset. From each image we extract N pixels randomly for training. We stratify the training examples by resampling to a uniform distribution in class labels in order to normalize the amount of training examples for object size. We will, however, have to consider the actual distribution of class labels in the training images at later stages in order to incorporate the prior probability of each class into the classifier.

We train the decision trees in a depth-first manner by choosing feature parameters θ and a threshold τ at each node and splitting the pixel set Q accordingly into left and right subsets Q_l and Q_r:

$$\begin{aligned} Q_l(\theta,\tau) &:= \{q \in Q \mid f_\theta(q) < \tau\} \text{ and} \\ Q_r(\theta,\tau) &:= \{q \in Q \mid f_\theta(q) \geq \tau\}. \end{aligned} \tag{8.2}$$

Since the parameter space cannot be evaluated analytically, we sample P random parameter sets and thresholds (e. g., $P = 2000$) and select feature and threshold that yield maximal information gain

$$I(\theta,\tau) := H(Q) - \sum_{s\in\{l,r\}} \frac{|Q_s(\theta,\tau)|}{|Q|} H\left(Q_s(\theta,\tau)\right), \tag{8.3}$$

where $H(Q) := -\sum_{c\in\mathcal{C}} p(c \mid Q) \log_2\left(p(c \mid Q)\right)$ is the Shannon entropy of the distribution of training class labels in pixel set Q. This splitting criterion finds feature parameters and threshold that most distinctively separate the pixel set at a node. Each node is split until a maximum depth is reached in the tree, or the number of pixels lies below a minimum support threshold.

At each leaf node l, we want to maintain the distribution $p(c \mid l, D)$ of pixels of class c that arrive at the node from the original training set. Since we train the decision tree from pixels with equally distributed class labels, we actually measure the class distribution $p(c \mid l, Q)$ of training pixels Q at the leaf, i.e.,

$$p(c \mid l, Q) := p(c_q \mid l, q \in Q) = p(c_q \mid l, q \in Q, q \in \mathcal{D}). \tag{8.4}$$

The distribution of interest can be obtained by applying Bayes rule:

$$\begin{aligned} p(c \mid l, Q, \mathcal{D}) &= \frac{p(q \in Q \mid c_q, l, q \in \mathcal{D})\ p(c_q \mid l, q \in \mathcal{D})}{p(q \in Q \mid l, q \in \mathcal{D})} \\ &= \frac{p(q \in Q \mid c_q, q \in \mathcal{D})\ p(c_q \mid l, q \in \mathcal{D})}{p(q \in Q \mid q \in \mathcal{D})}. \end{aligned} \tag{8.5}$$

For the desired distribution we obtain

$$p(c_q \mid l, q \in \mathcal{D}) = \frac{p(c_q \mid l, q \in Q)\ p(q \in Q \mid q \in \mathcal{D})}{p(q \in Q \mid c_q, q \in \mathcal{D})}. \tag{8.6}$$

We further reformulate the probability of a pixel of class c to be included in the class-equalized training data Q to

$$p(q \in Q \mid c_q, q \in \mathcal{D}) = \frac{p(c_q \mid q \in Q)\ p(q \in Q \mid q \in \mathcal{D})}{p(c_q \mid q \in \mathcal{D})} \tag{8.7}$$

and obtain

$$p(c_q \mid l, q \in \mathcal{D}) = \frac{p(c_q \mid l, q \in Q)\ p(c_q \mid q \in \mathcal{D})}{p(c_q \mid q \in Q)}. \tag{8.8}$$

By design, $p(c_q \mid q \in Q)$ is uniform among class labels and, hence, we incorporate the distribution of classes in the complete training set into the leaf distributions through

$$p(c \mid l, D) = \eta\ p(c \mid l, Q)\ p(c \mid \mathcal{D}), \tag{8.9}$$

where $\eta^{-1} := p(c \mid Q) = 1/|\mathcal{C}|$.

We found that if there is a large imbalance of class pixel occurrences in the image, single training pixels from frequent classes that reach a leaf may outweigh many pixels from less frequent classes, and hence degrade segmentation accuracy dramatically. In such unbalanced datasets we subtract a fraction ρ of the total pixels that reached the leaf from each class count.

8.2. Dense Real-Time Semantic Mapping of Object-Classes

8.2.1. Probabilistic 3D Mapping of Object-Class Image Segmentations

Our online SLAM approach (see Ch. 6) provides an estimate for the motion of the camera $\mathcal{S}$, while object segmentation yields a probabilistic labeling $\mathcal{Z}$ of the image according to the RGB-D images. Our aim is to fuse the object segmentations from individual images into a 3D semantic map. We use our efficient image aggregation techniques in MRSMaps to generate multi-resolution voxel maps that store beliefs on object classification in each voxel.

Formally, we store the belief $Bel(c_v)$ in each voxel v to be labeled as one of the object classes c_v,

$$Bel(c_v) = p(c_v \mid \mathcal{Z}, \mathcal{S}). \tag{8.10}$$

The labeled image pixels are projected into 3D to find corresponding voxels in the map. The beliefs in each voxel v are then updated in a Bayesian framework with the pixel observations $q_{1:N} := \{q_1, q_2, \ldots, q_N\}$ that fall into a voxel:

$$p(c_v \mid q_{1:N}, \mathcal{S}) = \sum_{c_{q,1}, \ldots, c_{q,N}} p(c_v, c_{q,1}, \ldots, c_{q,N} \mid q_{1:N}, \mathcal{S}). \tag{8.11}$$

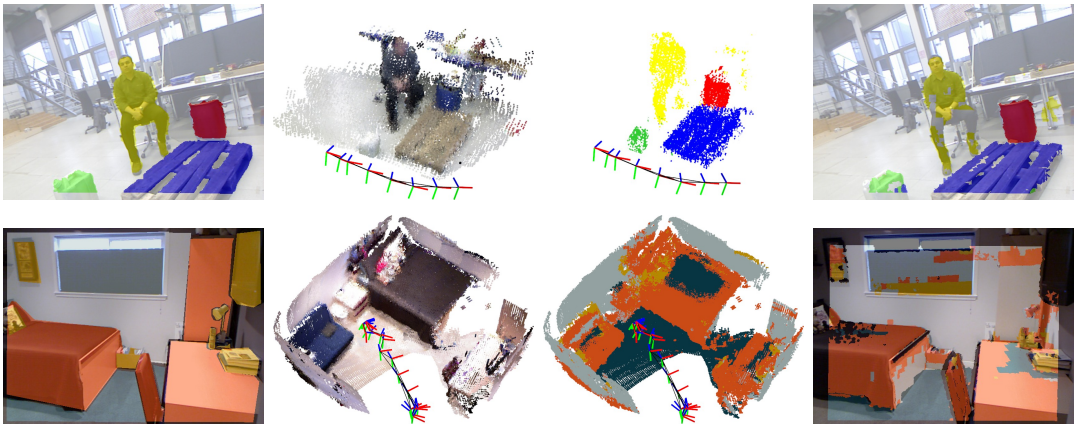

Figure 8.3.: Semantic mapping. From left to right: Ground-truth overlay on RGB image of a scene; Samples from MRSMaps overlayed in 3D and SLAM key view graph; Class belief for MRSMap samples in semantic 3D map; Object-class segmentation backprojected from semantic 3D map into image. Top: AIS Large Objects scene. Bottom: NYU Depth v2 scene.

Note that the known trajectory can be neglected in the further derivation to ease notation. Bayes rule yields

$$p(c_v \mid q_{1:N}) = \sum_{c_{q,1},\ldots,c_{q,N}} p(c_v \mid c_{q,1},\ldots,c_{q,N},q_{1:N}) p(c_{q,1},\ldots,c_{q,N} \mid q_{1:N}). \quad (8.12)$$

The left term is further factored using Bayes rule, while for the right term we impose independence between pixel observations. This yields

$$p(c_v \mid q_{1:N}) = p(c_v) \sum_{c_{q,1},\ldots,c_{q,N}} \prod_i \eta_i \; p(c_{q,i} \mid c_v) \; p(c_{q,i} \mid q_i), \quad (8.13)$$

where $\eta_i := 1/p(c_{q,i} \mid c(q_{i+1}),\ldots,c(q_N))$ are normalization factors for each observation. The RF classifier provides the likelihood $p(c_{q,i} \mid q_i)$ through $p(c_{q,i} \mid q_i, \mathcal{F})$, while the probability $p(c_v) =: Bel_0(c_v)$ incorporates prior knowledge on the belief which we set to uniform in our experiments. For the distribution $p(c_{q,i} \mid c_v) = \mathbf{1}_{\{c_v\}}(c_{q,i})$ we assume a deterministic one-to-one mapping such that

$$p(c_v \mid q_{1:N}, \mathcal{S}) = Bel_0(c_v) \prod_i \eta_i \; p(c_{q,i} = c_v \mid q_i, \mathcal{F}). \quad (8.14)$$

This belief update can be performed recursively in a time-sequential manner which is applied in our online semantic SLAM system.

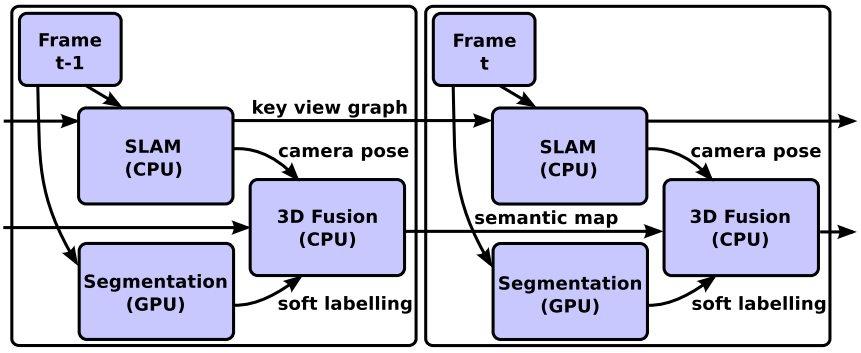

Figure 8.4.: Online semantic SLAM system. Each frame is segmented for object-classes on GPU and processed for SLAM (CPU) in parallel. Results are fused in 3D semantic maps.

8.2.2. Integrated Real-Time Semantic Mapping

We integrate object-class segmentation, SLAM, and semantic 3D fusion into a real-time operating semantic mapping system (see Fig. 8.4). We use the GPU implementation of our object-class segmentation method by Waldvogel (2013) for real-time segmentation. Since object-class segmentation and SLAM are performed on GPU and CPU, respectively, we can execute both components in parallel. Once pose estimate and semantic labeling of the RGB-D image is available, we fuse the labeling into the semantic map of the reference key view.

Each key view in the map maintains a local aggregated semantic map in our approach, since the relative poses of the key views are subject to graph optimization in each frame and, hence, a single global map cannot be maintained. Global segmentation beliefs at a 3D position $Bel(p)$ can be easily obtained by combining the beliefs $Bel(c_v^k)$ of individual key views $k \in \mathcal{K}$ in a single estimate according to

$$Bel(p) = \eta \prod_k Bel(c_v^k), \tag{8.15}$$

where η is a normalization constant and c_v^k is the classification of the maximum resolution node v that contains p in key view k. Note that this process can be significantly sped up by restricting the set of key views $\mathcal{K}$ to the views that contain query pixels or have sufficient frustum overlap with whole query images.

8.3. Experiments

We evaluate run-time and recognition performance of our semantic SLAM method in extensive experiments. We used two datasets to demonstrate our approach on two different qualities of object classes. Both datasets have been recorded using Microsoft Kinect cameras at VGA (640×480) RGB and depth

image resolutions. Since ground truth for the camera trajectory is not available for the datasets, the accuracy of the reconstruction could not be assessed.

The NYU Depth v2 dataset (Silberman et al., 2012) contains 590 RGB-D sequences recorded in 464 scenes with 408,473 frames in total. It comes with 1449 images with manual ground-truth labeling of object-classes. We evaluate on the four abstract object-classes *ground*, *structure*, *furniture*, and *props* that distinguish all 35,064 object instances in the dataset. The dataset has been split into disjunct training and test sets comprising 795 training and 654 test images with ground truth in 359 and 231 sequences, respectively.

We also use the AIS Large Objects dataset introduced in Stückler et al. (2012a) to classify four fine-grained object classes (large objects of *props*-type) from background. It consists of 40 sequences in different scene configurations and has been split into 30 training and 10 test sequences with 500 ground-truth labeled images each (50 per test sequence). The test sequences comprise 5,234 frames ranging between 254 and 858 frames per sequence.

We process the test sequences in real-time on a notebook PC with Intel Core i7-3610QM CPU (2.3 GHz) equipped with an NVIDIA GeForce GTX 675M GPU. Since our method does not process images at full 30 Hz image acquisition rate, it is required to skip frames. For assessing the segmentation, we compare segmentation accuracy of the direct RF maximum likelihood (ML) labeling with the ML labeling obtained by back-projecting the belief in the maps into the test images. Each pixel in the test image queries its corresponding node at maximum resolution in each key view. During SLAM, the image has been registered towards a reference key view. We require that the image pixel was visible in a key view and only consider those key views for which the corresponding node's resolution is equal or finer to the resolution in the reference key view. The belief for the pixel is then queried from this set of key views according to Eq. (8.15). We determine two kinds of labelings from the map: an instantaneous segmentation that is retrieved from the map in its current state when the test image is processed, and a final segmentation after the whole sequence has been mapped.

8.3.1. NYU Depth v2 Dataset

For the NYU Depth v2 dataset, we train RFs on average class accuracy for the four abstract structural object-classes. We optimize the hyper parameters of the RF, such as maximum tree depth, using the Hyperopt (Bergstra et al., 2011) framework and 5-fold cross validation on the training set. Hyperopt performs informed search on the parameter space to efficiently find an optimal solution within the parameter range specified. Still, to optimize the RF in a feasible amount of time, rapid training is required. We therefore accelerate computationally expensive parts of RF training on GPUs. Our implementation is able to train and test about 350 trees per day on a single NVIDIA GeForce GTX

Table 8.1.: RF parameters used in our experiments.

parameter	NYU Depth v2	AIS Large Objects
no. of trees	3	3
pixel samples per image	4537	2000
feature samples per node	5729	2000
threshold samples per node	20	50
max. offset radius (pixel m)	111	120
max. region size (pixel m)	3	10
max. tree depth	18	15
min. sample count in leaf	204	100
histogram bias ρ	0	0.2

Table 8.2.: Run-time per frame on the NYU Depth v2 dataset in ms.

processing step	min	avg	max
image preprocessing	12.0	13.0	29.0
RF segmentation	32.0	44.4	67.0
SLAM	8.0	60.5	346.0
total	51.0	78.0	366.0

TITAN. See Table 8.1 for resulting parameters. On this dataset, the distribution of pixels attributed to each object class is well-balanced, for which a setting of $\rho = 0$ is found through hyper-parameter optimization. While a region size of 3 appears to be small, most features that are selected by the RF are 3×3 regions.

Table 8.3 shows average per-class, class, and pixel accuracy achieved on the test set. Example segmentations are illustrated in Fig. 8.5. Note that the NYU Depth v2 dataset provides a tool for in-filling missing depth readings that is too time-expensive for real-time processing, but has been used in related work on object-class segmentation (Silberman et al., 2012; Couprie et al., 2013). For comparison, we also show results of our RF segmentation method on in-filled depth images. Since we trained our RF method on in-filled images, we fill-in the depth images during real-time experiments by constantly continuing depth from the right, the left, the top, and the bottom in the specified order. In-filling from the right first is motivated by the extrinsic setup of RGB and depth camera. Pixels without valid depth reading cannot be labeled in the 3D map. Hence, we discard them in the segmentation accuracy measure for the real-time experiments.

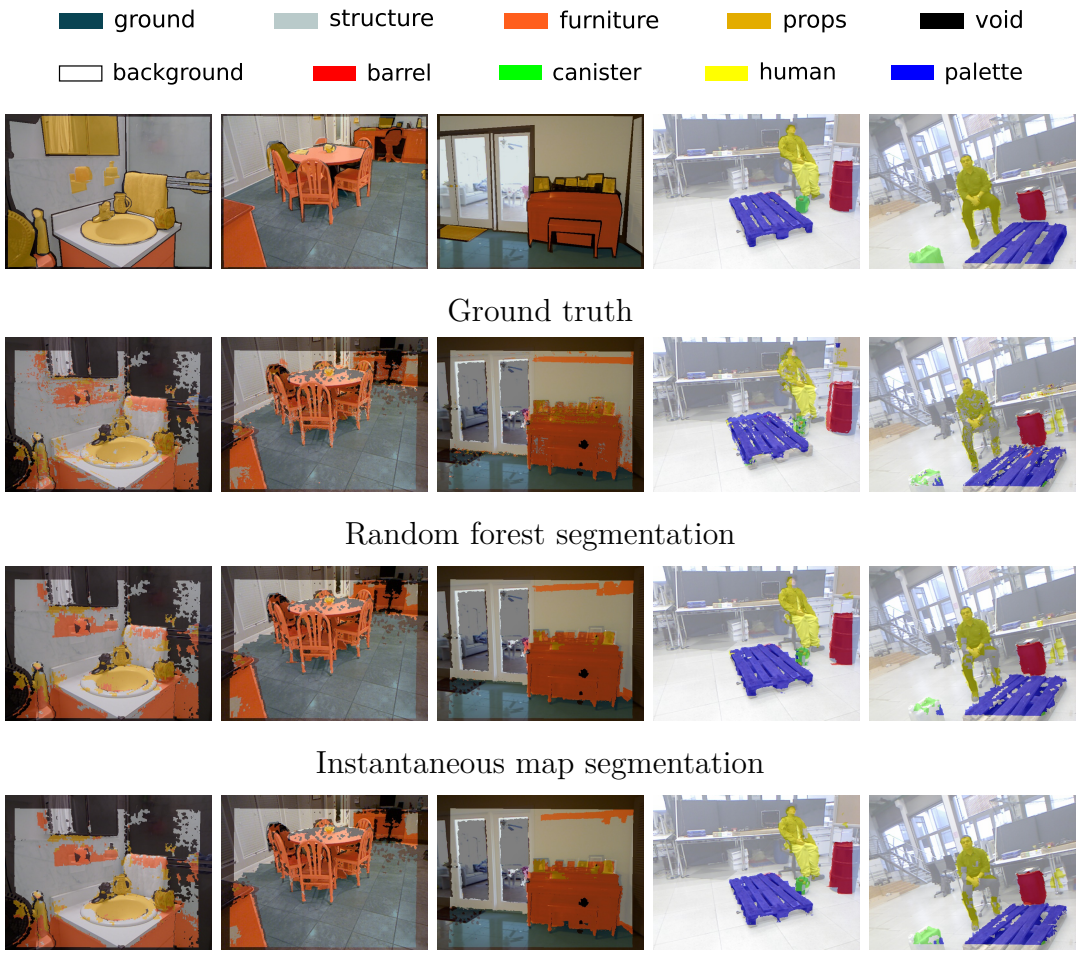

Figure 8.5.: Example labelings on the NYU Depth v2 (3 left) and the AIS Large Objects datasets (2 right).

The results clearly demonstrate that our RF approach already achieves state-of-the-art performance on this dataset. The probabilistic fusion of the individual image segmentations into 3D further boosts segmentation performance by about 2.2% for pixel accuracy and ca. 1.1% for average class accuracy. The larger structural classes improve in segmentation accuracy, while the performance of the smallest object-class (*props*) is decreased in the filtering process. The *props* class was already difficult to segment by our image-based RF approach. We attribute this to the fact that it has the most diversity in appearance, contains difficult objects, and is in parts inconsistently labeled. For instance, in bath room scenes, mirrors are labeled as *props*, which are difficult to distinguish from the reflected surface. Also, carpets on the floor are difficult to distinguish

Table 8.3.: Segmentation accuracy on the NYU Depth v2 test set for 4 structural object-classes. (*) Comparison of segmentation results for in-filled depth images.

method	ground	structure	furniture	props	class avg.	pixel avg.
RF	93.7	80.0	69.4	**20.5**	65.7	68.4
inst. map	95.1	82.3	74.5	14.7	66.4	70.2
final map	**95.6**	**83.0**	**75.1**	14.2	**66.8**	**70.6**
*Silberman et al. (2012)	68	59	70	**42**	59.6	58.6
*Couprie et al. (2013)	87.3	**86.1**	45.3	35.5	63.5	64.5
*RF (ours)	**90.7**	81.4	**68.1**	19.8	**65.0**	**68.1**

from the ground without considering the overall scene context. Our 3D fusion method reduces the segments to the consistently reoccuring parts in the RF segmentation. We note that few of the sequences could only be locally consistently mapped by our RGB-D SLAM approach, mainly due to the fact that only far distance measurements were available in frames or mostly a planar textureless region was visible.

Minimum, average, and maximum run-time per frame in milliseconds for individual processing steps and the overall semantic mapping pipeline are shown in Table 8.2. The average performance of semantic mapping is ca. 78 ms, i.e., 12.8 Hz. The largest part of the processing time is consumed by the SLAM method which is 60.5 ms on average. The time spent for the SLAM method strongly depends on the detail present in the image. If scenes are imaged from close distance, finer resolutions will be represented in the MRSMaps. If new spatial constraints need to be tested, a second image registration is performed which can further increase SLAM run-time to at most 346 ms. Nevertheless, the overall run-time of our approach has not been larger than 366 ms for the 231 test sequences. Note that the overall run-time is not a simple sum of the parts since object-class segmentation and SLAM run in parallel.

8.3.2. AIS Large Objects Dataset

Table 8.1 lists RF parameters used for the AIS Large Objects dataset. The dataset contains object classes of different sizes such as canisters, barrels, and palettes, while large parts of the scene are attributed to the background class. A histogram bias of $\rho = 0.2$ performs well on the dataset. The trained RF prefers large region sizes. In fact, most selected features have region sizes with 10 pixels width or height.

This dataset has been trained and real-time processed without depth in-filling. From Table 8.5 we find that fusion into 3D strongly improves per-class accuracy

Table 8.4.: Run-time per frame on the AIS Large Objects dataset in ms.

processing step	min	avg	max
image preprocessing	10.9	11.2	17.9
RF segmentation	30.4	33.0	42.9
SLAM	19.5	49.2	175.3
total	43.3	64.6	190.5

Table 8.5.: Segmentation accuracy on the AIS Large Objects test set for 5 large object-classes.

method	bg.	barrel	canister	human	palette	class avg. (no bg.)	pixel avg. (no bg.)
RF	97.2	89.5	44.2	58.8	83.3	74.6 (55.2)	92.9 (73.8)
inst. map	97.8	93.9	46.5	**65.6**	88.1	78.4 (58.8)	94.4 (79.1)
final map	**98.0**	**94.0**	**47.5**	65.4	**88.9**	**78.8 (59.2)**	**94.6 (79.4)**

as well as overall class and pixel accuracy.

In these test sequences, our semantic mapping achieved high frame-rates of about 15.5 Hz in average (64.6 ms). Similar to the NYU Depth v2 dataset, most processing time is spent for SLAM. The maximum overall run-time here is much less, since less close-by scenery has been recorded than in NYU Depth v2.

8.4. Related Work

Many mapping approaches build geometric representations of the environment, e.g., using sensors like 2D and 3D laser scanners, monocular and stereo cameras. Comparably less systems map semantics. While most approaches use SLAM as a front-end to obtain a sensor trajectory estimate (Zender et al., 2008; Vasudevan et al., 2007; Meger et al., 2008; Nüchter and Hertzberg, 2008; Castle et al., 2010; Civera et al., 2011), some methods also incorporate the spatial relation of objects into SLAM. Tomono and Shin'ichi (2003), for example, detect polyhedral object models in images and perform SLAM in 2D maps using the detected objects as landmarks. In contrast to our approach, this method is restricted to objects with clearly visible linear edges. Zender et al. (2008) apply SLAM in 2D maps using laser scanners, recognize objects using SIFT features, and map their locations in the 2D map. In addition to SIFT-based recognition, Vasudevan et al. (2007) also detect doors in laser scans since they are important topological objects that

connect rooms. Meger et al. (2008) combine semantic 2D mapping of objects with attention mechanisms. In contrast, we build 3D semantic maps with dense object information. Nüchter and Hertzberg (2008) use ICP, plane segmentation, and reasoning to label planar segments in 3D maps that they acquire using 3D laser scanners. They apply AdaBoost on Haar wavelets and SVM classifiers on contour descriptions to detect objects and persons in the 3D maps. In our approach, we segment the original image data and fuse segmentation evidence from multiple views. Castle et al. (2010) and Civera et al. (2011) propose vision-based mapping of objects. In both approaches, SLAM is solved through feature-based monocular EKF-SLAM. Objects are recognized using SIFT features and persistently maintained in the 3D feature map. The approach of Ranganathan and Dellaert (2007) learns 3D constellation models of places composed of objects using SIFT features. In this approach, the map consists of a set of places with associated models. The aforementioned approaches, however, do not build dense 3D semantic maps. Closely related to our approach are the works by Lai et al. (2012), Sengupta et al. (2013), and Salas-Moreno et al. (2013). Lai et al. (2012) use the confidence score of an object detector to generate a dense soft labeling of an image and integrate the labelings in a voxel representation. The approach requires about 4 seconds per frame and, to the best of our knowledge, has not yet been implemented to perform in real-time with SLAM in the loop. In urban scenes, Sengupta et al. (2013) label stereo images using conditional random fields and fuse the information in 3D stereo sequences. The run-time of this method is reported to be within seconds per frame. The approach by Salas-Moreno et al. (2013) recognizes specific object instances in a scene and estimates the pose of the objects in a map using SLAM techniques. Our method provides dense semantic classification of the surfaces in a map.

We integrate image-based object-class segmentation with SLAM from RGB-D images into a semantic 3D mapping framework. Each image is segmented pixel-wise into object classes and background. Based on the SLAM estimate, this information is then projected into 3D to fuse object recognition results from multiple views. This not only provides 3D segmentations of objects, but also improves classification accuracy.

RFs have been applied to a variety of image segmentation problems such as object-class segmentation (Shotton et al., 2008; Stückler and Behnke, 2010) and human body part labeling (Shotton et al., 2011). Semantic texton forests, proposed by Shotton et al. (2008), use simple features of luminance and color at single pixels or comparisons between two pixels in a RF classifier. Using image-level priors and a second stage of RFs, local and scene context is incorporated into the classification framework. Recently, RFs have been successfully applied for segmenting human body parts and tracking body pose in real-time using depth images. Shotton et al. (2011) propose to normalize feature queries with the available depth to obtain scale-invariant recognition. We extend RFs for object-class segmentation by incorporating both depth and color features. As

in previous own work (Stückler and Behnke, 2010), we use features in color and depth and normalize for scale changes to gain an efficient classifier for RGB-D images. For the problem of object-class segmentation, we also need to consider that objects may vary strongly in size between the classes. We propose a stratification mechanism to balance the training pixels over classes.

8.5. Summary

Our semantic mapping approach combines state-of-the-art object-class segmentation of RGB-D images with accurate RGB-D SLAM. Both methods perform real-time on GPU and CPU, respectively, such that an online semantic mapping system can be integrated.

Our object-class segmentation method is based on RF and makes use of the dense depth available for scale-invariant recognition. Using the camera pose estimates of SLAM, the probabilistic labelings of individual images by our RF approach are fused in multi-resolution voxel maps within a Bayesian framework.

In experiments, we demonstrate run-time efficiency and segmentation accuracy of our approach. We evaluated performance on two datasets with different qualities of object classes. The NYU Depth v2 dataset consists of 590 sequences recorded in indoor scenes, which we segment for structural object classes. Our approach outperforms state-of-the-art approaches to object-class segmentation on this massive dataset. Probabilistic fusion into 3D further increases segmentation accuracy. The whole processing pipeline operates online at approx. 12.8 Hz on average. The second dataset contains large objects that are segmented at good accuracy with our approach. It also performs real-time on these sequences at about 15.5 Hz on average. The semantic information made persistent in our maps could be used in many robotics applications such as object search and manipulation, exploration, or navigation.

Directions for further research include augmenting the RF classifier with concepts such as auto-context or hierarchical segmentation. The accuracy and robustness of the underlying SLAM approach also influences segmentation accuracy. The probabilistic semantic labelling of individual images could be used as a prior in our object detection and tracking methods. This could improve the robustness of the methods for clutter and occlusions. Semantic information could also be incorporated into SLAM to improve data association.

9. Conclusions

In this thesis, we presented innovative approaches for RGB-D environment perception. The approaches are based on multi-resolution surfel maps(MRSMaps), a concise dense representation for RGB-D images and multi-view models.

MRSMaps store the statistics of RGB-D measurements within 3D voxels at multiple resolutions. They respect typical error characteristics of RGB-D sensors by adapting the maximum resolution used for a measurement with squared distance from the sensor. We utilize image neighborhood to efficiently aggregate maps from RGB-D images. The maps are designed for run-time efficient dense registration at the expense of memory usage. Each surfel is added a description of shape and texture in its local context to aid association during registration.

In experiments, we evaluate run-time and memory requirements for aggregating MRSMaps from single as well as multiple images. Single-image maps are created at low run-times on a standard multi-core CPU. While a map of a single image requires more memory than the RGB-D image itself, the strength of the representation becomes apparent, if multiple images are stored in one map. We achieve good compression rate, while the run-time for adding an image barely increases with the number of integrated images. Hence, our representation is well suitable for multi-view models of scenes and objects.

Our first registration method assumes the viewed scene to be rigid. It is directly applicable for visual odometry. Maps are aligned in a dual refinement process that alternates between surfel association and probabilistic pose optimization. We exploit the multi-resolution structure of the maps for efficient association. While associations are made on all available resolutions to correct coarse as well as fine misalignments, we find the finest common resolution between the maps. We consider the matching in position, color, and shape-texture descriptor. Each component contributes to improve the basin of convergence of the registration. We compare our registration method with other approaches on a benchmark dataset, and demonstrate state-of-the-art results in run-time, accuracy, and robustness compared to other dense methods. Sparse interest point matching could well complement our approach in scenes, in which mostly far

and noise-affected depth is measured.

We apply rigid registration for learning 3D models of scenes and objects. Registration yields the camera motion between key views onto the scenes and objects. We extract key views from RGB-D video and process the video online by keeping track of the camera motion in the latest image through registration towards a reference key view. The registration results are spatial constraints between the view poses of the key views. We optimize for the view poses in a pose graph SLAM approach. We additionally register key views that are not in temporal sequence. An efficient hypothesis-and-test schedule allows for online SLAM. From the optimized poses, we overlay the key views in dense MRSMap models.

Object models are detected and tracked in real-time using registration. We combine the accuracy of our registration method with the robustness of particle filtering in a real-time capable tracking approach. For object detection, we propose a multi-resolution surfel-pair voting algorithm that detects objects and estimates their pose efficiently at high recall rates. We integrate object detection with tracking in a joint framework that initializes the filter with coarse pose estimates through detection. It also recovers from situations, in which tracking cannot be resumed, through reinitialization. We equip robots with these approaches to perceive objects for the execution of mobile manipulation tasks. These applications have been publicly demonstrated at RoboCup@Home competitions in 2011, 2012, and 2013. The demonstrations were well received by juries and were important contributions to winning the competitions.

We extend our rigid registration method for aligning and segmenting maps of dynamic scenes in which the moving parts are rigid. We propose a general EM framework for dense 3D motion segmentation for this purpose. A CRF models the likelihood of observing parts under the motion of the segments, while enforcing spatial coherence. We propose approximations based on graph cuts and variational mean fields to gain efficiency. Our approach finds the number of segments and estimates the rigid body motion of the segments. In experiments, we demonstrate high accuracy in segmentation and motion estimates, also under real-time constraints.

This rigid multi-body registration method is used to discover the moving objects in a dynamic scene. We integrate motion segmentation with our key view based SLAM approach. Now, pairs of key views are segmented for motion. The segments are related to each other in order to determine which objects a segment is observing. The motion estimates of the segments yield spatial constraints between the view poses of segments onto the objects, which are optimized through pose graph SLAM. Dense models of the singularized objects can then be retrieved by overlaying the segments from their estimated view poses. By observing objects split and merge, we infer hierarchical part relations between the objects in an unsupervised way.

Non-rigid deformations are recovered with our deformable registration me-

thod. Our approach extends the CPD method for registering RGB-D measurements. It exploits the multi-resolution structure of our maps to achieve run-time efficiency. We propose means to estimate the local 6-DoF transformation between maps at arbitrary points on the object surface from the estimated displacement field. We evaluate accuracy and run-time efficiency of our approach. It is superior in efficiency to aligning the raw RGB-D images using the CPD approach. If images are registered towards persistent models, we can precompute a significant part of the workload. Our method then achieves frame-rates between 1 to 5 Hz on a CPU. This facilitates applications in which deformations are to be estimated at high frame rate, e.g., to track hands or deformable objects such as clothing.

Objects with the same function frequently share a common topology of functional parts. We employ deformable registration to establish shape correspondences between objects and interpret these as correspondences between the functional parts. This allows robots to transfer object manipulation skills to novel, previously unseen objects of a known class of objects. We propose to define the object manipulation skills in terms of grasp poses and motions relative to specific reference frames such as tool end-effectors. These poses and frames are transfered to new objects using local deformations estimated from the displacement field between the object shapes. We also demonstrate this approach publicly at the RoboCup@Home competition 2013.

To recognize objects by semantic categories, we train RFs to segment RGB-D images into object classes. We fuse the semantic segmentations from multiple view points in semantic maps. This makes object-class knowledge persistent for a robot, e.g., to reason about tasks. Mapping is conducted online in real-time due to a highly efficient implementation of the RF classifier on GPU and our real-time capable SLAM approach that is executed in parallel on the CPU. Our approach achieves state-of-the-art results for RGB-D object-class segmentation.

Outlook and Future Work. The methods presented in this thesis open several directions for future research. One path is to transfer our approaches to different modalities of sensors and algorithms that provide dense depth. For instance, MRSMaps could be used to represent dense depth reconstructed with stereo cameras. Existing approaches such as in (Geiger et al., 2010) already provide good quality depth at high frame rate on CPUs. Recently, Newcombe et al. (2011b) proposed DTAM, which estimates dense depth of key frames from images of a moving monocular camera. These key frames could be consistently aligned with our SLAM approach to acquire scene and object models. In (Schadler et al., 2013), we recently applied SLAM with MRSMaps for mapping with 3D laser scanners. We propose a particle filter that localizes the robot based on odometry and matching 2D laser scan lines to a MRSMap, while the laser is continuously rotating. Also, our general framework for dense 3D motion seg-

mentation or principles of deformable registration could be applied to RGB-D images of other sensor systems.

We investigate the perception methods in this thesis to advance the development of intelligent robots that act autonomously in everyday environments. Such environments are highly complex and multifaceted in their appearance and composition. We can devise ever new types of objects and reconfigure our environments in nearly endless variations. To handle this complexity, approaches are promising that learn about the environment in unsupervised ways and generalize knowledge to novel situations. This thesis provides foundations for future research in these directions.

Our approach to unsupervised object discovery from motion cues could be used for interactive perception of objects by robots. It enables robots to singularize objects through manipulating the objects. The robot could explore its environment and novel objects to understand their composition into parts. Co-occurence hints could be combined with motion to generate hypotheses about single objects which are then tested through moving the objects. Once the objects are singularized, commonalities in their appearance and geometry give hints to categorize the objects in an unsupervised way.

Object-class segmentation could be extended for active learning in which the classifier is adapted online from novel training samples. In this way, object knowledge of a robot could be refined while it interacts with the environment. Beliefs on object categories could be validated either by the robot itself, or through communication with human users. This could enable life-long adaptation of object-class knowledge in changing environments.

We demonstrate the use of deformable registration for object manipulation skill transfer. Research could be invested to further understand how skill knowledge can be generalized to novel objects. This does not only concern the transfer of skills between instances of the same class. Knowledge about the usage of a category of objects could also be applied to similar categories. In this thesis, we assumed the manipulated objects to be rigid, and dynamics to be negligible for the tasks. It is also desirable to consider physical properties of objects such as mass, friction, and elasticity. It is an open research challenge to perceive these properties with robot sensors and to build adequate models.

Acronyms

3D-NDT	3D normal distribution transform.
AIC	Akaike information criterion.
AR	auto-regressive.
ATE	absolute trajectory error.
BA	bundle adjustment.
BIC	Bayesian information criterion.
BP	belief propagation.
CPD	coherent point drift.
CRF	conditional random field.
DoF	degree-of-freedom.
EKF	extended Kalman filter.
EM	expectation-maximization.
FGT	fast Gauss transform.
FPFH	fast point feature histogram.
GICP	generalized iterative closest points.
GMM	Gaussian mixture model.
HMM	hidden Markov model.
ICP	iterative closest points.
IR	infrared.
IRLS	iteratively re-weighted least squares.

KL-divergence	Kullback Leibler divergence.
LBP	loopy belief propagation.
LM	Levenberg-Marquardt.
ML	maximum likelihood.
MLN	Markov logic network.
MN	Markov network.
MRF	Markov random field.
MRSMap	multi-resolution surfel map.
RANSAC	random sample consensus.
RF	random decision forest.
RKHS	reproducing kernel Hilbert space.
RMSE	root mean squared error.
RPE	relative pose error.
SDF	signed distance function.
SfM	structure-from-motion.
SIFT	scale-invariant feature transform.
SLAM	simultaneous localization and mapping.
SLAMMOT	simultaneous localization mapping and moving object tracking.
SMOSLAM	simultaneous motion segmentation, localization, and mapping.
surfel	surface element.
SVD	singular value decomposition.
TPS-RPM	thin plate spline robust point matching.
voxel	volume element.

List of Figures

Bibliography

M. Agrawal, K. Konolige, and L. Iocchi. Real-time detection of independent motion using stereo. In *Proceedings of the IEEE Workshop on Motion*, 2005.

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974. ISSN 0018-9286. doi: 10.1109/TAC.1974.1100705.

A. Aldoma, M. Vincze, N. Blodow, D. Gossow, S. Gedikli, R.B. Rusu, and G. Bradski. CAD-model recognition and 6DOF pose estimation using 3D cues. In *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 585–592, 2011. doi: 10.1109/ICCVW.2011.6130296.

B. Allen, B. Curless, and Z. Popović. The space of human body shapes: reconstruction and parameterization from range scans. *ACM Transations on Graphics*, 22(3):587–594, July 2003. ISSN 0730-0301. doi: 10.1145/882262.882311.

B. Amberg, S. Romdhani, and T. Vetter. Optimal step nonrigid ICP algorithms for surface registration. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2007. doi: 10.1109/CVPR.2007.383165.

D. Anguelov, R. Biswas, D. Koller, B. Limketkai, S. Sanner, and S. Thrun. Learning hierarchical object maps of non-stationary environments with mobile robots. In *Proceedings of the 17th Annual Conference on Uncertainty in AI (UAI)*, 2002.

D. Anguelov, P. Srinivasan, H.-C. Pang, D. Koller, S. Thrun, and J. Davis. The correlated correspondence algorithm for unsupervised registration of nonrigid surfaces. In *Proceedings of the International Conference on Advances in Neural Information Processing (NIPS)*, 2004.

P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):898 –916, 2011.

M. Sanjeev Arulampalam, Simon Maskell, and Neil Gordon. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50:174–188, 2002.

K.S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-D point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(5):698–700, 1987. doi: 10.1109/TPAMI.1987.4767965.

A. Ayvaci and S. Soatto. Motion segmentation with occlusions on the superpixel graph. In *Proceedings of the IEEE ICCV Workshops*, 2009.

D. H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111 – 122, 1981. ISSN 0031-3203. doi: http://dx.doi.org/10.1016/0031-3203(81)90009-1.

R. C. Batra. *Elements of Continuum Mechanics*. AIAA education series. American Institute of Aeronautics and Astronautics, 2006. ISBN 9781600860485.

H. Bay, T. Tuytelaars, and L. Van Gool. SURF: speeded up robust features. *Computer Vision - ECCV*, 2006.

J. Bergstra, R. Bardenet, Y. Bengio, B. Kégl, et al. Algorithms for hyper-parameter optimization. In *Proceedings of the 25th Annual Conference on Neural Information Processing Systems (NIPS 2011)*, 2011. URL `https://github.com/jaberg/hyperopt`.

P. J. Besl and N. D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 14(2): 239–256, 1992.

A. Billard, S. Calinon, R. Dillmann, and S. Schaal. Robot programming by demonstration. In Bruno Siciliano and Oussama Khatib, editors, *Springer Handbook of Robotics*, pages 1371–1394. Springer Berlin Heidelberg, 2008.

N. Biresev. Semantic mapping using object-class segmentation of RGB-D images. Master's thesis, Autonomous Intelligent Systems Group, Computer Science Institute VI, University of Bonn, 2012.

C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0387310738.

M. Bokeloh, A. Berner, M. Wand, A. Schilling, and H.-P. Seidel. Slippage features. Technical Report WSI-2008-03, Wilhelm Schickard Institut, University of Tübingen, Tübingen, Germany, 2008.

Y. Boykov and M.-P. Jolly. Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images. In *Proceedings of the IEEE International Conference on Computer Vision*, 2001.

Y. Boykov and O. Veksler. Graph Cuts in Vision and Graphics: Theories and Applications. In Nikos Paragios, Yunmei Chen, and Olivier Faugeras, editors, *Handbook of Mathematical Models in Computer Vision*, chapter 5, pages 79–96. Springer US, New York, 2006. ISBN 0-387-26371-3. doi: 10.1007/0-387-28831-7_5. URL `http://dx.doi.org/10.1007/0-387-28831-7_5`.

Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:2001, 2001.

L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and Regression Trees*. Chapman & Hall/CRC, 1984.

T. Brox, A. Bruhn, and J. Weickert. Variational motion segmentation with level sets. In *Proceedings of the Europ. Conference on Computer Vision (ECCV)*, Lecture Notes in Computer Science, pages 471–483. 2006.

E. Bylow, J. Sturm, C. Kerl, F. Kahl, and D. Cremers. Real-time camera tracking and 3D reconstruction using signed distance functions. In *Proceedings of Robotics: Science and Systems Conference (RSS)*, 2013.

J. Carreira and C. Sminchisescu. CPMC: Automatic object segmentation using constrained parametric min-cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34:1312–1328, 2012.

R. O. Castle, G. Klein, and D. W. Murray. Combining monoSLAM with object recognition for scene augmentation using a wearable camera. *Image Vision Computing*, 28(11):1548 – 1556, 2010.

A. Censi. An accurate closed-form estimate of ICP's covariance. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3167–3172, 2007.

T. F. Chan and J. G. Lewis. Computing standard deviations: accuracy. *Communications of the ACM*, 22(9):526–531, September 1979.

T. F. Chan, G. H. Golub, and R. J. LeVeque. Updating formulae and a pairwise algorithm for computing sample variances. Technical report, Stanford, CA, USA, 1979.

Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image Vision Computing*, 10(3):145–155, 1992.

Z. Chen and S. Haykin. On different facets of regularization theory. *Neural Computation*, 14(12):2791–2846, 2002.

C. Choi and Henrik I Christensen. Robust 3D visual tracking using particle filtering on the SE(3) group. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

C. Choi and H.I. Christensen. 3D pose estimation of daily objects using an RGB-D camera. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012a.

C. Choi and H.I. Christensen. 3D textureless object detection and tracking: An edge-based approach. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3877–3884, 2012b. doi: 10.1109/IROS.2012.6386065.

C. Choi, Y. Taguchi, O. Tuzel, M.-Y. Liu, and S. Ramalingam. Voting-based pose estimation for robotic assembly using a 3D sensor. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

H. Chui and A. Rangarajan. A new point matching algorithm for non-rigid registration. *Computer Vision and Image Understanding*, 89(2-3):114–141, February 2003. ISSN 1077-3142. doi: 10.1016/S1077-3142(03)00009-2.

J. Civera, D. Galvez-Lopez, L. Riazuelo, D. Tardos, and J. M. M. Montiel. Towards semantic SLAM using a monocular camera. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2011.

A. I. Comport, É. Marchand, and F. Chaumette. Robust model-based tracking for robot vision. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2004.

W. J. Cook, W. H. Cunningham, W. R. Pulleyblank, and A. Schrijver. *Combinatorial optimization*. John Wiley & Sons, Inc., New York, NY, USA, 1998. ISBN 0-471-55894-X.

C. Couprie, C. Farabet, L. Najman, and Y. LeCun. Indoor semantic segmentation using depth information. *The Computing Resource Repository (CoRR)*, abs/1301.3572, 2013.

D. Cremers and S. Soatto. Motion competition: A variational approach to piecewise parametric motion segmentation. *International Journal of Computer Vision*, 62:249–265, 2005.

D. Cremers, M. Rousson, and R. Deriche. A review of statistical approaches to level set segmentation: Integrating color, texture, motion and shape. *International Journal of Computer Vision*, 72:195–215, 2007.

B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, pages 303–312, New York, NY, USA, 1996. ACM. ISBN 0-89791-746-4. doi: 10.1145/237170.237269.

T. A. Davis. *Direct Methods for Sparse Linear Systems (Fundamentals of Algorithms)*. Society for Industrial and Applied Mathematic, 2006. ISBN 0898716136.

A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. MonoSLAM: Real-time single camera SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 29(6):1052–1067, 2007.

A. Delong, A. Osokin, H. N. Isack, and Y. Boykov. Fast approximate energy minimization with label costs. *International Journal of Computer Vision*, 96 (1):1–27, 2012.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

A. Doucet, S. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Statistics and Computing*, 10(3):197–208, 2000. ISSN 0960-3174. doi: 10.1023/A:1008935410038.

D. Droeschel, S. May, D. Holz, P. Ploeger, and S. Behnke. Robust ego-motion estimation with ToF cameras. In *Proceedings of the 4th European Conference on Mobile Robots (ECMR)*, 2009.

B. Drost, M. Ulrich, N. Navab, and S. Ilic. Model globally, match locally: Efficient and robust 3D object recognition. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

T. Drummond and R. Cipolla. Real-time visual tracking of complex structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 24 (7):932–946, 2002.

J. Elseberg, D. Borrmann, and A. Nüchter. One billion points in the cloud - an octree for efficient processing of 3D laser scans. *ISPRS Journal of Photogrammetry and Remote Sensing*, 76(0):76 – 88, 2013. doi: http://dx.doi.org/10.1016/j.isprsjprs.2012.10.004.

F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard. An evaluation of the RGB-D SLAM system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

N. Engelhard, F. Endres, J. Hess, J. Sturm, and W. Burgard. Real-time 3D visual SLAM with a hand-held camera. In *Proceedings of RGB-D Workshop on 3D Perception in Robotics at European Robotics Forum*, 2011.

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2), 2010.

P. Fitzpatrick. First contact: an active vision approach to segmentation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2003.

A. Fix, A. Gruber, E. Boros, and R. Zabih. A graph cut algorithm for higher-order markov random fields. In *Proceedings of the 2011 International Conference on Computer Vision (ICCV)*, pages 1020–1027, Washington, DC, USA, 2011. IEEE Computer Society. ISBN 978-1-4577-1101-5. doi: 10.1109/ICCV.2011.6126347. URL `http://dx.doi.org/10.1109/ICCV.2011.6126347`.

B. Fornberg and J. Zuev. The runge phenomenon and spatially variable shape parameters in RBF interpolation. *Computers & Mathematics with Applications*, 54(3):379 – 398, 2007. doi: http://dx.doi.org/10.1016/j.camwa.2007.01.028.

D.M. Gavrila and V. Philomin. Real-time object detection for smart vehicles. In *Proceedings of the 7th International Conference on Computer Vision (ICCV)*, volume 1, pages 87–93, 1999. doi: 10.1109/ICCV.1999.791202.

A. Geiger, M. Roser, and R. Urtasun. Efficient large-scale stereo matching. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, 2010.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741, 1984. ISSN 0162-8828. doi: 10.1109/TPAMI.1984.4767596.

M. G. Genton. Classes of kernels for machine learning: a statistics perspective. *Journal of Machine Learning Research*, 2:299–312, March 2002. ISSN 1532-4435.

L. Greengard and J. Strain. The fast Gauss transform. *SIAM Journal on Scientific and Statistical Computing*, 12(1):79–94, 1991. doi: 10.1137/0912004.

G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with Rao-Blackwellized particle filters. *IEEE Transactions on Robotics*, 23(1):34–46, 2007.

A. Gruber and Y. Weiss. Multibody factorization with uncertainty and missing data using the EM algorithm. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

W. W. Hager and H. Zhang. A survey of nonlinear conjugate gradient methods. *Pacific Journal of Optimization*, 2(1):35–58, 2006.

D. Hähnel, R. Triebel, W. Burgard, and S. Thrun. Map building with mobile robots in dynamic environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.

A. Hanbury. Constructing cylindrical coordinate colour spaces. *Pattern Recognition Letters*, 29(4):494–500, March 2008. ISSN 0167-8655. doi: 10.1016/j.patrec.2007.11.002. URL `http://dx.doi.org/10.1016/j.patrec.2007.11.002`.

C. Harris. Tracking with rigid models. In *Active vision*, pages 59–73. MIT Press, 1993.

P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments. *The International Journal of Robotics Research*, 31(5):647–663, 2012.

M. Herbert, C. Caillas, E. Krotkov, I. S. Kweon, and T. Kanade. Terrain mapping for a roving planetary explorer. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 997–1002, 1989. doi: 10.1109/ROBOT.1989.100111.

E. Herbst, X. Ren, and D. Fox. RGB-D object discovery via multi-scene analysis. In *Proceedings of the IEEE International Conference on Robots and Systems (IROS)*, pages 4850–4856, 2011.

E. Herbst, X. Ren, and D. Fox. RGB-D flow: Dense 3-D motion estimation using color and depth. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit. Gradient response maps for real-time detection of texture-less objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

D. Holz and S. Behnke. Fast range image segmentation and smoothing using approximate surface reconstruction and region growing. In *Proceedings of the 12th International Conference on Intelligent Autonomous Systems (IAS)*, Jeju Island, Korea, June 2012.

D. Holz, S. Holzer, R. B. Rusu, and S. Behnke. Real-time plane segmentation using RGB-D cameras. In *Proceedings of the 15th RoboCup International Symposium*, volume 7416 of *Lecture Notes in Computer Science*, pages 307–317. Springer, July 2011.

A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: an efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 34:189–206, 2013.

A. Howard. Real-time stereo visual odometry for autonomous ground vehicles. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3946–3952, 2008. doi: 10.1109/IROS.2008.4651147.

A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy. Visual odometry and mapping for autonomous flight using an RGB-D camera. In *Proceedings of the International Symposium on Robotics Research (ISRR)*, 2011.

F. Huguet and F. Devernay. A variational method for scene flow estimation from stereo sequences. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2007.

B. Huhle, Martin Magnusson, W. Strasser, and A.J. Lilienthal. Registration of colored 3D point clouds with a kernel-based extension to the normal distributions transform. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4025–4030, 2008. doi: 10.1109/ROBOT.2008.4543829.

B. Jian and B. C. Vemuri. Robust point set registration using Gaussian mixture models. *IEEE Transations on Pattern Analysis and Machine Intelligence*, 33 (8):1633–1645, 2011.

A. Johnson. *Spin-Images: A Representation for 3-D Surface Matching*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, August 1997.

S. J. Julier and J. K. Uhlmann. A new extension of the Kalman filter to nonlinear systems. In *Proceedings of the 11th International Symposium on Aerospace/Defense Sensing (AeroSense), Simulations and Controls*, 1997.

D. Katz, M. Kazemi, J. A. Bagnell, and A. Stentz. Interactive segmentation, tracking, and kinematic modeling of unknown articulated objects. Technical report, Carnegie Mellon Robotics Institute, March 2012.

C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations*. Number 16 in Frontiers in Applied Mathematics. SIAM, 1995.

C. T. Kelley. Iterative Methods for Optimization. *Frontiers in Applied Mathematics*, 18, 1999.

J. Kenney, T. Buckley, and O. Brock. Interactive segmentation for manipulation in unstructured environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2009.

C. Kerl, J. Sturm, and D. Cremers. Dense visual slam for rgb-d cameras. In *Proceedings of the International Conference on Intelligent Robot Systems (IROS)*, 2013.

K. Khoshelham and S. O. Elberink. Accuracy and resolution of Kinect depth data for indoor mapping applications. *Sensors*, 12(2):1437–1454, 2012. ISSN 1424-8220. doi: 10.3390/s120201437. URL `http://www.mdpi.com/1424-8220/12/2/1437`.

E. Kim and G. Medioni. 3D object recognition in range images using visibility context. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3800–3807, 2011.

G. Klein and D. Murray. Full-3D edge tracking with a particle filter. In *British Machine Vision Conference*, pages 1119–1128, 2006.

G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *Proceedings of IEEE/ACM International Symp. on Mixed and Augmented Reality (ISMAR)*, pages 225–234, 2007.

V. Kolmogorov and C. Rother. Minimizing non-submodular functions with graph cuts - a review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(7):1274–1279, July 2007. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1031. URL `http://dx.doi.org/10.1109/TPAMI.2007.1031`.

V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26:65–81, 2004.

K. Konolige, J. Bowman, J.D. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua. View-based maps. *The International Journal of Robotics Research*, 29(8):941–957, 2010.

M. Krainin, P. Henry, X. Ren, and D. Fox. Manipulator and object tracking for in-hand 3D object modeling. *The International Journal of Robotics Research*, 30(11), 2011.

F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001. ISSN 0018-9448. doi: 10.1109/18.910572.

R. Kuemmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. G2o: A general framework for graph optimization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3607–3613, 2011.

M. P. Kumar, P. H. S. Torr, and A. Zisserman. Learning layered motion segmentations of video. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2005.

J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-778-1.

K. Lai, L. Bo, X. Ren, and D. Fox. A scalable tree-based approach for joint object and pose recognition. In *Proceedings of the 25th Conference on Artificial Intelligence (AAAI)*, August 2011.

K. Lai, L. Bo, X. Ren, and D. Fox. Detection-based object labeling in 3D scenes. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1330–1337, 2012.

Y. Lamdan and H.J. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. In *Proceedings of the 2nd International Conference on Computer Vision*, pages 238–249, 1988. doi: 10.1109/CCV.1988.589995.

V. Lepetit and P. Fua. *Monocular-Based 3D Tracking of Rigid Objects*. Now Pub, 2005.

V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 28(9): 1465–1479, 2006.

H. Li, R. W. Sumner, and M. Pauly. Global correspondence optimization for non-rigid registration of depth scans. *Computer Graphics Forum (Proceedings SGP'08)*, 27(5), July 2008.

L.-J. Li, R. Socher, and L. Fei-Fei. Towards total scene understanding: Classification, annotation and segmentation in an automatic framework. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, (2):91, 2004.

K. Madsen, H. B. Nielsen, and O. Tingleff. Methods for non-linear least squares problems (2nd ed.), 2004.

M. Magnusson, T. Duckett, and A. J. Lilienthal. Scan registration for autonomous mining vehicles using 3D-NDT. *Journal of Field Robotics*, 24(10): 803–827, 2007.

M. Martinez, A. Collet, and S. S. Srinivasa. MOPED: A scalable and low latency object recognition and pose estimation system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2043–2049, 2010.

S. May, S. Fuchs, D. Droeschel, D. Holz, and A. Nüchter. Robust 3D-mapping with time-of-flight cameras. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1673–1678, October 2009.

M. McElhone. Model, match, vote and track: 6-DoF pose filtering with multi-resolution surfel maps. Master's thesis, Autonomous Intelligent Systems Group, Computer Science Institute VI, University of Bonn, 2013.

D. Meger, P.-E. Forssén, K. Lai, S. Helmer, S. McCann, T. Southey, M. Baumann, J. J. Little, and D. G. Lowe. Curious George: An attentive semantic robot. *Robotics and Autonomous Systems*, 56(6):503–511, 2008.

E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd. Real time localization and 3D reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 363–370, 2006. doi: 10.1109/CVPR.2006.236.

A. Myronenko. *Non-rigid Image Registration: Regularization, Algorithms and Applications*. PhD thesis, Oregon Health & Science University (OHSU), School of Medicine, Department of Science and Engineering (OGI), 2010.

A. Myronenko and Xubo Song. Point set registration: Coherent point drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12): 2262–2275, 2010. ISSN 0162-8828. doi: 10.1109/TPAMI.2010.46.

P. Kohli N. Silberman, D. Hoiem and R. Fergus. Indoor segmentation and support inference from RGBD images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.

Y. Nesterov. *Introductory lectures on convex optimization : a basic course*. Applied optimization. Kluwer Academic Publ., Boston, Dordrecht, London, 2004. ISBN 1-4020-7553-7. URL `http://opac.inria.fr/record=b1104789`.

R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: real-time dense surface mapping and tracking. In *Proceedings of the 10th International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 127–136, 2011a.

R. A. Newcombe, S. Lovegrove, and A. J. Davison. DTAM: Dense tracking and mapping in real-time. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 2320–2327, 2011b.

T. S. Newman and H. Yi. A survey of the marching cubes algorithm. *Computers & Graphics*, 30(5):854–879, 2006.

D. Nister. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 26(6): 756–770, 2004. doi: 10.1109/TPAMI.2004.17.

D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. In *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 652–659, 2004.

A. Nüchter and J. Hertzberg. Towards semantic maps for mobile robots. *Robotics and Autonomous Systems*, 56(11):915–926, 2008.

A. Nuechter, K. Lingemann, J. Hertzberg, and H. Surmann. 6D SLAM with approximate data association. In *Proceedings of the International Conference on Advanced Robotics (ICAR)*, pages 242–249, 2005.

J. Ohtsubo and T. Asakura. Statistical properties of laser speckle produced in the diffraction field. *Applied Optics*, 16(6):1742–1753, Jun 1977. doi: 10.1364/AO.16.001742.

C.F. Olson and D.P. Huttenlocher. Automatic target recognition by matching oriented edge pixels. *IEEE Transactions on Image Processing*, 6(1):103–113, 1997. ISSN 1057-7149. doi: 10.1109/83.552100.

C. Papazov, S. Haddadin, S. Parusel, K. Krieger, and D. Burschka. Rigid 3D geometry matching for grasping of known objects in cluttered scenes. *International Journal of Robotics Research (IJRR)*, 31, April 2012.

P. Pfaff, R. Triebel, and W. Burgard. An efficient extension to elevation maps for outdoor terrain mapping and loop closing. *International Journal of Robotics Research*, 26(2):217–230, February 2007. ISSN 0278-3649. doi: 10.1177/0278364906075165.

S. Ramalingam, P. Kohli, K. Alahari, and P. H. S. Torr. Exact inference in multi-label CRFs with higher order cliques. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008. doi: 10.1109/CVPR.2008.4587401.

F. Ramos, D. Fox, and H. Durrant-Whyte. CRF-Matching: Conditional random fields for feature-based scan matching. In *Proceedings of Robotics: Science and Systems (RSS)*, 2007.

A. Ranganathan and F. Dellaert. Semantic modeling of places using objects. In *Proceedings of Robotics: Science and Systems (RSS)*, 2007.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2005. ISBN 026218253X.

D. Raviv, A.M. Bronstein, M.M. Bronstein, R. Kimmel, and N. Sochen. Affine-invariant diffusion geometry for the analysis of deformable 3D shapes. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2361–2367, 2011. doi: 10.1109/CVPR.2011.5995486.

M. Richardson and P. Domingos. Markov logic networks. *Journal of Machine Learning*, 62(1-2):107–136, February 2006. doi: 10.1007/s10994-006-5833-1.

D. Ross, D. Tarlow, and R. Zemel. Learning articulated structure and motion. *International Journal of Computer Vision*, 88:214–237, 2010.

H. Roth and M. Vona. Moving volume KinectFusion. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2012.

F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. Segmenting, modeling, and matching video clips containing multiple moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 477–491, 2007.

A. Roussos, C. Russell, R. Garg, and L. de Agapito. Dense multibody motion estimation and reconstruction from a handheld camera. In *Proceedings of the IEEE International Symp. on Mixed and Augmented Reality (ISMAR)*, 2012.

M. Ruhnke, B. Steder, G. Grisetti, and W. Burgard. Unsupervised learning of 3D object models from partial views. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2009.

R. B. Rusu, M. Beetz, Z. C. Marton, N. Blodow, and M. Dolha. Towards 3D point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 2008.

R. B. Rusu, N. Blodow, and M. Beetz. Fast Point Feature Histograms (FPFH) for 3D Registration. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3212–3217, 2009.

R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu. Fast 3D recognition and pose using the viewpoint feature histogram. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2155–2162, 2010. doi: 10.1109/IROS.2010.5651280.

J. Ryde and J. J. Corso. Fast voxel maps with counting bloom filters. In *Proceedings of the IEEE International Conference on Robots and Systems (IROS)*, pages 4413–4418. IEEE, 2012.

J. Ryde and H. Hu. 3d mapping with multi-resolution occupied voxel lists. *Autonomous Robots*, 28:169 – 185, 2010.

R. Sagawa, K. Akasaka, Y. Yagi, H. Hamer, and L. Van Gool. Elastic convolved icp for the registration of deformable objects. In *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1558–1565, 2009. doi: 10.1109/ICCVW.2009.5457428.

Y. Sahillioglu and Y. Yemez. Coarse-to-fine combinatorial matching for dense isometric shape correspondence. *Computer Graphics Forum*, 30(5):1461–1470, 2011.

Y. Sahillioglu and Y. Yemez. Minimum-distortion isometric shape correspondence using em algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2203–2215, 2012.

M. Saito, T. Okatani, and K. Deguchi. Application of the mean field methods to MRF optimization in computer vision. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1680–1687, 2012. doi: 10.1109/CVPR.2012.6247862.

R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. J. Kelly, and A. J. Davison. SLAM++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

Z. Santa and Z. Kato. Elastic registration of 3D deformable objects. In *Proceedings of the International Conference on Digital Image Computing Techniques and Applications (DICTA)*, pages 1–7, 2012. doi: 10.1109/DICTA.2012.6411674.

M. Schadler, J. Stückler, and S. Behnke. Multi-resolution surfel mapping and real-time pose tracking using a continuously rotating 3D laser scanner. In *Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, 2013.

K. Schindler and D. Suter. Two-view multibody structure-and-motion with outliers through model selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:983–995, 2006. ISSN 0162-8828.

R. Schnabel, R. Wessel, R. Wahl, and R. Klein. Shape recognition in 3D point-clouds. In V. Skala, editor, *Proceedings of the 16th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*. UNION Agency-Science Press, February 2008. ISBN 978-80-86943-15-2.

B. Schölkopf, R. Herbrich, and A. J. Smola. A generalized representer theorem. In *Proceedings of the Annual Conference on Computational Learning Theory*, pages 416–426, 2001.

J. Schulman, A. Gupta, S. Venkatesan, M. Tayson-Frederick, and P. Abbeel. A case study of trajectory transfer through non-rigid registration for a simplified suturing scenario. In *Proceedings of the 26th IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013a.

J. Schulman, J. H., C. Lee, and P. Abbeel. Learning from demonstrations through the use of non-rigid registration. In *Proceedings of the 16th International Symposium on Robotics Research (ISRR)*, 2013b.

G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6 (2):461–464, 1978. ISSN 00905364. doi: 10.2307/2958889.

S. Se, D. Lowe, and J. Little. Vision-based mobile robot localization and mapping using scale-invariant features. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2051–2058, 2001.

A. Segal, D. Haehnel, and S. Thrun. Generalized-ICP. In *Proceedings of Robotics: Science and Systems (RSS)*, 2009.

H. Sekkati and A. Mitiche. Concurrent 3-D motion segmentation and 3-D interpretation of temporal sequences of monocular images. *IEEE Transactions on Image Processing*, 15(3):641–653, 2006.

S. Sengupta, E. Greveson, A. Shahrokni, and P.H.S. Torr. Semantic modelling of urban scenes. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1297–1304. IEEE, 2011.

N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from RGBD images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.

A. J. Smola, B. Schölkopf, and K.-R. Müller. The connection between regularization operators and support vector kernels. *Neural Networks*, 11(4):637–649, June 1998. ISSN 0893-6080. doi: 10.1016/S0893-6080(98)00032-X.

F. Steinbruecker, J. Sturm, and D. Cremers. Real-time visual odometry from dense RGB-D images. In *Proceedings of the ICCV Workshop on Live Dense Reconstruction with Moving Cameras*, pages 719–722, 2011.

F. Steinbruecker, C. Kerl, J. Sturm, and D. Cremers. Large-scale multiresolution surface reconstruction from RGB-D sequences. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2013.

T. Stoyanov, M. Magnusson, H. Andreasson, and A. J. Lilienthal. Fast and accurate scan registration through minimization of the distance between compact 3D NDT representations. *The International Journal of Robotics Research*, 31 (12):1377–1393, 2012.

J. Stückler and S. Behnke. Combining depth and color cues for scale- and viewpoint-invariant object segmentation and recognition using random forests. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010.

J. Stückler, N. Biresev, and S. Behnke. Semantic mapping using object-class segmentation of RGB-D images. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2012a.

J. Stückler, D. Droeschel, K. Gräve, D. Holz, J. Kläß, M. Schreiber, R. Steffens, and S. Behnke. Towards robust mobility, flexible object manipulation, and intuitive multimodal interaction for domestic service robots. In *RoboCup 2011: Robot Soccer World Cup XV*, Lecture Notes in Computer Science. 2012b.

J. Stückler, D. Holz, and S. Behnke. RoboCup@Home: Demonstrating everyday manipulation skills in RoboCup@Home. *IEEE Robotics & Automation Magazine*, 19(2):34–42, June 2012. ISSN 1070-9932. doi: 10.1109/MRA.2012.2191993.

J. Stückler, I. Badami, D. Droeschel, K. Gräve, D. Holz, M. McElhone, M. Nieuwenhuisen, M. Schreiber, M. Schwarz, and S. Behnke. NimbRo@Home: Winning team of the RoboCup@Home competition 2012. In *RoboCup 2012: Robot Soccer World Cup XVI*, Lecture Notes in Computer Science. 2013.

J. Stückler, D. Droeschel, K. Gräve, D. Holz, M. Schreiber, A. Topalidou-Kyniazopoulou, M. Schwarz, and S. Behnke. Increasing flexibility of mobile manipulation and intuitive human-robot interaction in RoboCup@Home. In *RoboCup 2013: Robot Soccer World Cup XVII*, Lecture Notes in Computer Science. 2014. accepted for publication.

J. Stuehmer, S. Gumhold, and D. Cremers. Real-time dense geometry from a handheld camera. In *Proceedings of the 32nd DAGM Symposium*, pages 11–20, 2010.

J. Sturm, C. Stachniss, and W. Burgard. A probabilistic framework for learning kinematic models of articulated objects. *Journal on Artificial Intelligence Research (JAIR)*, 41:477–626, 2011.

J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *Proceedings of the International Conference on Intelligent Robot Systems (IROS)*, 2012.

J. Sun, M. Ovsjanikov, and L. Guibas. A concise and provably informative multi-scale signature based on heat diffusion. In *Proceedings of the Symposium on Geometry Processing*, pages 1383–1392. Eurographics Association, 2009.

M. Tenorth, S. Profanter, F. Balint-Benczedi, and M. Beetz. Decomposing CAD models of objects of daily use and reasoning about their functional parts. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

A. Tevs, M. Bokeloh, M. Wand, A. Schilling, and H.-P. Seidel. Isometric registration of ambiguous and partial data. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1185–1192, 2009. doi: 10.1109/CVPR.2009.5206775.

S. Thrun. Robotic mapping: A survey. In *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, 2005. ISBN 0262201623.

A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-Posed Problems*. V. H. Winston & Sons, Washington, D.C.: John Wiley & Sons, New York„ 1977.

G. D. Tipaldi and F. Ramos. Motion clustering and estimation with conditional random fields. In *Proceedings of the IEEE/RSJ International Conference on IROS*, 2009.

F. Tombari, S. Salti, and L. Stefano. Unique signatures of histograms for local surface description. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Proceedings of the European Conference on Computer Vision (ECCV)*, volume 6313 of *Lecture Notes in Computer Science*, pages 356–369. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-15557-4. doi: 10.1007/978-3-642-15558-1_26.

F. Tombari, S. Salti, and L. Di Stefano. A combined texture-shape descriptor for enhanced 3D feature matching. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 809–812, 2011. doi: 10.1109/ICIP.2011.6116679.

M. Tomono and Y. Shin'ichi. Object-based localization and mapping using loop constraints and geometric prior knowledge. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2003.

R. Triebel, P. Pfaff, and W. Burgard. Multi-level surface maps for outdoor terrain mapping and loop closing. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

H. Uchiyama and E. Marchand. Object detection and pose tracking for augmented reality: Recent approaches. In *Proceedings of the 18th Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV)*, 2012.

M. Unger, M. Werlberger, T. Pock, and H. Bischof. Joint motion estimation and segmentation of complex scenes with label costs and occlusion modeling. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1878 –1885, 2012.

L. Vacchetti, V. Lepetit, and P. Fua. Combining edge and texture information for real-time accurate 3D camera tracking. In *Proceedings of IEEE/ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, 2004.

J. Van de Ven, F. Ramos, and G.D. Tipaldi. An integrated probabilistic model for scan-matching, moving object detection and motion estimation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2010.

M. Van den Bergh and L. van Gool. Real-time stereo and flow-based video segmentation with superpixels. In *IEEE WS on Applications of Computer Vision (WACV)*, 2012.

N. Vaskevicius, A. Birk, K. Pathak, and S. Schwertfeger. Efficient representation in 3D environment modeling for planetary robotic exploration. *Advanced Robotics*, 24(8-9):1169–1197, 2010. doi: 10.1163/016918610X501291.

S. Vasudevan, S. Gächter, V. Nguyen, and R. Siegwart. Cognitive maps for mobile robots-an object based approach. *Robotics and Autonomous Systems*, 55(5):359–371, 2007.

B. Waldvogel. Accelerating random forests on CPUs and GPUs for object-class image segmentation. Master's thesis, Autonomous Intelligent Systems Group, Computer Science Institute VI, University of Bonn, 2013.

M. Wand, B. Adams, M. Ovsjanikov, A. Berner, M. Bokeloh, P. Jenke, L. Guibas, H.-P. Seidel, and A. Schilling. Efficient reconstruction of non-rigid shape and motion from real-time 3D scanner data. *ACM Transactions on Graphics*, 28(2):15:1–15:15, May 2009. ISSN 0730-0301.

C. Wang, C. Thorpe, M. Hebert, S. Thrun, and H. Durrant-whyte. Simultaneous localization, mapping and moving object tracking. *International Journal of Robotics Research*, 2004.

S. Wang, H. Yu, and R. Hu. 3D video based segmentation and motion estimation with active surface evolution. *Journal of Signal Processing Systems*, pages 1–14, 2012.

J. Weber and J. Malik. Rigid body segmentation and shape description from dense optical flow under weak perspective. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:139–143, 1997.

A. Wedel and D. Cremers. *Stereoscopic Scene Flow for 3D Motion Analysis*. 2011.

T. Weise, T. Wismer, B. Leibe, and L. Van Gool. Online loop closure for real-time interactive 3D scanning. *Computer Vision and Image Understanding*, 115(5):635–648, 2011.

H. Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in Computational Mathematics*, 4(1):389–396, 1995. doi: 10.1007/BF02123482.

T. Whelan, H. Johannsson, M. Kaess, J.J. Leonard, and J.B. McDonald. Robust tracking for real-time dense RGB-D mapping with Kintinuous. Technical Report MIT-CSAIL-TR-2012-031, Computer Science and Artificial Intelligence Laboratory, MIT, Sep 2012.

B. Willimon, I. Walker, and S. Birchfield. 3D non-rigid deformable surface estimation without feature correspondence. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

L. Zelnik-Manor, M. Machline, and M. Irani. Multi-body factorization with uncertainty: Revisiting motion consistency. *International Journal of Computer Vision*, 68(1), 2006.

H. Zender, O. Martinez Mozos, P. Jensfelt, G.-J. M. Kruijff, and W. Burgard. Conceptual spatial representations for indoor mobile robots. *Robotics and Autonomous Systems*, 56(6):493 – 502, 2008.

G. Zhang, J. Jia, and H. Bao. Simultaneous multi-body stereo and segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2011.

K. Zhou, M. Gong, X. Huang, and B. Guo. Data-parallel octrees for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 17(5):669–681, 2011.