Inaugural-Dissertation

zur
Erlangung der Doktorwürde
der
Naturwissenschaftlich–Mathematischen Gesamtfakultät
der
Ruprecht–Karls–Universität
Heidelberg

vorgelegt von
Dipl.–Inf. Dirk Breitenreicher
aus Mannheim

Tag der mündlichen Prüfung: 20. April 2010

Model-Based Multiple 3D Object Recognition in Range Data

Gutachter: Prof. Dr. Christoph Schnörr
Prof. Dr. Fred Hamprecht

To my grandparents*.

* Anna Hörner (1924 - 2007)
Alfred Breitenreicher (1929 - 2006)

Zusammenfassung

Bildgestützte Systeme bilden die Grundlage vieler industrieller Anwendungen, zum Beispiel in der Fertigungstechnik, im medizinischen Bereich oder für Service-Roboter. Eine Aufgabe, die all diese Anwendungen gemein haben, ist die Detektion und Lokalisierung von bekannten Objekten in unstrukturierten Bildern. Dies ist jedoch ein "Henne-Ei"-Problem, das daraus besteht, Teile des Bildes Modellteilen zuzuweisen, während simultan die Parameter der Lage des Objektes geschätzt werden müssen.

In dieser Arbeit betrachten wir Computer-Vision Verfahren für die spezielle industrielle Anwendung "Bin-Picking", deren Ziel es ist, die Lage von mehreren bekannten und zufällig angeordneten Objekten zuverlässig zu bestimmen. Auch wenn Vorwissen über die Form der Objekte das Problem vereinfacht, führen Symmetrien, gegenseitige Verdeckung der Objekte, strukturelle Messfehler sowie Laufzeitrestriktionen dazu, dass die Lösung des Problems komplex ist.

Ein gängiger Ansatz, sich dieses Problems anzunehmen, ist eine Zwei-Schritt-Strategie zu verfolgen, die anfangs eine grobe Schätzung der Lage der Objekte bestimmt, gefolgt von einer zusätzlichen Feinpositionierung. Etablierte Initialisierungsverfahren sind jedoch nur in der Lage, die Position einzelner Objekte zu bestimmen. Daher können sie kontextbezogene Restriktionen, die durch mehrere Instanzen verursacht werden, nicht auflösen, was wiederum zu ungenauen Positionierungen führt. Dies hat jedoch zur Folge, dass gängige Feinpositionierungsansätze die genaue Objektlage nicht mehr zuverlässig bestimmen können und das gesamte Verfahren nur ungenaue Resultate erreicht.

In dieser Arbeit schlagen wir einen neuen Ansatz zur groben Registrierung vor, welcher die Lage aller Objekte gleichzeitig bestimmt. Zusätzlich wird eine neue lokale Feinausrichtung erforscht, die einzelne Objektpositionen verfeinert. Dieser Ansatz beseitigt die Mängel gängiger Ansätze und führt zu hinreichend genauen Resultaten für eine Vielzahl von Initialisierungen. Beide Schritte nutzen erweiterte numerische Techniken wie konvexe, large-scale Programmierung und geometrische Optimierung im gekrümmten Raum der Starrkörpertransformationen. Zudem ergänzen sich die Einzelschritte, da sich widersprechende Schätzungen in einem globalen konvexen Problem beseitigt werden und hinreichend gute Initialisierungen im nachfolgenden lokalen, nicht-konvexen Schritt verfeinert werden.

Experimente auf künstlichen und realen Messungen bestätigen den vorgeschlagenen, neuen Ansatz und zeigen, dass das Verfahren robust gegen Messfehler und Verdeckungen ist sowie das Potential hat, die Laufzeitrestriktionen vieler industrieller Anwendungen zu erfüllen.

Abstract

Vision guided systems are relevant for many industrial application areas, including manufacturing, medicine, service robots etc. A task common to these applications consists of detecting and localizing known objects in cluttered scenes. This amounts to solve the *"chicken and egg"* problem consisting of data assignment and parameter estimation, that is to localize an object and to determine its pose.

In this work, we consider computer vision techniques for the special scenario of industrial bin-picking applications where the goal is to accurately estimate the positions of multiple instances of arbitrary, known objects that are randomly assembled in a bin. Although a-priori knowledge of the objects simplifies the problem, model symmetries, mutual occlusion as well as noise, unstructured measurements and run-time constraints render the problem far from being trivial.

A common strategy to cope with this problem is to apply a two-step approach that consists of rough initialization estimation for each objects' position followed by subsequent refinement steps. Established initialization procedures only take into account single objects, however. Hence, they cannot resolve contextual constraints caused by multiple object instances and thus yield poor estimates of the objects' pose in many settings. Inaccurate initial configurations, on the other hand, cause state-of-the-art refinement algorithms to be unable to identify the objects' pose, such that the entire two-step approach is likely to fail.

In this thesis, we propose a novel approach for obtaining initial estimates of all object positions *jointly*. Additionally, we investigate a new local, individual refinement procedure that copes with the shortcomings of state-of-the-art approaches while yielding fast and accurate registration results as well as a large region of attraction. Both stages are designed using advanced numerical techniques such as large-scale convex programming and geometric optimization on the curved space of Euclidean transformations, respectively. They complement each other in that conflicting interpretations are resolved through non-local convex processing, followed by accurate non-convex local optimization based on sufficiently good initializations.

Exhaustive numerical evaluation on artificial and real-world measurements experimentally confirms the proposed two-step approach and demonstrates the robustness to noise, unstructured measurements and occlusions as well as showing the potential to meet run-time constraints of real-world industrial applications.

Acknowledgment

Doing a PhD is a challenging and exhausting road that starts long before the first word has been written. Walking this way is only possible with the adequate support and help of many people. Here, I would like to take the possibility to thank all people who stood by my side or even made this work possible in the first place.

First and foremost, I want to thank Prof. Christoph Schnörr, head of the Image and Pattern Analysis Group at the University of Heidelberg. He introduced me to the field of image analysis during my years of study and gave me the possibility to work in this field. The way he was supervising this dissertation, the fruitful discussions and the profound background together with his straight personality and scientific working style impressed and shaped me a lot.

Next, I want to thank the members of the VMT Vision Machine Technic Bildverarbeitungssysteme GmbH for supporting my dissertation by giving access to technical devices. Especially, I want to thank Jan Linnenkohl who was always a good help regarding technical difficulties.

Additionally, I want to thank my present and former colleagues from the Heidelberg Collaboratory of Image Processing and especially those from the Image and Pattern Analysis Group who accompanied me while being a PhD student. To this end, I want to express my special gratitude to Florian Becker, Jörg Kappes, and Jan Lellmann for proof-reading this work as a whole or in part and providing several helpful comments. I also want to thank Prof. Fred Hamprecht for his accurate feedback concerning this work as well as Barbara Werner, Evelyn Verlinden, and Rita Schieker for helping with administrative problems.

I am eternally grateful to my parents, Petra and Dieter Breitenreicher and my brother, Marc, for their continuous support and for allowing me to pursue the route to my PhD. Without their continuous encouragement, I would not have written this thesis.

Last but not least, I want to thank my girlfriend Diana Wendler. She stood by my side and patiently gave me exceptional backup such that I could proceed working. Finally, she always lifted me up from ground for which I thank her so much.

Contents

Notations

The notation used within this work basically follows the one used in [37] and [56] and should be clear from the context. For convenience we summarize them in the following.

$\mathbb{R}$	real numbers
$\mathsf{SO}(3)$	space of rotations in $\mathbb{R}^3$
$\mathsf{SE}(3)$	space of Euclidean transformations in $\mathbb{R}^3$ (rotation and translation)
$\mathsf{GL}(4)$	space of regular matrices in $\mathbb{R}^{4\times4}$
$\mathfrak{so}(3)$	Lie algebra of $\mathsf{SO}(3)$
$\mathfrak{se}(3)$	Lie algebra of $\mathsf{SE}(3)$
$\theta \in \mathsf{SE}(3)$	specific element of $\mathsf{SE}(3)$
$X, Y \in \mathsf{SE}(3)$	specific element of $\mathsf{SE}(3)$ in matrix representation
$\Phi, \Psi, \Upsilon \in \mathfrak{se}(3)$	specific element of $\mathfrak{se}(3)$ in matrix representation
$\mathcal{M}$	smooth, differentiable manifold
$\mathcal{T}_Y$	tangent space at $Y \in \mathcal{M}$ with $\mathcal{T}_I = \mathcal{T}$
$\mathcal{TM}$	set of tangent vector spaces of $\mathcal{M}$
v_i	i-th element of vector v
A	real valued matrix in $\mathbb{R}^{M\times N}$
a_{ij}	i-th row and j-th column element of matrix A
$v^\top, A^\top$	transposed vector, transposed matrix
A^{-1}	inverse of a (regular) matrix
$\det(A)$	determinant of a matrix
$\operatorname{tr}(A)$	trace of a matrix
I	identity matrix
e	vector of ones, i.e. $e = (1, \ldots, 1)^\top$
0	matrix, vector of zeros
$\|x\|_2, \|A\|_2$	ℓ_2 norm of a vector, spectral norm of a matrix
$\|x\|_1$	ℓ_1 norm of a vector
$\|x\|_0$	pseudo ℓ_0-norm of a vector, i.e. number of non-zero elements
∇f	gradient of the function f
∂f	Euclidean derivative of f
$\overline{\nabla}$	covariant derivative
Γ_{ij}^k	Christoffel symbol specifying $\overline{\nabla}$
$\langle v, w\rangle = v^\top w$	inner product for vectors
$\langle A, B\rangle = \operatorname{tr}\left(A^\top B\right)$	canonical inner product for matrices
$[A, B] = AB - BA$	Lie bracket
$\Delta \subset \mathbb{R}^n$	multi-dimensional unit simplex, i.e. $w \in \Delta \Leftrightarrow w_i \geq 0$, $\sum_{i=1}^n w_i = 1$

Chapter 1
Introduction

1.1 Motivation

Vision guided systems are relevant for many industrial application areas, including manufacturing, medicine, service robots, etc. A task common to these applications consists of detecting and localizing known objects in cluttered scenes. In each case, reliability and accuracy are important besides a sufficiently short processing time.

In this thesis, we develop computer vision techniques for the special scenario of real-world industrial bin-picking applications where the goal is to accurately estimate the positions of multiple instances of arbitrary, but known objects that are randomly assembled in a bin. Figure 1.1 shows a typical setup. This application is required for different subsequent procedures such as quality inspection or picking individual objects by a robot. Although prior knowledge of the object's shape simplifies the problem, a considerable amount of noise, symmetries of the object's shape, and mutual occlusion renders the tasks far from being trivial. A close-up view depicted in the right panel of Fig. 1.1 illustrates these issues.

Besides these difficulties, the imaging process as depicted in Fig. 1.2 causes further problems to the bin picking scenario. The recording procedure can be roughly described as follows. A scanning device moves along a linear axis and subsequently emits laser rays in different directions that are reflected by object instances in the scene and return to a collector. Based on each ray's direction as well as the time required from emitter to collector, it is possible to infer the corresponding scene measurement in 3D. However, due to this special setup, almost half of the object is typically not covered by scanning points and the obtained point cloud is highly unstructured and sparse in general.

In this work we focus on the following requirements:

1. The approach to estimate each object's position should not rely on properties of specific objects, like the geometry of flat disks, for instance. Rather, we only require as input a sparse point sample of the object's surface, obtained from a CAD model if available or by direct measurements if not, so as to enable flexible adaption to novel scenarios by non-experts as user.
2. Numerous ambiguities due to object symmetries and occlusion require a non-local contextual first processing stage in order to reliably detect multiple object instances and rough pose estimates. The latter should be sufficiently accurate to avoid problems with local minima of subsequent pose estimation which is an intrinsically non-convex problem.
3. Subsequent numerical pose estimation should adequately take into account the geometry of the underlying space of transformations so as to minimize the number of iterations while having a large basin of attraction to the correct local minimum.

To this end, we propose a novel approach to model the pose estimation problem for single object instances and investigate possibilities to obtain the corresponding optimizer

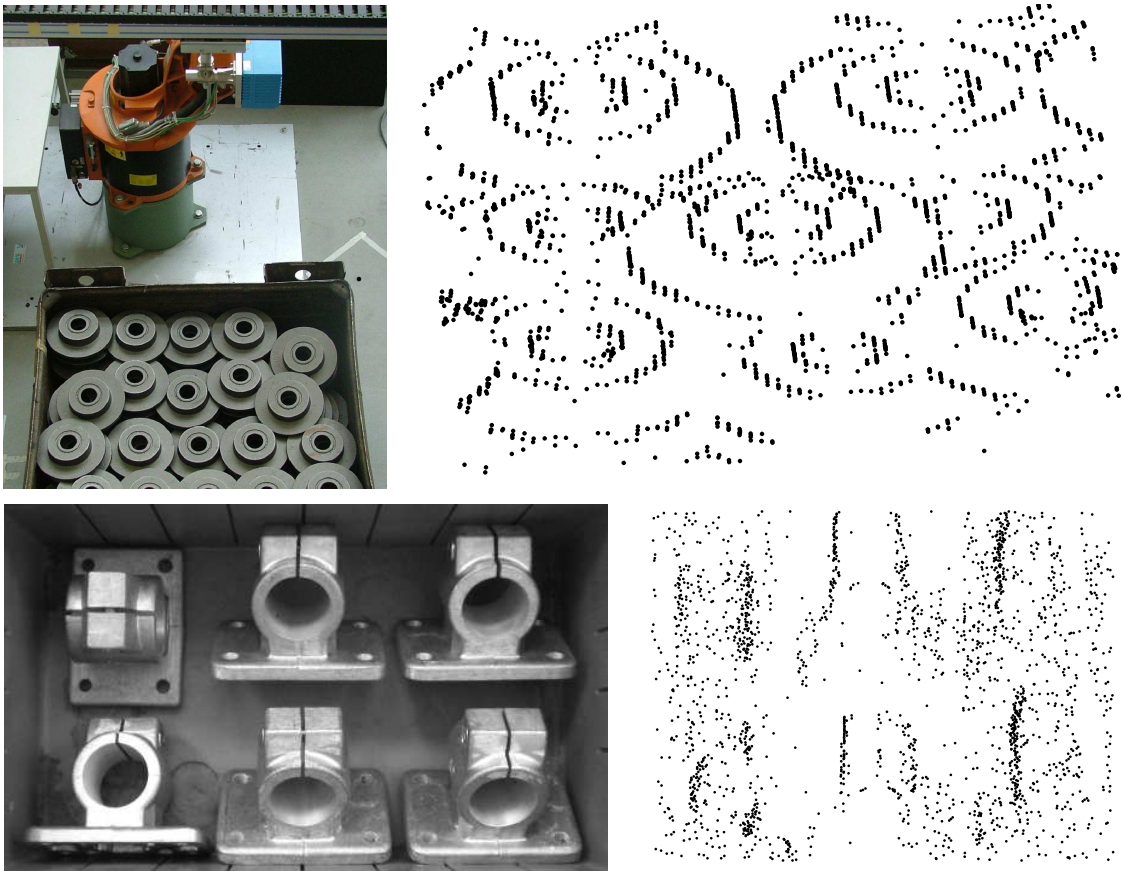


Figure 1.1: Visualization of typical industrial setup for automatic bin-picking applications (left). A SICK-LMS400 scanner is mounted on a linear axis recording a scene of multiple objects randomly assembled in a bin (top left). Due to self occlusion and nearby objects, large parts of a single model are typically not visible (right). Additional noise produced by the scanner further complicates the entire problem.

with respect to the group of rigid body transformations. Additionally, we present a non-local first processing stage that allows to roughly obtain estimates of the number of objects as well as the related position.

1.2 Related Work

There is a vast body of literature on the processing of range data and on object registration in 3D. As overview of established techniques we refer to the survey papers of Besl and Jian [13], Chin and Dyer [29], and the recent work of Salvi et al. [105]. The problem of three-dimensional object recognition generally divides into the three stages of

1. acquisition of input measurements,
2. extraction of salient features, and
3. object recognition by matching these features.

In this work, we do not focus on the first part, i.e. the imaging process, although there are multiple viable techniques to generate 3D measurements such as time-of-flight cameras,

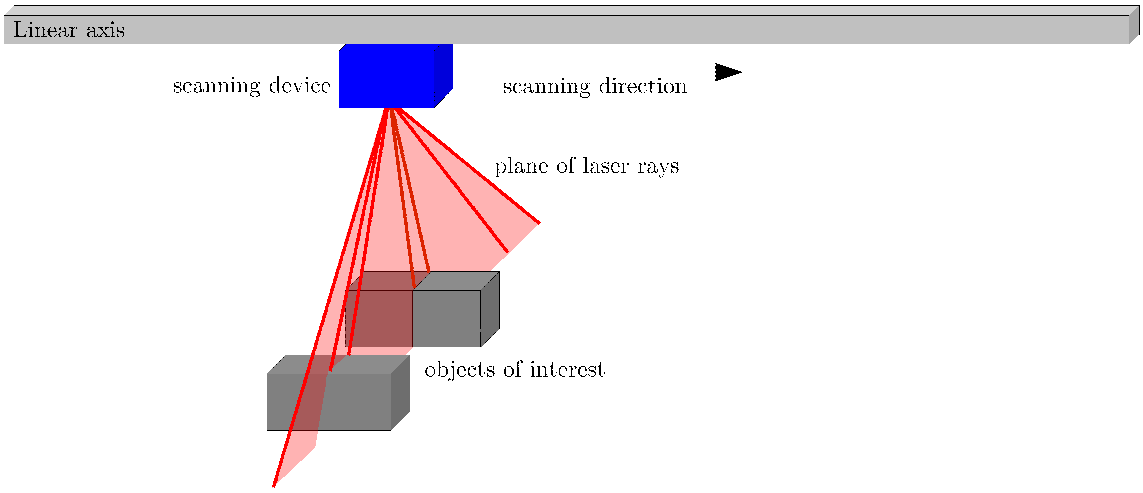


Figure 1.2: Prototypical setup of the imaging process. A scanning device, consisting of emitter and collector, moves linearly along a defined axis and subsequently emits laser rays in different directions. Based on the direction and the time from emitter to collector, it is possible to infer 3D measurements in the scene that are typically sparse and highly unstructured.

triangulation sensor, or stereo vision. For more information we refer the reader to [103] and the references therein.

1.2.1 Feature Extraction

Obtaining salient local landmarks in order to match a given model to a recorded scene is a field widely studied in computer vision literature. For the special case of 3D measurements an overview of recent developments in this area is given in [105].

In general, extracting features from 3D measurements amounts to group individual samples in order to derive characteristic and discriminant information from the point cloud. One of the most intuitive grouping strategies includes to extract lines and planar faces [45], higher order geometric primitives such as cylindrical and conic surfaces [44], or free-form curves and patches [51, 61] from the measured scene.

Closely related to these features are Gaussian images [65] and the local extraction of Gaussian and mean curvature [95] as both rely on local representations of the point cloud surfaces. With increasing measurement's noise, however, such approaches typically yield inaccurate reconstructions such that the quality of the local approximations of the discrete point set are poor.

Using global surface representations in contrast, such as eigen-shapes [24], superquadrics [74], or spherical harmonics [75] enables to cope with increasing noise ratios. However, due to global parametrization, they typically suffer from sensitivity to large amounts of occlusion, a problem occurring in industrial real-world applications.

Instead of reconstructing local parts of the scene, using point measurements directly typically copes with the problem of noisy measurements. Then, local information can be obtained using local grouping techniques such as shape context [8, 48], spin images [71], splash images [114], point signatures [30], and volume integral descriptors [53]. Although

these approaches reveal promising results for objects that have highly distinctive parts, for most man-made objects with lots of symmetries, theses features are typically not discriminate in order to obtain precise one-to-one relationships between scene and model features.

Thus, in the scenario considered in this work, landmark based approaches are likely to fail due to a high degree of occlusion, object symmetry, and sparse noisy measurements (Fig. 1.1). Consequently, we focus on the alternative to work directly with the observed point clouds in order to register objects to the scene and to determine their pose. Still, this includes the problem to establish the correspondence between object model and observations, however.

1.2.2 Feature Matching

Aligning a model to a recorded scene amounts to the *"chicken and egg"* problem of determining simultaneously correspondences between salient features of the model and the scene and a rigid transformation. This joint optimization problem is non-convex and highly interdependent since alignment depends on correspondence and vice versa. However, having solved either problem renders the problem convex in the missing variables, such that obtaining a solution becomes trivial [14].

Iterative Refinement Procedures

As a consequence, there is a wide range of approaches applying iterative search techniques by means of subsequently estimating the rigid transformation followed by an update of correspondences, and so forth. The prototypical representative of this group of matching strategies is the Iterative Closest Point algorithm (ICP) [14, 27] that has been extended several times in order to cope with different feature representations [134] and to address issues such as robustness [47, 54, 57, 96] and computational complexity [72, 104].

Although this algorithm is still a state-of-the-art algorithm [104, 111, 135], it is widely known that explicit correspondences increase both the non-convexity and non-smoothness of the related objective function such that poor initial configuration typically cause this approach to fail to converge to the globally optimal solution.

Due to this major limitation of iterative refinement algorithms, recent approaches decouple the joint optimization by means of estimating the global optimal configuration of either the transformation or the correspondence separately and to infer the missing variable correspondingly.

Correspondence Based Approaches

The problem of estimating the optimal correspondence amounts to solve a combinatorial problem that is closely related to the problem of vertex cover [43] and subgraph matching [107], where the latter belongs to Karp's list of NP-complete problems [73].

In case that a partial matching can be inferred from uniqueness of the individual features, highly efficient inference algorithms [17, 101] can be applied in order to avoid sifting through all possible combinations. However, as features are non-distinctive in general, exhaustive search algorithms [5, 58, 129, 130] have to be applied that are, due to the

problem's size, only applicable using well-tuned data structures and efficient reduction techniques such as coarse-to-fine scaling.

As a consequence, solutions of this problem are typically approached using heuristic, randomized techniques such as the random sample consensus [46, 118] and genetic algorithms [105]. Even though randomized algorithms typically reveal promising results, they provide at most probabilistic guarantees to obtain the global optimizer. In many settings, however, they are unable to detect an optimal configuration at all.

Relaxing the NP-complete optimization problem [107] in contrast yields to cope with a convex optimization problem that tightly approximates the original problem. Solving this problem, however, is computationally complex such that it is difficult to meet the strict time restrictions of industrial settings. Closely related to [107], a very recent approach of Enqvist et al. [43] approximates the underlying problem structure subsequently in order to derive tight lower bounds of the problem and to apply branch and bound strategies.

With increasing number of points, however, the problem's size grows exponentially such that correspondence based matching approaches tend to fail to meet the strict time and memory restrictions of many industrial applications.

Transformation Based Approaches

The alternative class of approaches is to optimize over the space of transformations and to infer the optimal point-to-point assignment accordingly. This is primarily based on the approximation of the objective functional in terms of a function that only depends on the transformation.

Mitra et al. [84] and Pottmann et al. [94] suggested to use a local piecewise-quadratic approximation of the objective function that encodes the correspondence of model points to scene measurements implicitly. This scheme avoids exploration of the correspondence space at run-time at the costs of exhaustive pre-computation.

Another possibility to cope with explicit correspondences is based on the work of Tsin and Kanade [121] that has been further extended by Jian and Vemuri [70]. They proposed to infer the optimal alignment by optimizing the distance between mixture distributions, where each distribution represents a continuous formulation of the model and scene points, respectively. Possible distance measures include the Euclidean distance [70] or the Jensen-Shannon [128] divergence. However, similar to iterative refinement procedures, the obtained objective functions are typically highly non-convex such that accurate initializations are of utmost importance.

In this context, we also refer to very recent work [64, 81, 90] on global optimization approaches for the problem of transformation estimation. These approaches apply Branch and Bound techniques for exploring the pose parameter space, where Hartley and Kahl [64] as well as Olsson et al. [90] require very special settings, i.e. explicit correspondence of scene points to convex model parts. In contrast, Li and Hartley [81] works without such correspondences at the costs of bad run-time scaling for real-world problems.

1.3 Contribution

In this thesis, we introduce a novel *initialization and refinement* approach for the problem of model-based detection as well as the determination of the transformations of multiple

objects for the real-world industrial bin-picking scenarios where the scene is represented by noisy, unstructured and sparse point measurements (cf. Fig. 1.1).

To this end, we propose a new initialization stage in terms of a global, large-scale, convex objective function that accurately describes the geometrical constraints of the pose estimation problem. Additionally, this enables to apply efficient preprocessing techniques derived from sufficient optimality conditions as well as to use dedicated algorithms for convex optimization. Finally, it yields promising performance making the approach attractive for solving real-world applications with tight run-time constraints.

As a subsequent refinement step, we investigate a novel non-convex objective distance measure yielding an objective functional for rigid point set alignment that provides convenient approximations of state-of-the-art approaches and avoids explicit correspondences of model points to scene measurements. To obtain the local optimizer of this new objective, we consider advanced optimization techniques based on higher order numerical integration as well as Newton-like optimization techniques that fully exploit the intrinsic geometry of the underlying space of Euclidean transformations and enables fast convergence to the local optimum for a large region of attraction.

A thorough numerical evaluation demonstrates the potential of our new two-step approach to meet the accuracy and run-time constraints of the industrial bin-picking scenario. Moreover, we believe that adopting our approach might be attractive in other related scenarios of computer vision as well. Contents of this work have been partially published in [21, 22, 23].

1.4 Organization

The remainder of this work is sectioned into four major parts. In Chap. 2, we mathematically introduce the problem of Euclidean point set alignment by means of determining the rigid transformation that matches a given model to a set of scene measurements best. To this end, we consider different possibilities to model the problem accurately, and propose a novel, smooth objective functional that provides convenient approximations of state-of-the-art approaches while exhibiting the ability to apply higher order optimization techniques.

To determine the optimizer of such an objective functional, in Chap. 3, we consider two conceptually different approaches based on higher order numerical integration techniques as well as on Newton-like algorithms that fully exploit the geometry of the underlying group of Euclidean transformations.

As these algorithms provide only local convergence properties, however, and as objective functions used for Euclidean point set registration are highly non-convex in general, in Chap. 4, we investigate the problem of jointly obtaining proper initializations in terms of rough estimates of each object's position. We present a non-local processing stage that encodes the numerous ambiguities due to object symmetries and occlusion, and that reliably detects multiple object instances and obtains rough pose estimates.

In Chap. 5, we apply the proposed approaches to artificial and real-world point sets in order to numerically evaluate the performance of the different individual approaches presented in Chap. 2, Chap. 3, and Chap. 4. This demonstrates that the single steps complement each other. Finally, in Chap. 6, we summarize the major results obtained and indicate promising directions of further research.

Chapter 2
Problem Statement and Optimization Criteria

2.1 Overview

2.1.1 Introduction and Motivation

Obtaining a mathematical model for a given real world problem is required in most application driven fields of computer vision. The general approach to cope with this problem is to use a model that admits to describe the scenario as accurate as necessary while keeping the model simple. As simplicity and accuracy are contradicting issues (sketched in Fig. 2.1), finding "good" problem descriptions turns out to be the crucial point in most applications.

In this chapter, we focus on the formulation of a mathematical model for the problem of Euclidean point set registration that allows to obtain a Euclidean transformation that aligns a given model to a recorded set of scene measurements best.

2.1.2 Related Work and Contribution

Finding the Euclidean transformation that aligns scene measurements to a set of model points best, amounts to a *"chicken and egg"* problem of determining simultaneously the point-to-point correspondences and the Euclidean transformation. Having solved either problem, the other one becomes trivial [14].

As a consequence, most approaches model the Euclidean registration as the problem of finding the transformation that minimizes the distance between corresponding points. As the Euclidean transformation and the correspondence is interdependent in general such that joint optimization is not possible, most approaches proceed in an iterative fashion, where given an estimate of the transformation, correspondence is estimated by some heuristic, followed by updating the transformation estimate, and so forth. The prototypical representative of this class of approaches is the Iterative Closest Point (ICP) algorithm developed in parallel by Besl and McKay [14] and Chen and Medioni [27]. Due to its simplicity and fast convergence, ICP is still a state-of-the-art algorithm [104, 111, 135] that is widely used within many industrial applications.

As is well known, this basic two-step iteration is susceptible to noise and poor initialization, however. To overcome these issues Rusinkiewicz and Levoy [104], Gold et al. [54], Rangarajan et al. [96], and Fitzgibbon [47] proposed robust variants that are more tolerant against imprecise initializations as well as against noise and outlying structures at the costs of introducing additional threshold parameters and annealing schedules. Anyway, a major drawback concerning the *representation* of the problem remains, in particular when dealing with unstructured point sets: explicit correspondences increase both the non-convexity and non-smoothness of the objective function, and gaining insight into the optimization problem is hampered by complicated structure of the domain of optimization comprising both Euclidean transformations and correspondence variables.

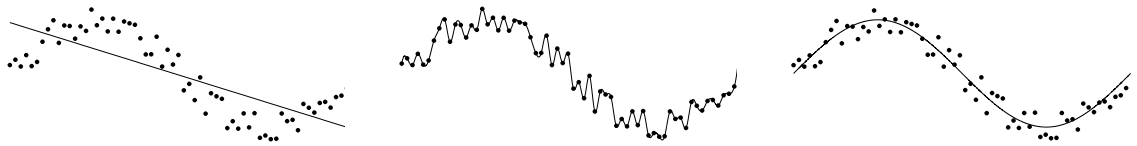


Figure 2.1: Sketch of the problem of finding "good" models for a given applications: Assuming a discrete set of points is to be described by a continuous function, too simple models like a straight line (left) cannot capture the scene characteristics accurately, while precise descriptions yield difficult models and overfitting to noise (middle). Determining a simple model coping with the properties of the scene (right) is to be studied in this chapter.

In order to model the uncertainties of correspondences more precisely, Granger and Pennec [57] considered a robust ICP approach in a probabilistic expectation maximization framework (EM-ICP). Using this framework facilitates the interpretation of involved thresholds. Yet, despite increased robustness, there appears to be still room for improvement, especially concerning the requirement of accurate initializations of correspondences.

To alleviate the computation of corresponding point-to-point matches in each iterate, Mitra et al. [84] as well as Pottmann et al. [94] suggested to approximate the objective distance function by local quadratic functions that represent the distance of certain model points to the entire scene.

Another way to avoid the explicit determination of correspondence has been suggested by Tsin and Kanade [121], Jian and Vemuri [70] and Wang et al. [128]. By representing point clouds of both the scene and the model by mixture distributions, registration can be achieved by minimizing the squared ℓ_2 distance [70, 121] or the Jensen-Shannon divergence [128] between two distributions. Compared to [84, 94], this avoids exhaustive pre-computation of local distance approximation at the cost of more expensive function evaluations.

Especially in industrial bin-picking applications where the scene is readjusted several times, we consider the latter class of approaches as advantageous in connection with *unstructured* noisy point sets. Thus, we adopt mixture models of model and scene points in this work as well.

We furthermore investigate a novel, smooth objective for the problem of Euclidean point set registration being independent of explicit point-to-point correspondences. We show the direct relation to state-of-the-art approaches while exhibiting the property for higher order optimization techniques, instead of embossing the optimization procedure into the model representation. Finally, we point out benefits and drawbacks of this formulation to cope with industrial bin-picking applications. The results of this chapter have been partially published in [22, 23].

2.1.3 Organization

The remainder of this chapter is organized as follows. In Sec. 2.2, we review state-of-the-art formulations of the problem of Euclidean point set alignment based on determining explicit correspondences between model and scene.

Next, in Sec. 2.3, we analyze a mathematical model that is independent of explicit correspondences and propose our novel objective function for registration problems based

on computing the natural distance between continuous point set representations. The relation of our new formulation to state-of-the-art approaches is analyzed in Sec. 2.4. Finally, we conclude in Sec. 2.5 and point out further work.

2.2 Correspondence-Dependent Functions

Let $\mathcal{U} = \{u_i,\, i = 1, \ldots, n\} \subset \mathbb{R}^3$ denote the set of scene measurements obtained by a scanning device, and let $\mathcal{V} = \{v_j,\, j = 1, \ldots, m\} \subset \mathbb{R}^3$ be the set of samples specified by a given model description, i.e. a CAD file or a sample scan.

In practice, explicit geometric mesh models are typically not available. Furthermore, generating meshes from thousands of 3D point measurements may not be error-free and causes a time-consuming preprocessing step. We therefore only consider models in terms of 3D point measurements in order to uniformly handle all practically relevant situations.

The objective of Euclidean point set registration is to determine a rigid body transformation $\theta \in \mathsf{SE}(3)$ such that model and scene are aligned best, given a certain distance measure. Here, $\mathsf{SE}(3)$ denotes the special Euclidean group of rigid body transformations whose elements are uniquely specified by a proper rotation of the 3D space, followed by a translation.

Typically and also most intuitively, point set alignment is achieved through correspondence based registration, that is to minimize the cumulative distance between related points in $\mathcal{U}$ and $\mathcal{V}$. Due to the embedding of $\mathcal{U}, \mathcal{V}$ in $\mathbb{R}^3$, a common measure is the sum of squared Euclidean point-to-point distances given by

$$\sum_{i=1}^{n} \left\| u_i - \theta\big(v_{\kappa(i)}\big) \right\|_2^2 , \tag{2.1}$$

where $\theta\big(v\big)$ rigidly transforms $v \in \mathcal{V}$ and $\kappa \in \mathcal{F} = \{\eta | \eta : \{1, \ldots, n\} \mapsto \{1, \ldots, m\}\}$ denotes a correspondence function that assigns each scene point u_i to its model counterpart $v_{\kappa(i)}$.

Besides the intuitive meaning and the simplicity of (2.1), modeling the registration problem in terms of squared Euclidean point-to-point distances reveals the ability to separately determine the unknowns in closed form while suffering from few shortcomings.

The mathematical model (2.1) is based on the assumption that each scene measurement corresponds to an object point. This however does not hold for most industrial setups due to clutter and noise, whereas the reverse problem, i.e. exchanging model and scene points, does not capture the bin-picking setup too, due to mutual and self occlusion of the object instances. Although these limitations can be coped by introducing a further "background point" [54] that represents clutter and occlusion or by rejecting "poor" matches [76, 104], for the moment, let us consider the simplified scenario of having no clutter or no occlusion, however. We will get back to this issue in Sec. 2.2.1.

Fixed Correspondence and Missing Transformation

Assume that the correspondence assignment function κ is known in prior, such that, without loss of generality, $v_i \in \mathcal{V}$ corresponds to scene measurement u_i. Then, minimizing (2.1) with respect to $\theta \in \mathsf{SE}(3)$ yields

$$\underset{\theta \in \mathsf{SE}(3)}{\arg\min} \sum_{i=1}^{n} \left\| u_i - \theta\big(v_i\big) \right\|_2^2 . \tag{2.2}$$

Algorithm 1 Iterative Closest Point Algorithm (ICP)

Require: $\theta \in \mathsf{SE}(3)$

repeat

update correspondence:

$$\kappa(i) = \underset{j=1,\dots,m}{\arg\min} \left\| u_i - \theta\big(v_j\big) \right\|_2^2$$

update transformation:

$$\theta = \underset{\vartheta \in \mathsf{SE}(3)}{\arg\min} \sum_{i=1}^{n} \left\| u_i - \vartheta\big(v_{\kappa(i)}\big) \right\|_2^2$$

until convergence

Depending on the representation of $\mathsf{SE}(3)$, this problem can be solved in closed form, where we refer to [42] for details.

Fixed Transformation and Missing Correspondence

In contrast, assuming the Euclidean transformation θ to be known, the optimal correspondence assignment function $\kappa \in \mathcal{F} = \{\eta | \eta : \{1,\dots,n\} \mapsto \{1,\dots,m\}\}$ is the solution of

$$\underset{\kappa \in \mathcal{F}}{\arg\min} \sum_{i=1}^{n} \left\| u_i - \theta\big(v_{\kappa(i)}\big) \right\|_2^2 \ . \tag{2.3}$$

As $\kappa(i)$ only depends on scene measurement u_i, the optimal solution can be obtained by minimizing $\left\| u_i - \theta\big(v_{\kappa(i)}\big) \right\|_2^2$ separately for all $i = 1,\dots,n$, such that the optimal $\kappa(i)$ reads as

$$\kappa(i) = \underset{j=1,\dots,m}{\arg\min} \left\| u_i - \theta\big(v_j\big) \right\|_2^2 \ .$$

As a consequence, given $\theta \in \mathsf{SE}(3)$, the optimal correspondence is given by assigning each u_i to its closest transformed model point. To sum up, knowing either the Euclidean transformation θ or the assignment function κ renders the minimization of problem (2.1) being trivial.

However, as the determination of the Euclidean transformation and the correspondence depends on each other and no prior knowledge exists in general, joint optimization of (2.1) has to be applied that turns out to be involved.

2.2.1 Hard Correspondence Assignment

To cope with the problem of joint estimation of correspondence and Euclidean transformation, the objective (2.1) is typically updated in an alternating, iterative fashion. This means, given an initial estimate on the Euclidean transformation θ, using (2.3) allows to determine the optimal correspondence followed by an update of θ according to (2.2) subject to a fixed assignment function, and so forth.

This intuitive approach of minimizing (2.1) is commonly known in literature as the Iterative Closest Point (ICP) algorithm [14, 27, 134], summarized by Alg. 1. Due to

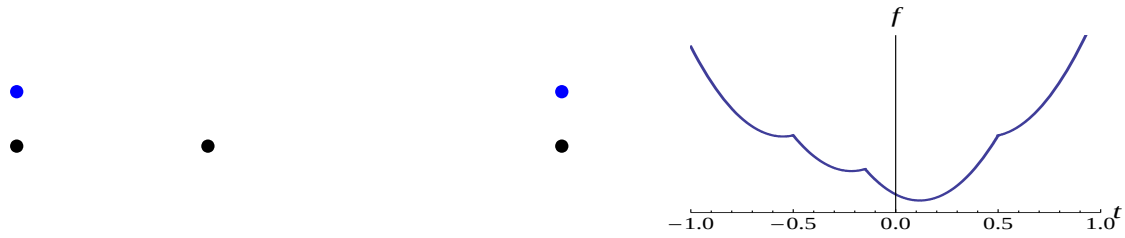


Figure 2.2: Simple scenario showing the major drawbacks of ICP algorithm. While having a simple model (blue) and scene (black) supplemented by a single outlier (left), the objective function of ICP with respect to horizontal translation amounts to be highly non-smooth and non-convex (right) such that determining the global optimizer is difficult.

the simplicity of the single steps of ICP, it still belongs to the class of state-of-the-art algorithms for Euclidean point-set alignment and is widely applied in industrial vision applications [111, 135]. Due to the subsequent minimization of the objective function, Alg. 1 is guaranteed to converge to a minimizer of (2.1) [94].

However, because of the non-convexity of (2.1) resulting from hard correspondence assignment, as well as presence of outliers and noise sketched in Fig. 2.2, Alg. 1 is known to be susceptible to poor initialization such that it converges to local optimizer only.

To overcome this issue, a reasonable strategy is to remove outliers in prior or within early iterates at the costs of introducing an additional threshold parameter [76, 104]. Let d_{max} be the maximal allowed distance of a scene sample to its transformed model counterpart $\theta(v_{\kappa(i)})$. Then, the set of outliers can be defined as

$$\bar{\mathcal{U}} = \left\{ u_i \,\middle|\, \|u_i - \theta\big(v_{\kappa(i)}\big)\|_2^2 \geq d_{max} \right\} \ ,$$

where the set of inliers is given by $\widetilde{\mathcal{U}} = \mathcal{U} \setminus \bar{\mathcal{U}}$, respectively. Hence, rejection of outliers yields an update of the Euclidean transformation θ aligning $\widetilde{\mathcal{U}}$ to $\mathcal{V}$ best.

Instead of hard outlier rejection that typically further increases the non-smoothness of the objective, smooth rejection strategies are widely applied within literature [15, 76, 104]. Let $w_i \geq 0, i = 1, \ldots, n$ denote weights that encode the relative certainty of u_i being an inlier. Then, aligning $\mathcal{U}$ to $\mathcal{V}$ amounts to restate the objective function in terms of a weighted least squares problem as

$$\sum_{i=1}^{n} w_i \left\| u_i - \theta\big(v_{\kappa(i)}\big) \right\|_2^2 \ ,$$

where w_i are typically determined relative to the inverse of $\left\|u_i - \theta(v_{\kappa(i)})\right\|_2^2$ in prior.

However, similar to the hard rejection strategies, by definition, outliers highly depend on the initial configuration of the Euclidean transformation. Hence, having lots of noise and nearby outliers, a typical scenario in industrial applications, simple outlier removing strategies are likely to fail as only "far away" outliers can be removed reliably [104, 105].

2.2.2 Soft Correspondence Assignment

To alleviate the non-convexity and the non-smoothness as well as the related sensitivity to poor initializations, in the following we study smooth approximations of the objective

alignment function (2.1). The smoothing of the objective function is generally based on the following theorem.

Theorem 2.1. *[100] Let $x \in \mathbb{R}^n$ and let $\Delta \subset \mathbb{R}^n$ denote the unit simplex, i.e. $\Delta = \{w \,|\, w \in \mathbb{R}^n, w_i \geq 0, \sum_{i=1}^n w_i = 1\}$. Then, the value of a minimal entry of x is given by*

$$\min_{i=1,\dots,n} x_i = \min_{w \in \Delta} \sum_{i=1}^{n} w_i x_i \ .$$

The relation of Thm. 2.1 to (2.1) can be seen by rewriting the cumulative quadratic Euclidean distance of point-to-point matches as the joint minimization problem of θ and the closest point assignment according to (2.2) and (2.3) as

$$\min_{\theta \in \mathsf{SE}(3)} \sum_{i=1}^{n} \min_{j=1,\dots,m} \left\| u_i - \theta\big(v_j\big) \right\|_2^2$$

such that we obtain the equivalent optimization problem

$$\min_{\theta \in \mathsf{SE}(3)} \sum_{i=1}^{n} \min_{w_i \in \Delta} \sum_{j=1}^{m} w_{ij} \left\| u_i - \theta\big(v_j\big) \right\|_2^2 \ , \tag{2.4}$$

where w_{ij} refers to the j-th entry of $w_i \in \mathbb{R}^m$. Thus, instead of considering the discrete assignment problem that is part of (2.1), the optimal configuration can be obtained by considering a continuous optimization problem on a continuous domain.

Soft Correspondences by Relaxation

Despite the redefinition of (2.1) as an optimization problem on a continuous domain, minimizing (2.4) suffers from non-smoothness and non-convexity like the original objective of ICP, as the optimal weights w_i are indicator vectors in $\mathbb{R}^n$ that correspond to hard assignment functions.

One possibility to cope with the non-smoothness of the objective introduced by hard assignment and to remove the sensitivity to too local optima, is to modify (2.4) by adding an additional regularization prior. Using the convex negative entropy barrier function [96] with parameter $\lambda \geq 0$, a suitable relaxed version of (2.4) reads

$$\min_{\theta \in \mathsf{SE}(3)} \min_{\substack{w_i \in \Delta, \\ i=1,\dots,n}} \left(\sum_{i=1}^{n} \sum_{j=1}^{m} w_{ij} \left\| u_i - \theta\big(v_j\big) \right\|_2^2 + \lambda \sum_{i=1}^{n} \sum_{j=1}^{m} w_{ij} \log w_{ij} \right) \ . \tag{2.5}$$

Obviously, if $\lambda \to 0$ the solution of (2.5) equals the optimal configuration of (2.4) while for $\lambda \to \infty$, (2.5) is a convex function on a convex domain. Hence, for any $\lambda > 0$ the corresponding objective is smoother and less sensitive to poor initializations then (2.1) as sketched in Fig. 2.3. Moreover, similar to (2.1), the relaxed variant (2.5) admits closed form solutions in case of a-priori knowledge of either w_i or θ as we will see in the following.

Assuming w_{ij} to be known, optimizing (2.5) yields to solve a weighted least squares problem with respect to $\theta \in \mathsf{SE}(3)$ that can be done in closed form [42] similar to (2.2).

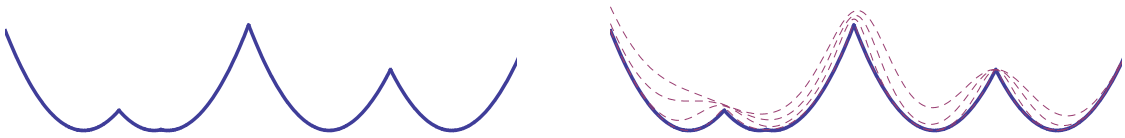


Figure 2.3: Visualization of the objective functions of the Iterative Closest Point algorithm (left) and the smoothed variant of ICP using different control parameters (right). While the objective of ICP is highly non-smooth and non-convex (solid line), the smoothed variant of ICP is less sensitive to local extrema and yields a smooth objective (dashed lines) at the cost of additional control parameters.

On the other hand, determining the weights $w_i \in \Delta$ for fixed θ, yields the sufficient Karush-Kuhn-Tucker (KKT) conditions [18] given by

$$\begin{aligned} \left\| u_i - \theta\big(v_j\big) \right\|_2^2 + \lambda \log w_{ij} + \lambda + \xi_i = 0 \,, & \\ \sum_{j=1}^{m} w_{ij} = 1 \,, \quad \forall\, i = 1, \dots, n \,, & \\ w_{ij} \geq 0 \,, & \end{aligned}$$

where $\xi_i, i = 1, \dots n$ denote the Lagrangian parameters caused by the simplex constraints. The optimal weights w_{ij} satisfying the KKT conditions read

$$w_{ij} = \frac{\exp\left(-\frac{1}{\lambda}\left\| u_i - \theta\big(v_j\big) \right\|_2^2\right)}{\sum_{k=1}^{m} \exp\left(-\frac{1}{\lambda}\left\| u_i - \theta\big(v_k\big) \right\|_2^2\right)} \,.$$

Consequently, similar to Alg. 1, it is reasonable to optimize (2.5) by iteratively updating the Euclidean transformation θ and the weights w_{ij}, where the choice of λ highly affects the final result.

However, finding the optimal parameter λ is difficult in general as choosing λ too small increases the non-smoothness of the objective as well as the sensitivity to inaccurate initializations while large λ typically result in poor approximations of the underlying objective defined by (2.1).

This trade-off of robustness to poor initializations while assuring the final result being an optimizer of (2.1) suggests to apply an annealing procedure [102, 96], i.e. starting with $\lambda > 0$ and subsequently reducing λ to almost 0. Using a simple linear annealing schedule, the relaxed variant of the ICP procedure can be summarized by Alg. 2.

Soft Correspondences by Probabilistic Modelling

However, deriving a precise meaning of the values of w_{ij} is difficult in general, due to the dependency on λ. In order to model uncertainties more directly, we consider a probabilistic description of the Euclidean alignment problem next.

Let us assume that each scene measurement u_i is a concrete realization of a random variable. Then, given a set of model samples $\mathcal{V}$ and a Euclidean transformation θ, the

Algorithm 2 Relaxed Variant of ICP using Linear Annealing

Require: $\theta \in \mathsf{SE}(3)$, $0 < \lambda_{stop} \leq \lambda_{start}$, $\rho \in (0, 1)$

$\lambda = \lambda_{start}$

while $\lambda > \lambda_{stop}$ **do**

repeat

update weights:

$$w_{ij} = \frac{\exp\left(-\frac{1}{\lambda}\left\|u_i - \theta\big(v_j\big)\right\|_2^2\right)}{\sum_{l=1}^{m}\exp\left(-\frac{1}{\lambda}\left\|u_i - \theta\big(v_l\big)\right\|_2^2\right)}$$

update transformation:

$$\theta = \underset{\vartheta\in\mathsf{SE}(3)}{\arg\min}\sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}\left\|u_i - \vartheta\big(v_j\big)\right\|_2^2$$

until convergence

$\lambda = \rho\,\lambda$

end while

likelihood of measuring the realizations specified by $\mathcal{U}$ reads as

$$p(u_1, \ldots, u_n|\theta, v_1, \ldots, v_m) = \prod_{i=1}^{n} p(u_i|\theta, v_1, \ldots, v_m)\ , \tag{2.6}$$

where we assume the scene measurements to be pairwise independent given the model samples and the transformation such that the joint distribution factorizes.

Albeit the lack of an analytic form of (2.6) that allows direct determination of the Euclidean transformation maximizing the likelihood, we can tackle the optimization of (2.6) using the expectation-maximization framework [36].

Let $\omega_i \in \{1, \ldots, m\}, i = 1, \ldots, n$ denote the hidden random variables that indicate the correspondence of the scene measurement u_i to model sample v_j. Marginalizing over the domain of correspondences, i.e. summing over all possible configurations of ω_i, enables to rewrite (2.6) as

$$\prod_{i=1}^{n} p(u_i|\theta, v_1, \ldots, v_m) = \prod_{i=1}^{n}\sum_{\omega_i} p(u_i, \omega_i|\theta, v_1, \ldots, v_m)\ .$$

Moreover, instead of maximizing (2.6), due to the concavity of the log-function, it is common to consider the maximization of the log-likelihood

$$\log\prod_{i=1}^{n}\sum_{\omega_i}\left(p(u_i, \omega_i|\theta, v_1, \ldots, v_m)\,\frac{p(\omega_i|u_i, \hat{\theta}, v_1, \ldots, v_m)}{p(\omega_i|u_i, \hat{\theta}, v_1, \ldots, v_m)}\right)\ ,$$

where we augmented each term by the prior probability of the correspondence variable $p(\omega_i|u_i, \hat{\theta}, v_1, \ldots, v_m)$ given the model as well as a fixed $\hat{\theta} \in \mathsf{SE}(3)$. Then, applying Jensen's inequality, i.e. $\log\big(\sum_x f(x)\,p(x)\big) \geq \sum_x \log\big(f(x)\big)\,p(x)$ where $p(x)$ is a probability density function, as well as using the features of the log function, yields the log-likelihood to be

lower bounded by

$$\sum_{i=1}^{n} \sum_{\omega_i} \log\left(p(u_i, \omega_i|\theta, v_1, \ldots, v_m)\right) p(\omega_i|u_i, \hat{\theta}, v_1, \ldots, v_m) - \sum_{i=1}^{n} \sum_{\omega_i} \log\left(p(\omega_i|u_i, \hat{\theta}, v_1, \ldots, v_m)\right) p(\omega_i|u_i, \hat{\theta}, v_1, \ldots, v_m) \ . \tag{2.7}$$

In literature, this function is known to be the so-called Q function [36] that is to be optimized in the following using the alternating optimization approach of updating the distribution of correspondences (expectation) followed by deriving the unknown Euclidean transformation (maximization). For the problem of Euclidean point set registration, this approach is commonly called the Expectation-Maximization (EM-) ICP algorithm [57].

Expectation: Updating the Q function with respect to the unknown prior distribution of correspondences amounts to determine $p(\omega_i|u_i, \hat{\theta}, v_1, \ldots, v_m)$ that solves the KKT conditions of (2.7) given by

$$\sum_{\omega_i} \log\left(p(u_i, \omega_i|\theta, v_1, \ldots, v_m)\right) - \log\left(p(\omega_i|u_i, \hat{\theta}, v_1, \ldots, v_m)\right) + \xi_i = 0 \ ,$$

$$\sum_{\omega_i} p(\omega_i|u_i, \hat{\theta}, v_1, \ldots, v_m) = 1 \ ,$$

for all $i = 1, \ldots, n$ where ξ_i denote Lagrangian parameters. Then, the solution of the KKT conditions can be computed in closed form and is given by

$$p(\omega_i = j|u_i, \hat{\theta}, v_1, \ldots, v_m) = \frac{p(u_i, \omega_i = j|\theta, v_1, \ldots, v_m)}{\sum_{k=1}^{m} p(u_i, \omega_i = k|\theta, v_1, \ldots, v_m)} \ . \tag{2.8}$$

Maximization: To maximize the Q function with respect to the unknown Euclidean transformation θ, we consider a parametric form of $p(u_i, \omega_i = j|\theta, v_1, \ldots, v_m)$ next. According to Bayes' formula the joint distribution of scene measurement and correspondence reads

$$p(u_i, \omega_i = j|\theta, v_1, \ldots, v_m) = p(u_i|\omega_i = j, \theta, v_1, \ldots, v_m)\, p(\omega_i = j|\theta, v_1, \ldots, v_m) \ , \tag{2.9}$$

where $p(\omega_i = j|\theta, v_1, \ldots, v_m)$ denotes the prior distribution encoding the probability that any scene measurement corresponds to model sample v_j. As the prior distribution can be assumed to be independent of θ, we can rewrite the latter term of (2.9) as $p(\omega_i = j|v_1, \ldots, v_m)$.

Moreover, assuming the i-th scene sample being caused by the j-th model point, i.e. $\omega_i = j$, it is reasonable to model the probability of measuring the realization of scene sample u_i as the relative deviation from the expected position $\theta(v_j)$ defined by the normal distribution

$$p(u_i|\omega_i = j, \theta, v_1, \ldots, v_m) = \frac{1}{z} \exp\left(-\frac{1}{2\sigma^2} \left\|u_i - \theta\big(v_j\big)\right\|_2^2\right) \ , \tag{2.10}$$

where $\sigma > 0$ is the user defined standard deviation of p depending on the expected noise ratio of the measurements and z denotes the denominator of the normal distribution.

As the second part of (2.7) does not depend on the Euclidean transformation, applying (2.9) to (2.7) and inserting the normal distribution (2.10) yields the optimal θ being the argument maximizing

$$\begin{aligned}&\sum_{i=1}^{n}\sum_{j=1}^{m}\left(-\frac{1}{2\sigma^2}\left\|u_i-\theta\big(v_j\big)\right\|_2^2\right)\,p(\omega_i=j|u_i,\hat{\theta},v_1,\ldots,v_m)\,+\\&\sum_{i=1}^{n}\sum_{j=1}^{m}\log\big(p(\omega_i=j|v_1,\ldots,v_m)\big)\,p(\omega_i=j|u_i,\hat{\theta},v_1,\ldots,v_m)\;,\end{aligned}\tag{2.11}$$

where we used the fact that ω_i is defined over a discrete domain to avoid numerical integration.

The prior distribution $p(\omega_i = j|u_i, \hat{\theta}, v_1, \ldots, v_m)$ is assumed to be known from the expectation step. Thus, optimizing (2.11) with respect to $\theta \in \mathsf{SE}\,(3)$ turns out to be a weighted least squares problem that can be solved in closed form similar to (2.2).

Finally, it remains to update $p(\omega_i = j|v_1, \ldots, v_m)$ introduced in (2.9). Solving the KKT conditions subject to p being a probability distribution, lets the optimal prior read as

$$p(\omega_i = j|v_1, \ldots, v_m) = \frac{1}{n}\sum_{i=1}^{n} p(\omega_i = j|u_i, \hat{\theta}, v_1, \ldots, v_m)$$

for all $j = 1, \ldots m$.

This two step procedure of updating the distribution of the hidden correspondence variables according to the expectation step (2.8) followed by maximizing the Q function with respect to θ and the prior distribution, gradually increases (2.7) and is guaranteed to converge to a local optimizer [36] while providing precise interpretations of the variables involved.

Moreover, letting $\sigma \to 0$, the update of the hidden variables using (2.8) results in Dirac-distributions, such that the complete EM-ICP approach turns out to be the probabilistic generalization of Alg. 1. However, similar to the relaxation of (2.1) defined by (2.5), the performance of EM-ICP highly depends on the choice of the uncertainty parameter σ of (2.10) that controls the sensitivity to noise and poor initializations as well as the accuracy of approximation of the original objective (2.1).

2.3 Correspondence-Independent Functions

In spite of the precise modeling of uncertainties and noise as well as the relaxation of (2.1), there still appears to be room for improvements, especially concerning the requirement of accurate initializations of point-to-point correspondences, where single wrong assignments yield poor registration results in general. Moreover, as the optimization procedure is hampered into the model representation it does not allow to apply sophisticated frameworks to increase the performance in terms of speed and robustness.

To alleviate the explicit estimation of correspondences and related drawbacks, recent approaches adopted continuous representations of point sets using probabilistic mixture distributions [70, 121]. These representations for the model and the scene, respectively,

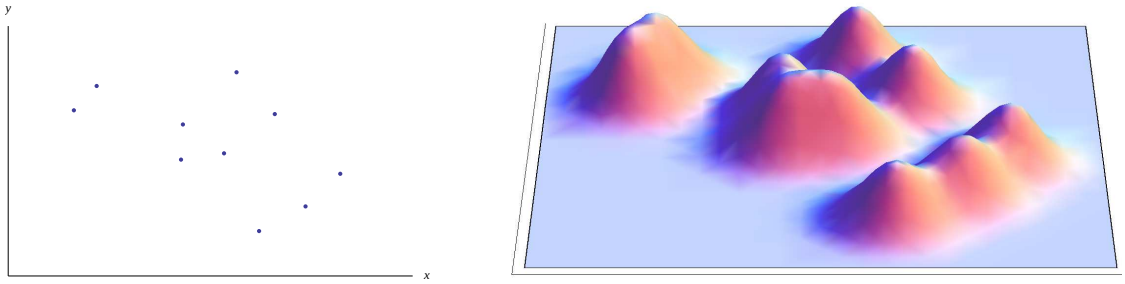


Figure 2.4: Visualization of a discrete point set (left) and its related continuous representation (right) based on the probabilistic representation in terms of mixture models of eqn. (2.12).

read

$$m(x;\theta,\sigma_m) = \sum_{j=1}^{m} \pi_j K\left(x, \theta(v_j); \sigma_m\right) \ , \tag{2.12a}$$

$$s(x;\sigma_s) = \sum_{i=1}^{n} \tau_i K\left(x, u_i; \sigma_s\right) \ , \tag{2.12b}$$

with $\tau_i, \pi_j \geq 0$, $\sum_{i=1}^{n} \tau_i = \sum_{j=1}^{m} \pi_j = 1$, and $K(\cdot,\cdot;\sigma)$ denoting a kernel function whose scale is controlled by the parameter σ. In [70], Gaussian mixture distributions with covariance σI where chosen, such that the kernel reads

$$K(x,y;\sigma) = \frac{1}{(2\pi\sigma^2)^{\frac{n}{2}}} \exp\left(-\frac{1}{2\sigma^2}\left\|x-y\right\|_2^2\right) \ , \tag{2.13}$$

where $x, y \in \mathbb{R}^n$. Despite of the dependency of K on its scale σ, to improve readability of the formulas we simply denote $s(x;\sigma_s), m(x;\theta,\sigma_m)$ by $s(x), m(x;\theta)$, respectively.

The representation of discrete point sets in terms of (2.12) illustrated in Fig. 2.4, casts the problem of aligning point sets to the task of minimizing the distance between mixture distributions on the space of Euclidean transformations.

A generic distance measure for probability distributions, introduced by Basu et al. [6], is given by the density power divergence

$$d_\alpha(s,\, m) = \int_x \left(\frac{1}{\alpha} s(x)^{1+\alpha} - \left(1+\frac{1}{\alpha}\right) s(x)\, m(x;\theta)^\alpha + m(x;\theta)^{1+\alpha}\right)$$

with $\alpha \geq 0$. The most prominent realizations of this parametrized family of distance measures include the ℓ_2 distance ($\alpha = 1$)

$$d_{\ell_2}(s,m) = \int_x \left\|s(x) - m(x;\theta)\right\|_2^2$$

as well as the Kullback-Leibler divergence ($\alpha \to 0$)

$$d_{KL}(s,m) = \int_x s(x) \log \frac{s(x)}{m(x;\theta)} \ .$$

While both the ℓ_2 and the Kullback-Leibler distance are widely applied within literature, there are significant differences in terms of performance and robustness sketched in Fig. 2.5. In the following, we analyze both measures in detail.

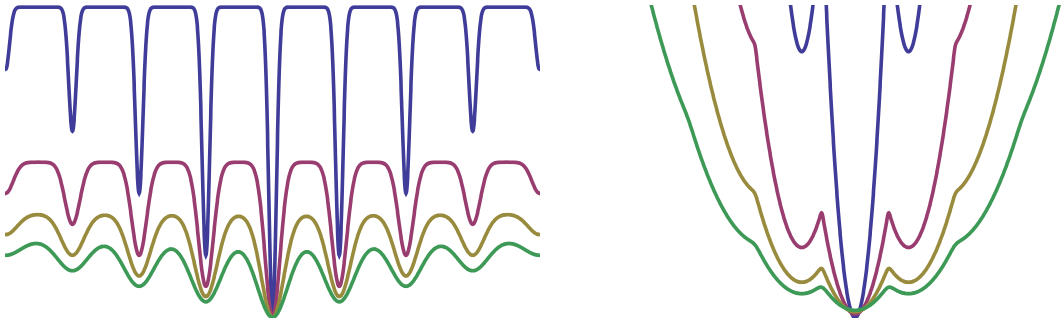


Figure 2.5: ℓ_2 distance measure (left) and Kullback-Leibler divergence (right) for two one-dimensional mixture distributions with respect to translation for varying σ. We can clearly see that with increasing σ, the KL distance effectively convexifies the objective function, leading to increased robustness of registration.

2.3.1 ℓ_2 Distance Based Registration

Minimizing the ℓ_2 distance between two probability density functions with respect to $\theta \in \mathsf{SE}(3)$ amounts to solve

$$\min_{\theta \in \mathsf{SE}(3)} \int_x \left(s(x)^2 - 2s(x)m(x;\theta) + m(x;\theta)^2 \right) \, , \tag{2.14}$$

where the first part is independent of the Euclidean transformation and can be omitted from further considerations. Additionally, using the concrete representation of $m(x, \theta)$ in terms of Gaussian mixtures, it turns out that the last part of (2.14) is independent of $\theta \in \mathsf{SE}(3)$, too. This can be seen by inserting the kernel's definition (2.13) into (2.14). Due to the well known properties of the Gaussian kernels, i.e. the product of two Gaussian kernels is except of normalization a Gaussian kernel, the latter part of (2.14) reads as

$$\sum_{j=1}^{m} \sum_{l=1}^{m} \pi_j \pi_l K\left(\theta(v_j), \theta(v_l); 2\sigma_m\right) K\left(x, \frac{1}{2}\big(\theta(v_l) + \theta(v_j)\big); \frac{1}{2}\sigma_m\right) \, . \tag{2.15}$$

Owing to the fact that Euclidean transformations preserve the metric, i.e. $\forall\, x, y \in \mathbb{R}^3$, $\|x - y\|_2 = \|\theta(x) - \theta(y)\|_2$, varying $\theta \in \mathsf{SE}(3)$ does not affect the first part of (2.15). Moreover, computing the integral of (2.15) according to (2.14) causes integration of the latter Gaussian kernel of (2.15) exclusively, that is 1 by definition for any value of $\theta \in \mathsf{SE}(3)$. Hence, optimizing the ℓ_2 distance simplifies to minimize the central part of (2.14).

According to the properties of the representation in terms of Gaussian mixtures, the simplified objective function that corresponds to the ℓ_2 distance (2.14) is given in closed form as

$$\sum_{i=1}^{n} \sum_{j=1}^{m} \tau_i \pi_j K\left(u_i, \theta(v_j); \sigma_m + \sigma_s\right)$$

that in general admits fast evaluation compared to the continuous formulation (2.14) requiring numerical integration.

In contrast, the Kullback-Leibler divergence does not admit closed form evaluation in general. Thus, despite of the high non-convexity of (2.14), depicted on the left hand side of Fig. 2.5, for the problem of Euclidean point set registration, minimizing the ℓ_2 distance is typically preferred [70].

2.3.2 Kullback-Leibler Divergence Based Registration

To benefit from the convexity and its related robustness of the Kullback-Leibler (KL) divergence, evaluating the natural distance between two probability distributions [33] is typically done using Monte-Carlo simulations [98] or approximating the divergence measure [55]. These approximate evaluations, however, typically permit the application of sophisticated higher order optimization techniques and increase of computational complexity.

In the following, we will see that for the special case of Euclidean point set registration and under reasonable assumptions a closed form expression of the KL divergence exists. This allows to efficiently evaluate the distance measure and enables the application of higher order optimization techniques.

Minimizing the KL divergence for Euclidean point set registration amounts to determine $\theta \in \mathsf{SE}(3)$, by optimizing

$$\begin{aligned} d_{KL}(s,m) &= \int_x s(x) \log \frac{s(x)}{m(x;\theta)} \\ &= \int_x s(x) \log s(x) - \int_x s(x) \log m(x;\theta) \ , \end{aligned} \tag{2.16}$$

where the first part does not depend on the Euclidean transformation and can be omitted from further considerations.

Using the Gaussian mixture distribution of (2.12), the continuous point set representations contain scale parameters σ_s, σ_m, respectively. These parameters model noise and uncertainty of the point correspondences. Hence, it is a reasonable assumption to confine this degree of freedom to either the scene or the model. Taking into account, without loss of generality, uncertainty and noise only in the representation of the model points, i.e. $\sigma_s \to 0$, the continuous representation of the scene distribution reduces to a mixture of Dirac deltas

$$s(x) = \lim_{\sigma_s \to 0} s(x) = \sum_{i=1}^{n} \tau_i \delta(x - u_i) \ ,$$

where $\delta(x) = \infty$ if $x = 0$ and $\delta(x) = 0$ otherwise. Insertion into the second part of (2.16) and using the model representation (2.12) with Gaussian kernels (2.13) leads to the problem

$$\max_{\theta \in \mathsf{SE}(3)} \sum_{i=1}^{n} \tau_i \log \sum_{j=1}^{m} \pi_j \exp\left(-\frac{1}{2\sigma_m^2} \left\| u_i - \theta(v_j) \right\|_2^2 \right) \ , \tag{2.17}$$

where we omitted all normalizing constants.

Thus, minimizing the KL divergence between two Gaussian mixture distributions on the space of Euclidean transformations yields an objective function that, under reasonable assumptions, can be evaluated in closed form. Additionally, compared to (2.14) optimizing (2.17) convexifies the objective with increasing σ_m and leads to increased robustness of registration, as sketched in Fig. 2.5.

2.4 Relations to Correspondence-Dependent Functions

Determining the optimal alignment of two given point sets in terms of (2.17), yields the optimization of a smooth objective defined on the space of Euclidean transformations. In

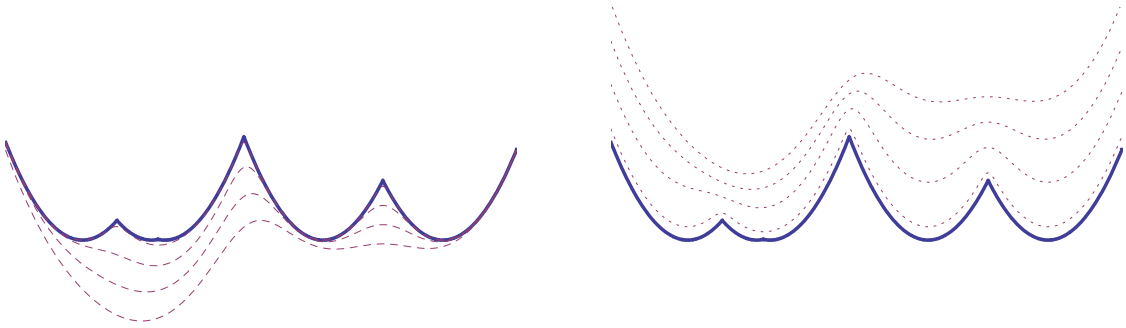


Figure 2.6: Visualization of the lower (left) and upper (right) bounds of the objective of the ICP approach (2.1) (cf. Fig. 2.3) specified by the log-exponential function for varying values of σ_m. Despite the convergence of the bounds with $\sigma_m \to 0$, both lower and upper bounds provide accurate and smooth approximations of the Euclidean matching problem.

the following, we will show that for specific choices of the parameters involved, minimizing the Kullback-Leibler divergence turns out to be a generalized objective functional of the aforementioned correspondence dependent registration approaches such as ICP, EM-ICP and the ICP relaxation presented in Sec. 2.2.

2.4.1 Relation to ICP

To illustrate the relation of the objective functional (2.17) to the cumulative squared point-to-point distance of corresponding samples given by (2.1), we highlight the role of the scale parameter σ_m.

Under the assumption of having uniform weights $\pi_j = \frac{1}{m}, \tau_i = \frac{1}{n}$, $\forall\, i = 1, \ldots, n, j = 1, \ldots m$ dropping all constants yields (2.17) to be the sum of well known log-exponential functions [99]

$$\log \sum_{j=1}^{m} \exp\left(-\frac{1}{2\sigma_m^2} \left\| u_i - \theta(v_j) \right\|_2^2 \right) ,$$

that provide (up to scaling) for all $\sigma_m > 0$ upper and lower bounds on the "max" function given by

$$\begin{aligned} &\sigma_m^2 \log \sum_{j=1}^{m} \exp\left(-\frac{1}{2\sigma_m^2} \left\| u_i - \theta(v_j) \right\|_2^2 \right) - \sigma_m^2 \log m \\ &\leq \max_{j=1,\ldots,m} \left\{ -\frac{1}{2} \left\| u_i - \theta(v_j) \right\|_2^2 \right\} \\ &\leq \sigma_m^2 \log \sum_{j=1}^{m} \exp\left(-\frac{1}{2\sigma_m^2} \left\| u_i - \theta(v_j) \right\|_2^2 \right) . \end{aligned}$$

As visualized in Fig. 2.6, these bounds illustrate that (2.17) is a smoothed version of (2.1) and (including scaling) uniformly approximates for $\sigma_m \to 0$ the cumulative squared distance of transformed model points $\theta(v_j)$ to measurement samples u_i.

This result also shows that the log-exponential functional (2.17) includes as special case the objective function suggested in [55], i.e. the approximation of the KL-distance using the sum of "min" functions.

Although, the optimal configuration of (2.17) is a solution of (2.1), in general it is beneficial to let $\sigma_m > 0$, as for increasing values of σ_m the objective function (2.17) not only is convexified but also exhibits a less biased global optimum (see Fig. 2.7). Thus, using a finite value of σ_m allows to properly capture the uncertainties of implicit correspondence and increases the robustness to noise and outlying structures occurring in most vision applications.

Similar to these considerations, we can also establish a connection to outlier rejection strategies previously introduced in Sec. 2.2.1. Using an additional background kernel $K_{bg} \propto \exp(-\frac{1}{\sigma^2}\, c)$ in (2.17) for a user parameter $c \geq 0$, the sum of log-exponential functions provides upper and lower bounds to

$$\max_{j=1,\dots,m} \left\{ -\frac{1}{2} \left\| u_i - \theta(v_j) \right\|_2^2 , \, -c \right\} ,$$

that is closely related to the hard outlier rejection of Sec. 2.2.1.

2.4.2 Relation to ICP-Relaxation

The fact that (2.17) also provides a generalization to the smoothed variant of ICP defined by (2.5) is primarily based on the following theorem.

Theorem 2.2. *[117] For any given $d_j \in \mathbb{R}$, $j = 1, \dots m$ and $\pi \in \Delta$, with $\Delta = \{w \in \mathbb{R}^m | \sum_{j=1}^m w_j = 1, w_j \geq 0\}$*

$$\log \sum_{j=1}^{m} \pi_j e^{d_j} = \max_{w \in \Delta} \left\{ \sum_{j=1}^{m} w_j \, d_j - \sum_{j=1}^{m} w_j \log \frac{w_j}{\pi_j} \right\} .$$

Thus, the log-exponential function turns out to be directly related to the entropy barrier function used in [96]. To apply Thm. 2.2 to (2.17), let us assume to have uniform weights $\tau_i = \frac{1}{n}, \pi_j = \frac{1}{m}, \forall\, i = 1, \dots n,\, j = 1, \dots, m$, respectively. Moreover, abstraction of the cumulative log-exponential function (2.17) by replacing the weighted negative squared Euclidean distance $-\frac{1}{2\sigma_m^2} \| u_i - \theta(v_j) \|_2^2$ by $d\big(u_i, \theta(v_j)\big)$ lets the objective functional being a composition of the log-exponential and d given by

$$\sum_{i=1}^{n} \log \sum_{j=1}^{m} \exp\left(d\big(u_i, \theta(v_j)\big) \right) .$$

This directly applies to Thm. 2.2 such that we obtain an equivalent formulation of (2.17) given by

$$\max_{\theta \in \mathsf{SE}(3)} \sum_{i=1}^{n} \max_{w_i \in \Delta} \left\{ \sum_{j=1}^{m} w_{ij} \, d\big(u_i, \theta(v_j)\big) - \sum_{j=1}^{m} w_{ij} \log \, w_{ij} \right\} , \tag{2.18}$$

where again we dropped all constants. Then, back insertion of the definition of $d\big(u_i, \theta(v_j)\big)$, transfers (2.18) into a minimization problem, and by substituting $2\sigma_m^2$ for λ, we directly obtain (2.5).

This shows that each optimizer of (2.5) is also an optimal configuration of (2.17) and vice versa. Additionally, to obtain an optimizer of (2.17), we can apply the two step approach of Alg. 2.

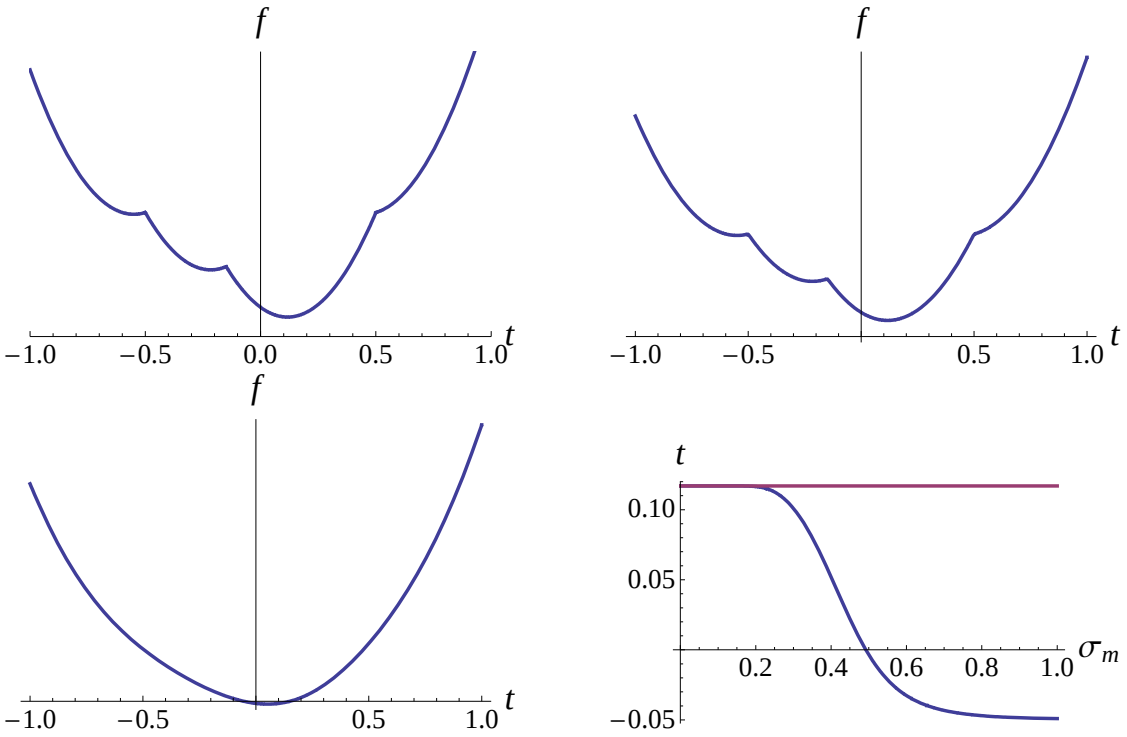


Figure 2.7: Comparison of the smooth objective functional (2.17) with the criterion (2.1) where the correspondence function $\kappa(i)$ assigns the observation u_i to the closest model point $v_{\kappa(i)}$. The "model" consists of two scalar values $v_1 = 0\,,\ v_2 = 1$, and we assume to have observed the same two values as u_1, u_2, and a single additional value $u_3 \in (0, 1)$ at some arbitrary position in between as sketched in Fig. 2.2. We inspect the optimal registration in terms of the scalar translational value t – which obviously is $t = 0$ – in terms of both objective functions. Top left: Objective function (2.1) not only is non-convex but also has a biased global minimum. Top right, bottom left: For increasing values of σ_m the objective function (2.17) not only is convexified but also exhibits a less biased global optimum. Bottom right: Position of the global optimum as a function of σ_m. For a significant range of this parameter value minimizing (2.17) give a more accurate result. The constant value on the top corresponds to the global minimum of (2.1) depicted on the upper left panel.

2.4.3 Relation to EM-ICP

The expectation maximization algorithm is a workhorse for many statistical estimation problems. Below we provide the background, allowing to interpret the EM-ICP approach presented in Sec. 2.2.2 as special instance of the problem of maximizing the log-exponential defined by (2.17).

Let us assume $\tau_i = \frac{1}{n}, i = 1, \dots n$ to be constant and the objective functional being represented in terms of the composition of the log-exponential and $d\big(u_i, \theta(v_j)\big) = -\frac{1}{2\sigma_m^2}\|u_i - \theta(v_j)\|_2^2$, as analyzed previously. Then, direct application of Thm. 2.2 yields

the equivalent optimization problem

$$\max_{\theta\in\mathsf{SE}(3)} \sum_{i=1}^{n} \max_{w_i\in\Delta} \left\{ \sum_{j=1}^{m} w_{ij}\, d\big(u_i, \theta(v_j)\big) - \sum_{j=1}^{m} w_{ij} \log\, w_{ij} + \sum_{j=1}^{m} w_{ij} \log\, \pi_j \right\} . \tag{2.19}$$

Using the well known property of the log-function, i.e. $x \log\, y + x \log\, z = x \log\,(yz)$, we obtain

$$\max_{\theta\in\mathsf{SE}(3)} \max_{\substack{w_i\in\Delta \\ i=1,\ldots,n}} \sum_{i=1}^{n} \left\{ \sum_{j=1}^{m} w_{ij} \log \left(e^{d\left(u_i,\theta(v_j)\right)} \pi_j \right) - \sum_{j=1}^{m} w_{ij} \log\, w_{ij} \right\} ,$$

where optimization with respect to $w_i \in \Delta$ corresponds to the computation of the expectation step defined in (2.8). This can be seen by letting $w_{ij} = p(\omega_i = j|u_i, \hat{\theta}, v_1, \ldots, v_m)$ and (up to scaling) $\exp\big(d\big(u_i, \theta(v_j)\big)\big)\pi_j$ be the joint probability of measuring scene sample u_i and the correspondence, i.e. $p(u_i, \omega_i = j|\theta, v_1, \ldots, v_m)$, that can be decomposed according to (2.9).

Moreover, assuming to have fixed $w_i \in \Delta$, maximizing (2.19) with respect to $\theta \in \mathsf{SE}\,(3)$ and π_j such that $\sum_{j=1}^{n} \pi_j = 1$, amounts to solve (2.11) being the maximization steps of EM-ICP. Thus, the formulation of the expectation maximization approach in terms of an Euclidean point set registration problem can be interpreted as a specific version of (2.17).

2.5 Summary and Further Work

Summary

In this chapter we presented and analyzed different approaches to model the problem of Euclidean point set registration. This includes the established Iterative Closest Point procedure that is based on the cumulative squared point-to-point distance of corresponding samples where the optimal configuration is obtained by iteratively updating the Euclidean transformation followed by readjusting the correspondences.

To address the drawbacks of this approach, i.e. sensitivity to poor initializations and the lack of robustness to noise, in Sec. 2.2 we presented two extensions derived from relaxation of the correspondence assignment function as well as probabilistic modelling of the uncertainties. For exhaustive numerical evaluations of sensitivity to noise and to poor initial configurations of these approaches, we refer to Chap. 5.

However, the major drawback concerning the representation of the problem persists, i.e. explicit correspondences increase the non-convexity and non-smoothness of the objective function, and gaining insight into the optimization problem is hampered by the complicated structure of the domain of optimization comprising both Euclidean transformations and correspondence variables. Thus, in Sec. 2.3 we tackled the problem of determining explicit correspondences by representing model samples and scene measurements in terms of mixture distributions, where the optimal Euclidean transformation is derived from a parametric family of distance measures that include the ℓ_2 and the Kullback-Leibler distance as special instances.

We proposed to use the natural distance measure between probability distribution, i.e. the KL divergence. Additionally, we showed that under reasonable assumptions, in the case of Euclidean registration this measure can be evaluated in closed form without

requiring numerical integration and finally turns out to consist of the sum of well-known log-exponential functions. Addressing the Euclidean alignment problem in terms of measuring the KL divergence between mixture models, we showed in Sec. 2.4 that minimizing the sum of log-exponential functions provides lower and upper bounds for objective defined by ICP and can (for specific choices of the parameters) be optimized using EM-ICP and the ICP-relaxation.

Further Work

Despite of the generalization of this novel formulation for the problem of Euclidean registration, there are three major issues that have to be addressed. Due to the removal of explicit correspondences, determining the Euclidean transformation cannot be obtained in closed form. However, because of the smoothness, it enables to apply higher order optimization techniques. Thus, we have to investigate techniques to numerically optimize an objective defined on the space of rigid body transformations. This problem will be addressed in Chap. 3.

Moreover, formulating the problem of Euclidean alignment in terms of minimizing mixture distribution only removes the non-smoothness of the objective function. Hence, convergence to a stationary point only guarantees local optimality. Especially in the setting of having multiple objects in the scene (see Fig. 1.1) many local optima can appear such that accurate initializations are important to guarantee proper registration results. The problem of determining likely positions of objects in the scene will be investigated in Chap. 4.

Finally, while there are dozens of extensions to speed up the run-time of ICP, such as using efficient data structures [11] or applying multi-resolution schemes [72], evaluating the sum of log-exponential functions requires $O(n\,m)$ operations, despite of its closed form expression. Due to the steadily increasing resolution of modern scanning devices, in further work we will focus on fast evaluation of the objective without destroying significant structural and geometric properties of the corresponding point sets.

Chapter 3
Geometric Optimization on $\mathsf{SE}(3)$

3.1 Overview

3.1.1 Introduction and Motivation

Determining the optimum of an objective function forms the basis of most computer vision algorithms, where the concrete representation of the objective as well as the domain of definition influence the complexity of the problem [18].

In this chapter, we investigate different optimization techniques for determining the solution of problems with geometric constraints, i.e. the domain of definition is a smooth manifold as sketched in Fig. 3.1. Problems of this kind arise in different fields of application such as object localization [111, 135], camera calibration [62], or reverse engineering [77]. In concrete terms, we consider problems of type

$$\min_{Y \in \mathsf{SE}(3)} f(Y) \ ,$$

where $f : \mathsf{SE}(3) \mapsto \mathbb{R}$ is assumed to be a smooth function and $\mathsf{SE}(3)$ denotes the space of Euclidean transformations. To determine the optimal configuration $Y \in \mathsf{SE}(3)$ minimizing f, we propose different optimization approaches based on higher order geometric integration techniques and investigate a novel Newton-like algorithm that fully exploits the underlying properties of the smooth manifold.

3.1.2 Related Work and Contribution

The minimizer of an objective function defined on a continuous domain, is typically computed using optimization methods such as gradient descent or Newton-like approaches. However, if the underlying domain is a smooth manifold as depicted in Fig. 3.1, applying such schemes significantly differs from standard techniques.

A common approach to cope with optimization problems whose solution lies on a smooth manifold, is to represent the manifold by a local parametrization. In the special case of Euclidean transformations such representations include Euler angles [47, 122], quaternions or dual-number quaternions [66], or transformation matrices. While Euler angles allow to reformulate the constrained optimization problem in terms of an unconstrained counterpart, this representation suffers from singularities. In contrast, quaternion representation copes with singularities at the costs of additional quadratic equality constraints and the more involved Hamiltonian algebra.

For optimization and numerical algorithm design, working with the matrix representation of the group of Euclidean transformation is more appropriate [2].

While direct search algorithms such as Nelder-Mead optimization [79] can be extended to these representations, more sophisticated approaches using first or second order information of the objective are typically superior in terms of speed of convergence and are easier to characterize mathematically.

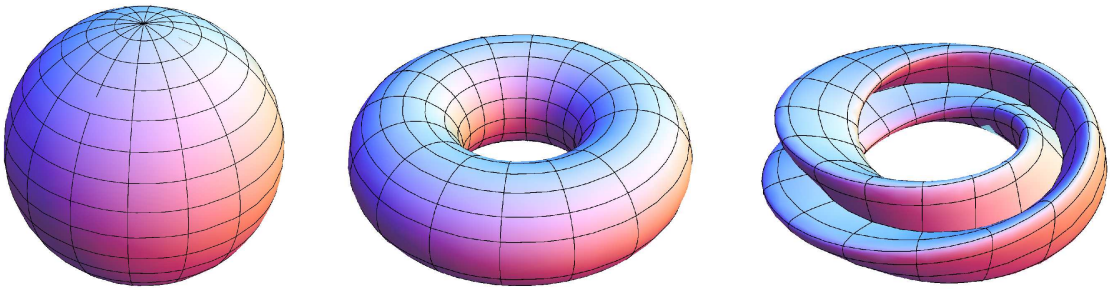


Figure 3.1: Visualization of exemplary smooth manifolds embedded in $\mathbb{R}^3$: a sphere, a torus and the so-called Klein bagel, that is a figure-8 torus with a 180 degree Mobius twist inserted.

Based on the abstract work of Gabay [50], Smith [112] outlines a rather general mathematical framework for optimization algorithms such as gradient descent, conjugate gradients, and Newton's method on Riemannian manifolds. Edelman et al. [41], Hüper and Trumpf [68], and Owren and Welfert [91] presented specific continuous optimization methods by considering matrix manifolds related to the orthogonal group (Grassmann and Stiefel manifolds). Related to this work, Adler et al. [3], for instance, proposed a corresponding Newton-like algorithm for human spine alignment.

In the case of optimization with respect to Euclidean transformations, the problem is typically addressed by either representing the objective in terms of an orthogonal matrix [80] exclusively, that is only possible for few objectives, or by optimizing with respect to translation and rotation separately [62]. However, as translation and rotation are interdependent in general, such approaches are likely to suffer from more restrictive numerical convergence properties.

Pottmann et al. [94] suggested an optimization method for iterative registration based on successive local first- and second-order approximation of the manifold of Euclidean transformations at the current iterate. Related problems of computer vision, including multiple point set alignment and tracking, were studied e.g. by Krishnan et al. [77], Taylor and Kriegman [116], Benhimane and Malis [9], and Drummond and Chipolla [40]. We consider the geometric optimization approach [94] in more detail and work out differences to the approach suggested in this chapter.

Finally, we refer to very recent work [64, 81, 90] on global optimization approaches to solve the problem of model-based pose estimation. While all of them apply branch and bound techniques for exploring the pose parameter space, Hartley and Kahl [64] as well as Olsson et al. [90] require very special settings, i.e. explicit correspondence of scene points to convex model parts. In contrast, Li and Hartley [81] works without such correspondences. However concerning applications such as point set alignment, the major problem with these works is that run-time badly scales with the problem size, e.g. about 20 min for 200 points. Therefore, these sophisticated approaches are unfortunately not applicable to realistic industrial settings with hundreds of points.

In this chapter, we investigate two conceptually different ways to determine the optimum of a smooth objective defined on the group of Euclidean transformations. By extending Runge-Kutta-type integration directly to the group of Euclidean transformations, we infer the optimal configuration by computing the gradient flow directly on the related manifold. Additionally, we point out the relation to standard methods such as gradient descent.

Moreover, we devise and study a second-order optimization method that fully exploits the geometry of the manifold of Euclidean transformations in order to optimize a smooth objective functional. Finally, we analyze the differences to state-of-the-art second-order approaches. Contents of this chapter have been published previously in [22, 23].

3.1.3 Organization

To determine the optimum of an objective function that is defined on a smooth manifold requires detailed analysis of the underlying geometry. In Sec. 3.2 we summarize properties of the space of rigid body transformations such as group operations, tangent spaces and derivatives.

Using these properties, in Sec. 3.3 we devise geometric integration techniques to reconstruct the flow induced by the gradient of the objective while staying on the manifold of Euclidean transformations. This guarantees to determine the optimizer of an objective, given a proper initialization.

However, as these algorithms typically converge slowly, in Sec. 3.4 we propose a novel Newton-like optimization scheme that fully exploits the properties of the underlying manifold while using up to second-order information of the objective functional. We analyze relations to state-of-the-art approaches and comment on benefits and drawbacks of this approach. Finally, we conclude in Sec. 3.5 and point out further work.

3.2 The Geometry of the Euclidean Group

Determining the optimizer of an objective functional that evolves from a curved space, i.e. a manifold, requires to analyze the detailed structure of the domain of definition.

In the sequel, we briefly review some general properties of a differentiable manifold such as curves and tangents, and analyze the concrete realizations of these concepts concerning Euclidean transformations represented as a matrix group. Throughout this section, we assume all functions and mappings to be differentiable of arbitrary order. For further details on differentiable manifolds, we refer to [37, 78, 113].

3.2.1 Differentiable Manifolds

Informally, a smooth, differentiable d-dimensional manifold can be defined as a smooth surface $\mathcal{M} \subset \mathbb{R}^n$, covered by a "suitable" collection of patches $\mathcal{S}_i$ that locally look like the Euclidean space $\mathbb{R}^d$. Mathematically, this yields the following definition illustrated by Fig. 3.2.

Definition 3.1. *A d-dimensional, differentiable manifold is a set $\mathcal{M}$ and a family of injective mappings $\phi_i : S_i \subseteq \mathbb{R}^d \mapsto \mathcal{S}_i \subseteq \mathcal{M}$ such that*

1. $\bigcup_i \phi_i(S_i) = \mathcal{M}$
2. *for any pair i, j with $\phi_i(S_i) \cap \phi_j(S_j) = \mathcal{W} \neq \emptyset$, the sets ${\phi_i}^{-1}(\mathcal{W}), {\phi_j}^{-1}(\mathcal{W})$ are open in $\mathbb{R}^d$ and the mapping ${\phi_i}^{-1} \circ \phi_j$ is differentiable.*

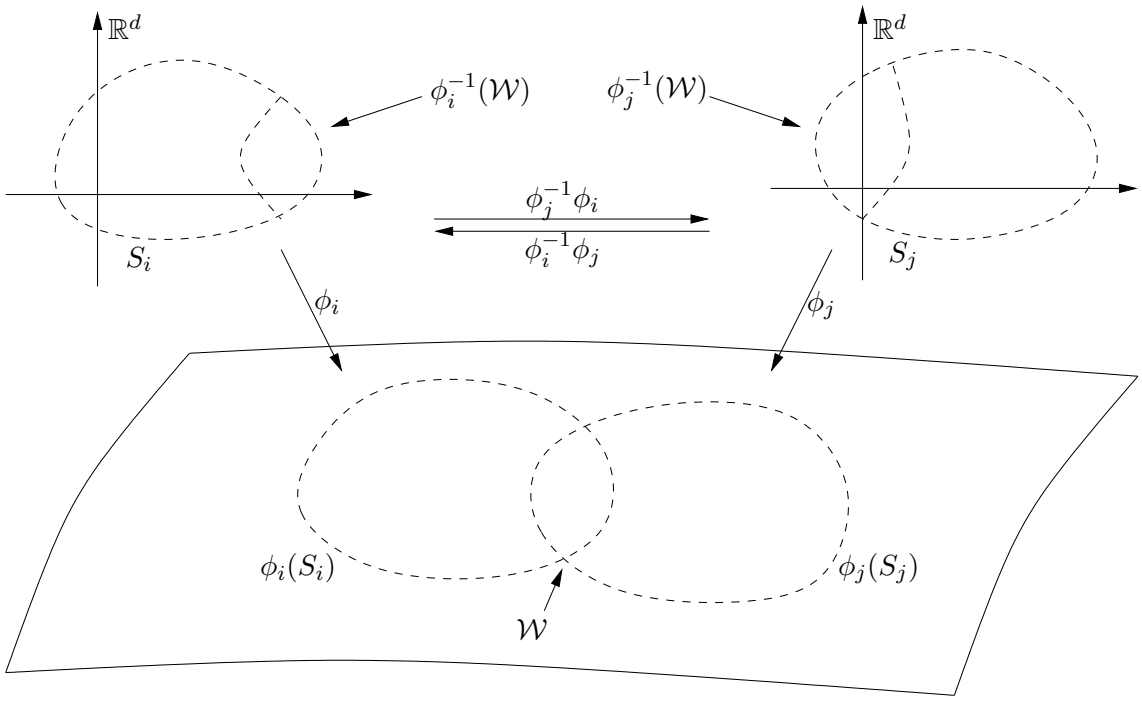


Figure 3.2: Sketch of a d-dimensional, differentiable manifold that is partially covered by two sets $\phi_i(S_i), \phi_j(S_j)$, such that $S_i, S_j \in \mathbb{R}^d$, $\phi_i(S_i) \cap \phi_j(S_j) \neq \emptyset$ and the composed mappings $\phi_j^{-1}\phi_i, \phi_i^{-1}\phi_j$ are differentiable.

Each pair (S_i, ϕ_i) is called a local parametrization of $\mathcal{M}$ at $Y \in \phi_i(S_i)$ and $\phi_i(S_i)$ is the coordinate neighborhood of $Y \in \mathcal{M}$. A family $\{(S_i, \phi_i)\}$ satisfying the properties of Def. 3.1 is called a differentiable structure [37].

The simplest example of a smooth, differentiable manifold is the Euclidean space $\mathbb{R}^n$, where the family of injective mappings is given by the identity map. In this case, the inverse mappings of two overlapping open sets are open and overlap smoothly.

Next, we consider the definition of tangent vectors to differentiable manifolds, i.e. directions on the curved space.

Tangent Spaces

The subsequent steps will motivate the definition of a tangent vector. Let $\gamma : (-\epsilon, \epsilon) \mapsto \mathbb{R}^d$ be a curve in $\mathbb{R}^d$ with $\gamma(t) = \big(x_1(t), \ldots, x_d(t)\big)$. The derivative of $\gamma(t)$ with respect to t, evaluated at $t = 0$ is given by

$$\dot{\gamma}(0) = (\dot{x}_1(0), \ldots, \dot{x}_d(0)) \in \mathbb{R}^d \, .$$

Now, let h be a function defined in the local neighborhood of $Y \in \mathcal{M}$. Restricting h to the curve γ allows us to express the directional derivative of the composite function as

$$\lim_{t \to 0} \frac{d}{dt} (h \circ \gamma) = \left(\sum_i \dot{x}_i(0) \frac{\partial}{\partial x_i} \right) h \, ,$$

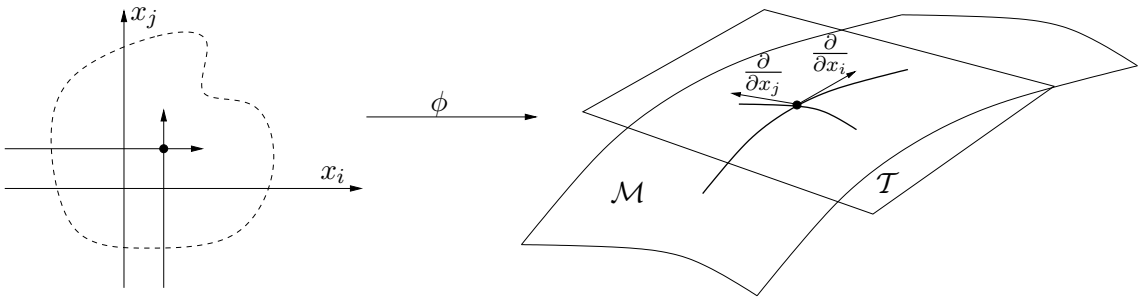


Figure 3.3: Illustration of tangent space $\mathcal{T}$ parametrized such that the basis of $\mathcal{T}$ equals the tangents of the corresponding coordinate curves.

where $\frac{\partial}{\partial x_i} h$ refers to the derivative of h with respect to x_i. Hence, the directional derivative is an operator on functions that depends on $\dot{\gamma}(0)$. This characteristic property allows to define tangent vectors on a manifold.

Definition 3.2. *Let $\mathcal{M}$ be a differentiable manifold, and $\gamma : (-\epsilon, \epsilon) \mapsto \mathcal{M}$ a curve in $\mathcal{M}$ such that $\gamma(0) = Y \in \mathcal{M}$. Let $\mathcal{D}$ be the set of functions on $\mathcal{M}$ that are differentiable at Y. Then, the map $\dot{\gamma}(0) : \mathcal{D} \mapsto \mathbb{R}$ given by*

$$\dot{\gamma}(0) f = \lim_{t \to 0} \frac{d}{dt} \left(f \circ \gamma \right), \qquad f \in \mathcal{D}$$

is called a tangent vector of $\mathcal{M}$ at Y.

The set of all tangent vectors to $\mathcal{M}$ at Y is denoted by $\mathcal{T}_Y$ and is called the *tangent space* at Y. Moreover, the tangent space is closed under addition and scalar multiplication and forms a vector space of dimension d [82]. The collection of all tangent spaces $\mathcal{T}_Y$ for all $Y \in \mathcal{M}$ is called the tangent bundle and is denoted by $\mathcal{TM}$.

Choosing a parametrization $\phi : S \mapsto \mathcal{S} \subseteq \mathcal{M}$, where $S \subseteq \mathbb{R}^d$ and $Y \in \mathcal{S}$, such that $\phi^{-1} \circ \gamma(t) = \big(x_1(t), \ldots, x_d(t)\big)$ allows $\dot{\gamma}(0)$ to be expressed in the parametrization x by

$$\dot{\gamma}(0) = \sum_i \dot{x}_i(0) \left(\frac{\partial}{\partial x_i} \right) . \tag{3.1}$$

Thus, the choice of the parametrization determines the associated basis of the tangent vector space $\mathcal{T}_Y$. This is illustrated in Fig. 3.3. Again, in the simple case of interpreting $\mathbb{R}^n$ as a Euclidean manifold, where the parametrization can be set as the identity, the tangent space at a certain point turns out to be $\mathbb{R}^n$ itself.

Tangent Vector Fields

Due to the geometry of the underlying manifold $\mathcal{M}$, tangent spaces at distinct points differ in general. Thus, to travel along a manifold properly, we need to analyze the relations between tangent vectors in terms of vector fields defined on $\mathcal{TM}$.

Definition 3.3. *A tangent vector field Ψ on a differentiable manifold $\mathcal{M}$ is a correspondence that associates to each point $Y \in \mathcal{M}$ a tangent vector $\Psi(Y) \in \mathcal{T}_Y$. The vector field Ψ is differentiable, if the mapping $\Psi : \mathcal{M} \mapsto \mathcal{TM}$ is differentiable.*

Using the local parametrization of a tangent vector, given by (3.1), allows to write the tangent vector field locally as

$$\Psi(Y) = \sum_i \psi_i(Y) \left(\frac{\partial}{\partial x_i} \right) ,$$

where $\psi_i : \mathcal{M} \mapsto \mathbb{R}$ is a function defined in the local neighborhood of Y. Hence, Ψ is differentiable if and only if all functions ψ_i are differentiable.

The set of all differentiable vector fields on $\mathcal{M}$ is denoted by $\mathfrak{X}(\mathcal{M})$. Closely related to tangent vectors, vector fields can be interpreted as mapping defined on the set of differentiable functions.

Affine and Riemannian Connections

Let $\Psi \in \mathfrak{X}$ be a vector field on $\mathcal{M}$ and $Y_1, Y_2 \in \mathcal{M}$ with $Y_1 \neq Y_2$. Due to the geometry of $\mathcal{M}$, $\Psi(Y_1)$ and $\Psi(Y_2)$ are typically not element of the same tangent space. Thus, to fully analyze Ψ, we have to consider the connection between nearby tangent spaces. However, this requires to introduce one of the most fundamental concepts in differential geometry namely the affine connection, i.e. the changing of tangent vector fields when traveling along a curve on the manifold that is induced by a vector field.

Definition 3.4. *An affine connection $\overline{\nabla}$ on a differentiable manifold $\mathcal{M}$ is defined as a mapping*

$$\begin{array}{rccc} \overline{\nabla}\colon & \mathfrak{X}(\mathcal{M}) \times \mathfrak{X}(\mathcal{M}) & \mapsto & \mathfrak{X}(\mathcal{M}) \\ & (\Phi, \Psi) & \mapsto & \overline{\nabla}_\Phi \Psi \end{array}$$

which satisfies for $\Phi, \Psi, \Upsilon \in \mathfrak{X}(\mathcal{M})$ and $g, h \in \mathcal{D}$ the following properties:

$$\begin{array}{lrcl} 1. & \overline{\nabla}_{g\Phi + h\Psi} \Upsilon & = & g\overline{\nabla}_\Phi \Upsilon + h\overline{\nabla}_\Psi \Upsilon \ , \\ 2. & \overline{\nabla}_\Phi (\Psi + \Upsilon) & = & \overline{\nabla}_\Phi \Psi + \overline{\nabla}_\Phi \Upsilon \ , \\ 3. & \overline{\nabla}_\Phi (g\Psi) & = & g\overline{\nabla}_\Phi \Psi + \Phi(g)\Psi \ . \end{array}$$

Let us choose a system of local coordinates according to (3.1), with a local basis given by $\mathcal{L}_i = \frac{\partial}{\partial x_i}$ such that each $\Psi, \Phi \in \mathfrak{X}$ reads as

$$\Phi = \sum_i \phi_i(Y)\mathcal{L}_i \ , \quad \Psi = \sum_j \psi_j(Y)\mathcal{L}_j \ ,$$

respectively, where $\phi_i, \psi_j : \mathcal{M} \mapsto \mathbb{R}$ specify the coefficients of the vector field in terms of local functions on the manifold. In the sequel, it will be convenient to write ϕ_i, ψ_j instead of $\phi_i(Y), \psi_j(Y)$ to improve readability of formulas.

Then, according to Def. 3.4, the affine connection can be rewritten as

$$\begin{aligned} \overline{\nabla}_\Phi \Psi &= \overline{\nabla}_{\sum_i \phi_i \mathcal{L}_i} \sum_j \psi_j \mathcal{L}_j = \sum_i \phi_i \overline{\nabla}_{\mathcal{L}_i} \Big(\sum_j \psi_j \mathcal{L}_j \Big) \\ &= \sum_{i,j} \phi_i \psi_j \overline{\nabla}_{\mathcal{L}_i} \mathcal{L}_j + \sum_{i,j} \phi_i \mathcal{L}_i(\psi_j) \mathcal{L}_j \ , \end{aligned}$$

where setting $\overline{\nabla}_{\mathcal{L}_i} \mathcal{L}_j = \sum_k \Gamma_{ij}^k \mathcal{L}_k$, with Γ_{ij}^k being the so called Christoffel symbols, simplifies the affine connection to

$$\overline{\nabla}_\Phi \Psi = \sum_k \Big(\sum_{i,j} \phi_i \psi_j \Gamma_{ij}^k + \Phi(\psi_k) \Big) \mathcal{L}_k \ , \tag{3.2}$$

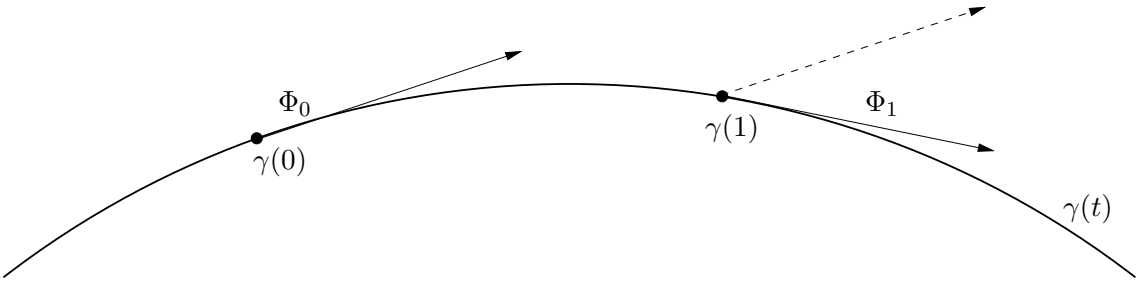


Figure 3.4: Parallel transport of the tangent vector Φ_0 along the curve $\gamma(t)$. While simple parallel moving in the Euclidean space typically cause non-tangential vectors (dashed), transporting the tangent Φ_0 according to $\overline{\nabla}$ along $\gamma(t)$ yields Φ_1 to be a tangent to $\gamma(t)$ at $\gamma(1)$.

which shows that $\overline{\nabla}_\Phi \Psi$ depends on ϕ_i, ψ_j as well as on $\Phi(\psi_k)$. For given vector fields Φ, Ψ, these expressions can be computed in an analytical manner such that the affine connection consequently can be uniquely specified in terms of the Christoffel symbols Γ_{ij}^k.

Additionally, using the definition of the affine connection, the concept of parallelism follows in a natural manner.

Definition 3.5. *Let $\mathcal{M}$ be a differentiable manifold with an affine connection $\overline{\nabla}$. A vector field Φ along a curve $\gamma(t) \in \mathcal{M}$ is called parallel when $\overline{\nabla}_\Psi \Phi = 0$, where Ψ is the tangent vector field induced by γ, i.e. $\Psi\big(\gamma(t)\big) = \dot{\gamma}(t)$.*

For every curve γ, where Ψ_0 is the tangent vector at $\gamma(t_0)$, Φ is called the *parallel transport* of Ψ_0 along γ. A curve γ, transporting its own tangent vector parallel to the curve is called a *geodesic*, i.e. $\overline{\nabla}_\Psi \Psi = 0$, where Ψ is the tangent vector field induced by γ. The principle of parallel transport is depicted in Fig. 3.4. The affine connection $\overline{\nabla}_\Psi \Phi$ along an induced vector field Ψ, is commonly called the *covariant derivative* of Φ.

While there are multiple affine connections for a smooth, differentiable manifold $\mathcal{M}$, some are of special interest with respect to geometric analysis. To derive these connections however, requires $\mathcal{M}$ to be endowed with a Riemannian metric [37].

Definition 3.6. *A Riemannian metric on a differentiable manifold $\mathcal{M}$ is a correspondence which associates to each point $Y \in \mathcal{M}$ an inner product $\langle \cdot, \cdot \rangle$ on the tangent space $\mathcal{T}\mathcal{M}$.*

Definition 3.7. *Let $\mathcal{M}$ be a smooth differentiable manifold with an affine connection $\overline{\nabla}$. A connection is said to be compatible with the metric, if for any smooth curve γ and any pair of parallel vector fields Φ, Ψ along γ, $\langle \Phi, \Psi \rangle = const$.*

Hence, whenever transporting tangent vectors along a smooth curve in $\mathcal{M}$, the inner product should not change. Additionally, a connection that is compatible with the metric lets us differentiate the inner product according to the usual product rule.

Definition 3.8. *A connection $\overline{\nabla}$ on a Riemannian manifold is compatible with the metric if and only if*

$$\Phi \langle \Psi, \Upsilon \rangle = \left\langle \overline{\nabla}_\Phi \Psi, \Upsilon \right\rangle + \left\langle \Psi, \overline{\nabla}_\Phi \Upsilon \right\rangle \ , \quad \Phi, \Psi, \Upsilon \in \mathfrak{X}(\mathcal{M}) \ .$$

Despite preserving the metric, it may happen that the tangent space twists around the curve $\gamma(t)$ when being parallel transported. This is commonly known as torsion. To guarantee that no torsion occurs, the affine connection has to be symmetric [78].

Definition 3.9. *An affine connection $\overline{\nabla}$ on a smooth manifold is said to be symmetric if*

$$\overline{\nabla}_{\Phi}\Psi - \overline{\nabla}_{\Psi}\Phi = [\Phi, \Psi] \ , \quad \text{for all } \Phi, \Psi \in \mathfrak{X}(\mathcal{M}) \ .$$

An affine connection being symmetric and compatible with the metric endowed to $\mathcal{M}$ is called the Levi-Civita- (or Riemannian-) connection and is unique [37].

For an illustration of these concepts, observe that in the special case of the Euclidean space $\mathbb{R}^n$ all Christoffel symbols vanish [37], i.e. $\Gamma_{ij}^k = 0$. Hence, the affine connection in terms of its classical expression (3.2), coincides with the usual derivative in the Euclidean space. Compatibility of the metric according to Def. 3.8 yields the well known product rule in $\mathbb{R}^n$.

3.2.2 The Manifold of Euclidean Transformations

After reviewing the general concept of a smooth, differentiable manifold, let us now focus on concrete realizations for the space of Euclidean transformations.

The Lie Group $\mathsf{SE}(3)$ and the Algebra $\mathfrak{se}(3)$

To show that Euclidean transformations naturally form a differentiable manifold, we primarily make use of the fact that $\mathsf{SE}(3)$ forms a group that is defined as follows.

Definition 3.10. *A group $\mathcal{G}$ is a set of elements together with a group operation $\oplus$ that satisfy the fundamental properties:*

1. *Closure:* $A, B \in \mathcal{G} \Rightarrow A \oplus B \in \mathcal{G}$
2. *Associativity:* $\forall A, B, C \in \mathcal{G}\ , (A \oplus B) \oplus C = A \oplus (B \oplus C)$
3. *Identity:* $\exists\, I \in \mathcal{G}$ *such that* $I \oplus A = A \oplus I = A$
4. *Inverse:* $\forall A \in \mathcal{G}\ , \exists\, B = A^{-1} \in \mathcal{G}$ *such that* $A^{-1} \oplus A = A \oplus A^{-1} = I$

Using matrix representation, Euclidean transformations $Y = \{R, t\} \in \mathsf{SE}(3)$ map a point $x \in \mathbb{R}^3$ to $Yx = Rx + t \in \mathbb{R}^3$ and form a group via concatenation $Y_1 Y_2 = \{R_1, t_1\}\{R_2, t_2\} = \{R_1 R_2, t_1 + R_1 t_2\}$. As $R \in \mathsf{SO}(3)$, i.e. $RR^\top = I$ and $\det(R) = 1$, the inverse element of Y is $Y^{-1} = \left\{R^\top, -R^\top t\right\}$.

For the purpose of optimization and numerical analysis, it is convenient to identify $\mathsf{SE}(3) \subset \mathsf{GL}(4)$ with a subgroup of all 4×4 regular matrices with respect to matrix multiplications. Keeping the symbols for simplicity, this representation reads

$$Y = \begin{pmatrix} R & t \\ 0^\top & 1 \end{pmatrix} \ , \qquad Y^{-1} = \begin{pmatrix} R^\top & -R^\top t \\ 0^\top & 1 \end{pmatrix} \ . \tag{3.3}$$

In this way $\mathsf{SE}(3)$ becomes a differentiable manifold embedded into $\mathsf{GL}(4)$, hence a Lie group and enables us directly to apply the theory of Sec. 3.2.1. Associated to each Lie group is the Lie algebra [124], whose underlying vector space completely captures the local structure of the group. In the case of $\mathsf{SE}(3)$ represented as matrix group, the corresponding Lie algebra is given by

$$\mathfrak{se}(3) = \left\{ \begin{pmatrix} \Upsilon_R & \Upsilon_t \\ 0^\top & 0 \end{pmatrix} \middle| \ {\Upsilon_R}^\top = -\Upsilon_R \ , \Upsilon_t \in \mathbb{R}^3 \right\} \ . \tag{3.4}$$

The family of injective mappings required by Def. 3.1, which allows to recover the local group structure from the local parametrization $\mathfrak{se}(3)$ to the curved space $\mathsf{SE}(3)$, is given by the exponential map $\exp : \mathfrak{se}(3) \mapsto \mathsf{SE}(3)$, that reads in the case of matrix representation as

$$\exp(\Upsilon) = \sum_{k=0}^{\infty} \frac{1}{k!} \Upsilon^k = I + \Upsilon + \frac{1}{2}\Upsilon\Upsilon + \frac{1}{6}\Upsilon\Upsilon\Upsilon + \dots \ ,$$

i.e. the usual matrix exponential [56]. Due to the special structure of $\Upsilon \in \mathfrak{se}(3)$, the matrix exponential can be evaluated in closed form.

Theorem 3.1. *[86] Let $\Upsilon \in \mathfrak{se}(3)$ be an element of the Lie algebra associated with $\mathsf{SE}(3)$. Then, the exponential mapping $\exp : \mathfrak{se}(3) \to \mathsf{SE}(3)$ is given by*

$$\exp(\Upsilon) = \begin{pmatrix} \exp(\Upsilon_R) & A\Upsilon_t \\ 0^\top & 1 \end{pmatrix} \ ,$$

where

$$\exp(\Upsilon_R) = I + \frac{\sin(\|\Upsilon_R\|_2)}{\|\Upsilon_R\|_2}\Upsilon_R + \frac{1-\cos(\|\Upsilon_R\|_2)}{\|\Upsilon_R\|_2^2}\Upsilon_R\Upsilon_R \ ,$$
$$A = I + \frac{1-\cos(\|\Upsilon_R\|_2)}{\|\Upsilon_R\|_2^2}\Upsilon_R + \frac{\|\Upsilon_R\|_2 - \sin(\|\Upsilon_R\|_2)}{\|\Upsilon_R\|_2^3}\Upsilon_R\Upsilon_R \ .$$

The closed form expression of the matrix exponential in terms of its rotational part, i.e. $\exp(\Upsilon_R)$, is commonly known as Rodrigues' formula. The idea to prove Thm. 3.1 is primarily based on the characteristic property of skew symmetric matrices such as $\Upsilon_R\Upsilon_R\Upsilon_R = -\|\Upsilon_R\|_2^2\,\Upsilon_R$, as well as on the representation of the trigonometric functions $\sin, \cos$ as infinite series.

Moreover, the exponential mapping, given by Thm. 3.1, defines a surjective mapping from $\mathfrak{se}(3)$ to $\mathsf{SE}(3)$ [86]. Thus, $\mathfrak{se}(3)$ provides a proper parametrization of the manifold of Euclidean transformations.

Tangent Spaces of $\mathsf{SE}(3)$

The geometric link between the Lie group and its related Lie algebra is the fact that the algebra can be viewed as the tangent space to the Lie group at the identity [124]. To illustrate this, we will have a detailed look to the property of the existence of the inverse for each $Y \in \mathsf{SE}(3)$ (cf. Def. 3.10), i.e.

$$YY^{-1} = I \ .$$

Let $Y(t)$ be a smooth curve on the manifold $\mathsf{SE}(3)$ parametrized by t. Computing the derivative of the inverse equation with respect to t, yields

$$\dot{Y}Y^{-1} + Y\dot{Y}^{-1} = 0 \ .$$

Together with the definition of Y, Y^{-1} given by (3.3), this simplifies to

$$\Upsilon = \begin{pmatrix} \Upsilon_R & \Upsilon_t \\ 0^\top & 0 \end{pmatrix} = \dot{Y}Y^{-1} \ ,$$

where Υ_R is a skew- or anti-symmetric matrix, i.e. $\Upsilon_R = -{\Upsilon_R}^\top$. Right hand side multiplication of both sides by Y yields the ordinary differential equation

$$\dot{Y} = \Upsilon Y \ , \tag{3.5}$$

that specifies the general structure of a tangent element of $\mathsf{SE}(3)$ at Y as

$$\mathcal{T}_Y = \left\{ \dot{Y} \,\middle|\, \dot{Y} = \Upsilon Y, \Upsilon_R = -{\Upsilon_R}^\top \right\} \ . \tag{3.6}$$

Hence, in the special case $Y = I$, the tangent space $\mathcal{T}_I$ coincides with the Lie algebra specified in (3.4). Additionally, we can obtain a parametrization of $\mathcal{T}_I$ according to (3.1), using the canonical basis of $\mathfrak{se}(3)$ given by

$$\mathcal{L}_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} , \quad \mathcal{L}_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} , \quad \mathcal{L}_3 = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} ,$$

$$\mathcal{L}_4 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} , \quad \mathcal{L}_5 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} , \quad \mathcal{L}_6 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} ,$$

such that each $\Upsilon \in \mathcal{T}_I$ is given by

$$\Upsilon = \sum_{k=1}^{6} \upsilon_k \mathcal{L}_k \tag{3.7}$$

with $\upsilon_k \in \mathbb{R}, k = 1, \ldots, 6$.

Tangent Vector Fields

Due to the dependency of the tangent space $\mathcal{T}_Y$ on $Y \in \mathsf{SE}(3)$, tangent spaces differ between distinct points. Hence, we have to consider tangent vector fields on $\mathsf{SE}(3)$ as introduced in Def. 3.3, next. From the viewpoint of optimization [2] and flow field computation [69], there are tangent vector fields that are of special interest, such as fields induced by a curve $Y(t)$ on the manifold $\mathsf{SE}(3)$.

The most intuitive way to define a curve in $\mathsf{SE}(3)$ is to use its corresponding Lie algebra. Let $\Theta(t) : \mathbb{R} \mapsto \mathfrak{se}(3)$ be a smooth curve with $\Theta(0) = 0 \in \mathfrak{se}(3)$. Due to the properties of the exponential map, we can define a curve in the local neighborhood of $Y_0 \in \mathsf{SE}(3)$ as

$$Y(t) = \exp\big(\Theta(t)\big) Y_0 \ ,$$

that is differentiable. To compute the tangent vector field induced by $Y(t)$ we have to consider the first order derivative with respect to t.

In the scalar case, where $\theta(t) : \mathbb{R} \mapsto \mathbb{R}$ and the curve is given by $y(t) = \exp\big(\theta(t)\big) y_0 \in \mathbb{R}$, simple differentiation shows that the corresponding tangent reads as $\dot{y}(t) = \dot{\theta}(t) y(t)$. In the case of the matrix valued representation of $\mathsf{SE}(3)$ however, the corresponding results are more involved.

Definition 3.11. *Let* $\Theta(t) : \mathbb{R} \mapsto \mathfrak{se}(3)$ *be smooth and differentiable. The differential of the exponential mapping* dexp *is defined as the tangent of the exponential given by*

$$\frac{d}{dt} \exp\big(\Theta(t)\big) = \mathrm{dexp}_{\Theta(t)}\big(\dot{\Theta}(t)\big) \exp\big(\Theta(t)\big) \ .$$

According to the rules of differentiation, the first order derivative of the curve $Y(t)$ with respect to t is given by

$$\dot{Y}(t) = \mathrm{dexp}_{\Theta(t)}\big(\dot{\Theta}(t)\big) \exp\big(\Theta(t)\big) Y_0 \ ,$$

where the first part denotes the derivative of the exponential map. Due to the possibility of representing exp as an infinite sum, the derivative of the exponential can be expressed analytically [69] as

$$\mathrm{dexp}_{\Psi}(\Phi) = \Phi + \frac{1}{2!}[\Psi, \Phi] + \frac{1}{3!}[\Psi, [\Psi, \Phi]] + \dots \ ,$$

where $[\Psi, \Phi] = \Psi\Phi - \Phi\Psi \in \mathfrak{se}(3)$ denotes to the Lie bracket [124]. Thus, the tangent vector to the curve $Y(t)$ at $t = 0$, is given by the linearization of $\Theta(t)$ at $t = 0$, i.e. $\dot{\Theta}(0)$.

Levi-Civita Connection on $\mathsf{SE}(3)$

Additional to the computation of tangent vector fields, for higher order optimization algorithms, it is essential to analyze the variation of an arbitrary vector field when moving along a curve, i.e. the covariant derivative along tangent vector fields.

Theorem 3.2. *[37] Let* $\mathcal{M}$ *be a smooth manifold with an affine connection* $\overline{\nabla}$*. There exists a unique correspondence which associates to a vector field* $\Phi(t) = \Psi\big(Y(t)\big)$ *with* $\Psi \in \mathfrak{X}(\mathcal{M})$*, another vector field* $\frac{d}{dt}\Phi$ *such that*

$$\frac{d}{dt}\Phi = \overline{\nabla}_{\dot{Y}(t)} \Psi \ .$$

Thus, the derivative of vector fields along curves is directly related to the affine connection on the manifold. Let us now consider the concrete realization [113, 125] of the affine connection on $\mathsf{SE}(3)$.

As we have seen previously in (3.2), the affine connection can be uniquely specified using the Christoffel symbols Γ_{ij}^{k}. Hence, to compute the covariant derivative in terms of Thm. 3.2, we have to specify Γ_{ij}^{k} for connections assigned to the space of Euclidean transformations.

While there are infinitely many affine connections satisfying the requirements of Def. 3.4, we are looking for the connection that additionally preserves the metric and is symmetric, i.e. fulfills the properties of Def. 3.8 and Def. 3.9

Hence, we have to specify a Riemannian metric for the manifold of Euclidean transformation. Due to the fact that $\mathsf{SE}(3) \subset \mathsf{GL}(4)$, we use the standard inner product of the ambient Euclidean space [41]

$$\langle \Phi, \Psi \rangle = \mathrm{tr}\left(\Phi^{\top}\Psi\right) \ , \tag{3.8}$$

which is the canonical inner product for matrices.

Symmetry and preservation of the metric has to be fulfilled for all vector fields $\Phi, \Psi \in \mathfrak{X}(\mathcal{M})$, and consequently for all basis vectors $\mathcal{L}_i$ such that

$$\begin{aligned} \overline{\nabla}_{\mathcal{L}_i}\mathcal{L}_j - \overline{\nabla}_{\mathcal{L}_j}\mathcal{L}_i &= [\mathcal{L}_i, \mathcal{L}_j] \ , \\ \mathcal{L}_k \langle \mathcal{L}_i, \mathcal{L}_j \rangle &= \langle \overline{\nabla}_{\mathcal{L}_k}\mathcal{L}_i, \mathcal{L}_j \rangle + \langle \mathcal{L}_i, \overline{\nabla}_{\mathcal{L}_k}\mathcal{L}_j \rangle \ , \end{aligned}$$

for all $i, j, k = 1, \ldots 6$.

Let $\mathcal{L}_1, \ldots, \mathcal{L}_6$ be the canonical basis of the Lie algebra $\mathfrak{se}(3)$. For the Lie bracket $[\mathcal{L}_i, \mathcal{L}_j]$, we obtain $[\mathcal{L}_i, \mathcal{L}_j] = 0$ for all $i, j = 1, \ldots, 6$ except of

$$\begin{aligned} [\mathcal{L}_1, \mathcal{L}_2] &= \mathcal{L}_3 \ , & [\mathcal{L}_1, \mathcal{L}_5] &= \mathcal{L}_6 \ , & [\mathcal{L}_2, \mathcal{L}_3] &= \mathcal{L}_1 \ , \\ [\mathcal{L}_3, \mathcal{L}_1] &= \mathcal{L}_2 \ , & [\mathcal{L}_3, \mathcal{L}_4] &= \mathcal{L}_5 \ , & [\mathcal{L}_4, \mathcal{L}_2] &= \mathcal{L}_6 \ , \\ [\mathcal{L}_5, \mathcal{L}_3] &= \mathcal{L}_4 \ , & [\mathcal{L}_6, \mathcal{L}_1] &= \mathcal{L}_5 \ , \end{aligned}$$

with corresponding antisymmetric counterparts, i.e. $[\mathcal{L}_i, \mathcal{L}_j] = -[\mathcal{L}_j, \mathcal{L}_i]$. Additionally, as the standard metric induces a constant metric tensor g, with $g_{ij} = \langle \mathcal{L}_i, \mathcal{L}_j \rangle$, derivation of g_{ij} along $\mathcal{L}_k$ vanishes and consequently

$$0 = \langle \overline{\nabla}_{\mathcal{L}_k}\mathcal{L}_i, \mathcal{L}_j \rangle + \langle \mathcal{L}_i, \overline{\nabla}_{\mathcal{L}_k}\mathcal{L}_j \rangle \ .$$

Finally, as the covariant derivative of basis vector fields simplifies to

$$\overline{\nabla}_{\mathcal{L}_i}\mathcal{L}_j = \sum_k \Gamma_{ij}^k \mathcal{L}_k \ ,$$

insertion into the characteristic equations of symmetry and metric preservation yields a system of linear equations determining uniquely the Christoffel symbols that specify the Levi-Civita connection. The non-zero Christoffel symbols yielding a connection that is symmetric and compatible with the metric [22, 125] are given by

$$\begin{aligned} \Gamma_{12}^3 = \Gamma_{23}^1 = \Gamma_{31}^2 &= \frac{1}{2}, \\ \Gamma_{13}^2 = \Gamma_{21}^3 = \Gamma_{32}^1 &= -\frac{1}{2}, \\ \Gamma_{15}^6 = \Gamma_{26}^4 = \Gamma_{34}^5 &= 1, \\ \Gamma_{16}^5 = \Gamma_{24}^6 = \Gamma_{35}^4 &= -1. \end{aligned}$$

This lets us uniquely specify the covariant derivative of vector fields while moving along a curve in the space of Euclidean transformations.

3.2.3 Derivation of Functions on $\mathsf{SE}(3)$

As the primary concern of this chapter is to investigate optimization algorithms for smooth objective functions defined on the space of Euclidean transformations, in the following we analyze first and second order information of functions $f : \mathsf{SE}(3) \mapsto \mathbb{R}$.

Gradient of f

Despite of the general definition of the tangent space according to (3.6), for further simplicity, we henceforth assume that only tangent vectors at $Y = I$ are considered. This assumption imposes no loss of generality for the scenario of Euclidean point set registration, the general application of this work, as the current pose Y can be regarded as an offset redefining the model's original pose.

Taking the usual matrix derivative ∂f of an objective function f, yields no element of $\mathcal{T} = \mathcal{T}_I$ in general. In contrast, the gradient ∇f of f is defined by the relation [82]

$$\langle \partial f, \Psi \rangle = \langle \nabla f, \Psi \rangle \ , \qquad \forall \Psi \in \mathcal{T} \ , \tag{3.9}$$

where $\langle \cdot, \cdot \rangle$ is the Riemannian metric of the underlying manifold and ∂f denotes the usual matrix derivative of f given by

$$(\partial f)_{ij} = \frac{\partial}{\partial Y_{ij}} f(Y) \ .$$

This means that the gradient ∇f is an element of $\mathcal{T}$, i.e. can be represented in form of (3.5), and locally behaves like the matrix derivative in the ambient space. Hence, to obtain ∇f in (3.9), we consider the projection to the tangent space

$$\nabla f = \underset{\Phi \in \mathcal{T}}{\arg\min} \ \frac{1}{2} \left\| \Phi - \partial f \right\|^2 \ , \tag{3.10}$$

where $\| \cdot \|$ is induced by the matrix inner product $\langle \cdot, \cdot \rangle$.

As ∂f can be factorized analogously to the decomposition in (3.3) as

$$\partial f = \begin{pmatrix} \partial f_{11} & \partial f_{12} \\ \partial f_{21} & \partial f_{22} \end{pmatrix}$$

with $\partial f_{11} \in \mathbb{R}^{3\times 3}, \partial f_{12} \in \mathbb{R}^{3\times 1}, \partial f_{21} \in \mathbb{R}^{1\times 3}$ and $\partial f_{22} \in \mathbb{R}$, using the general form of the tangent space (3.5), the minimization problem (3.10) can be solved in closed form as

$$\begin{aligned} \Upsilon_t &= \partial f_{12} \ , \\ \Upsilon_R &= \frac{1}{2} \left(\partial f_{11} - \partial f_{11}^\top \right) \ , \end{aligned}$$

respectively. As a result, the projection of ∂f to $\nabla f \in \mathcal{T}$ reads as

$$\pi_I(\partial f) = \begin{pmatrix} \frac{1}{2} \left(\partial f_{11} - \partial f_{11}^\top \right) & \partial f_{12} \\ 0^\top & 0 \end{pmatrix} \ . \tag{3.11}$$

By inserting (3.11) into (3.9), direct computation shows that (3.9) holds for all $\Psi \in \mathcal{T}$. Additionally, for each $\Phi \in \mathcal{T}$, $\pi_I(\Phi) = \Phi$. This enables to uniquely define the gradient of a function as an element of the tangent space $\mathcal{T}$.

Hessian Map of f

To conclude this section, we discuss the notion of a Hessian map for functions defined on $\mathsf{SE}(3)$. In the Euclidean space, the Hessian matrix of a function $f : \mathbb{R}^n \mapsto \mathbb{R}$ is classically defined as an operator H computing the change of ∇f in direction of $y \in \mathbb{R}^n$ as

$$H(y) = \sum_{ij} \frac{\partial^2}{\partial x_i \partial x_j} f(x)\, y_j\, e_i \ ,$$

where $e_i, i = 1, \ldots, n$ denotes an orthonormal basis of $\mathbb{R}^n$. Hence, the Hessian operator defines a mapping of directions. Analogously, the Hessian operator can be generalized to arbitrary Riemannian manifolds as follows.

Definition 3.12. *Let $f : \mathcal{M} \mapsto \mathbb{R}$ be a real valued function defined on a Riemannian manifold. The Hessian operator H at $Y \in \mathcal{M}$ is a linear mapping of $\mathcal{T}_Y$ onto itself defined by*

$$H(\Phi) = \overline{\nabla}_\Phi \nabla f \ ,$$

for all $\Phi \in \mathcal{T}_Y$ where $\overline{\nabla}$ denotes the Riemannian connection on $\mathcal{M}$ and $\nabla f \in \mathcal{T}_Y$ is the gradient of f.

Thus, it follows directly from Def. 3.12 that the Hessian operator defines a linear mapping that satisfies for all $\Psi \in \mathcal{T}_Y$

$$\langle H(\Phi), \Psi \rangle = \left\langle \overline{\nabla}_\Phi \nabla f, \Psi \right\rangle \ .$$

In the sequel, we want to analyze the Riemannian Hessian on $\mathsf{SE}(3)$ in terms of a function $f\big(Y(t)\big)$, where $Y(t)$ is a curve in $\mathsf{SE}(3)$ with tangent Υ [41]. Then, the Hessian satisfies the relation

$$\langle H(\Upsilon), \Upsilon \rangle = \frac{\partial^2}{\partial t} f\big(Y(t)\big) \ ,$$

that simplifies due to the definition of $\dot{Y}(t) = \Upsilon$ to

$$\partial\partial f(\Upsilon, \Upsilon) + \left\langle \partial f, \dot{\Upsilon} \right\rangle \ ,$$

where ∂f refers to the matrix derivative of f introduced in the previous section and

$$\partial\partial f(\Upsilon, \Upsilon) = \sum_{ij,kl} \frac{\partial^2}{\partial Y_{ij} \partial Y_{kl}} f(Y) \Upsilon_{ij} \Upsilon_{kl} \ ,$$

where $\frac{\partial^2}{\partial Y_{ij} \partial Y_{kl}} f(Y)$ denotes the second order derivative of f in the ambient space $\mathsf{GL}(4)$. Using the representation of Υ in terms of the canonical basis, the time derivative of the tangent Υ in the ambient space is given by $\sum_k \frac{d}{dt}(\upsilon_k)\mathcal{L}_k$.

On the other hand, considering the covariant derivative, given by Thm. 3.2, together with the fact that each geodesic parallel transports its own tangent [37, 41], we obtain

$$\overline{\nabla}_\Upsilon \Upsilon = \sum_k \frac{d}{dt}(\upsilon_k)\mathcal{L}_k + \sum_{ijk} \Gamma_{ij}^k \upsilon_i \upsilon_j \mathcal{L}_k = 0 \ .$$

This yields for the time derivative $\dot{\Upsilon}$ the expression

$$\dot{\Upsilon} = -\Gamma(\Upsilon, \Upsilon) \ ,$$

where $\Gamma(\Upsilon,\Upsilon) = \sum_{ijk} \upsilon_i \upsilon_j \Gamma_{ij}^k \mathcal{L}_k$.

Consequently, insertion into $\langle H(\Upsilon),\Upsilon\rangle$ yields second order information of f given by

$$\langle H(\Upsilon),\Upsilon\rangle = \partial\partial f(\Upsilon,\Upsilon) - \langle \partial f, \Gamma(\Upsilon,\Upsilon)\rangle \ . \tag{3.12}$$

Thus, using (3.11) and (3.12) we can determine up to second order information of the objective functional f, so as to fully exploit the geometry of the underlying space of Euclidean transformations.

3.3 Geometric Integration on $\mathsf{SE}(3)$

In this section, we want to construct generalized Runge-Kutta methods for integration of ordinary differential equations defined on the space of Euclidean transformations in order to determine local optima of an objective functional f. This is motivated by the following.

As we have seen previously, curves on the group of rigid body transformation $\mathsf{SE}(3)$ are naturally defined in terms of an ordinary differential equation

$$\dot{Y}(t) = A\big(t, Y(t)\big) \tag{3.13}$$

where $A(t,Y(t)) = \Upsilon(t)Y(t)$, $\Upsilon(t) \in \mathfrak{se}(3)$, and $Y(t) \in \mathsf{SE}(3)$. Hence, computing $Y(t) \in \mathsf{SE}(3)$ such that its tangent $\dot{Y}(t)$ coincides with $-\nabla f\big(Y(t)\big)$, i.e. the negative gradient of f at $Y(t)$, yields a curve that follows the negative gradient to a local minimizer of f.

However, analytically solving (3.13) cannot be done in general, unless $A(t,Y(t))$ is particularly simple. Thus, we have to adopt numerical integration schemes. Traditional numerical integrators of ordinary differential equations such as Runge-Kutta and multistep methods work in the Euclidean space. Hence, it is a reasonable strategy to apply standard integration schemes in the ambient Euclidean space $\mathsf{GL}(4)$ followed by retracting temporary solutions to the manifold each few iterations. However, such approaches in general cannot guarantee a certain order of approximation accuracy and typically fail to determine the correct flow. This motivates to study different types of numerical integration schemes of ordinary differential equations that are guaranteed to stay on the manifold of Euclidean transformations.

In what follows, we consider Runge-Kutta integration methods on the manifold of Euclidean transformations. For the sake of completeness, we start with a brief review of numerical integration in $\mathbb{R}^n$. For further details we refer the reader to [63, 108].

3.3.1 Runge-Kutta Integration of Differential Equations in $\mathbb{R}^n$

Let us consider the ordinary differential equation defined by (3.13), where $\dot{y}(t) \in \mathbb{R}^n$, and $y(t=0) = y_0 \in \mathbb{R}^n$ is fixed. Given a concrete form of $A\big(t, y(t)\big)$, the tangent to the curve $y(t)$ at $t=0$ is given by $\dot{y}(0) = A\big(0, y_0\big)$.

Euler's Method

The most elementary approach to solve (3.13) subject to $y(t=0) = y_0$ is to let $y(t)$ be a linearization with respect to the tangent at y_0 such that we obtain a local approximation of the curve $y(t)$ around $t=0$ by

$$y(t) \approx y_0 + t\, A\big(0, y_0\big) \ .$$

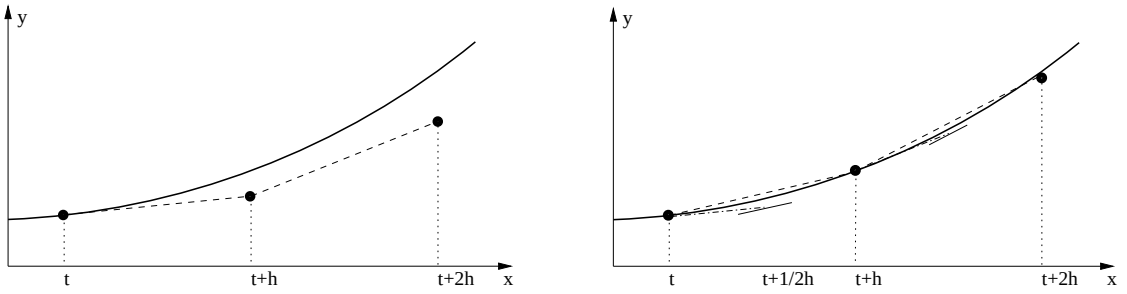


Figure 3.5: Sketch of schemes for obtaining the gradient flow using Euler's (left) and the improved Euler (right) algorithm for a 1D curve. We see that the computed values (circle) of the flow are closer to the true function (solid curve) for the improved Euler Method than for the ordinary algorithm.

Using the finite step-size $h > 0$, we compute a successive approximation of the curve $y(t)$ by

$$y_{t+1} = y_t + h\, A(ht, y_t)\ ,$$

for $t = 0, 1, \ldots$ and y_t denoting the discrete value in iteration t. This simple approach is known as Euler's method [108] and gives accurate results, as long as h is sufficiently small since the error of approximation decreases linearly with h [63]. On the other hand, however, the smaller h, the more steps are required to reconstruct the flow. This causes extremely long run-times in general.

In contrast, for large h, simple forward computations typically fail to approximate the underlying curve $y(t)$ accurately, see Fig. 3.5 for an illustration in the scalar case.

Improving Euler's Method

Let us now consider a simple improvement of Euler's method yielding more accurate results regarding the approximation of the curve $y(t)$. Assuming we want to obtain the value y_{t+1} starting at y_t while using two different step sizes h and $h/2$. The discrete value of y at $t+1$ can be approximated by

$$\begin{aligned} y_{t+1}^{(1)} &= y_t + h\, A(ht, y_t) \\ y_{t+1}^{(2)} &= y_{t+\frac{1}{2}} + \frac{h}{2}\, A\Big(ht + \frac{h}{2}, y_t + \frac{h}{2}\, A(ht, y_t)\Big)\ , \end{aligned}$$

respectively, where the second approximation $y_{t+1}^{(2)}$ results from a double step using step size $h/2$. Using Richardson-Extrapolation [108], i.e. $y_{t+1} \approx 2y_{t+1}^{(2)} - y_{t+1}^{(1)}$, we obtain that

$$y_{t+1} = y_t + h\, A\Big(ht + \frac{h}{2}, y_t + \frac{h}{2}\, A(ht, y_t)\Big)\ .$$

Geometrically, improving Euler's method means that instead of moving along the tangent at y_t to compute y_{t+1}, we only move half the distance to an intermediate point $y_{t+\frac{1}{2}}$ and use the corresponding tangent vector to obtain the value of y_{t+1} at the next iterate. From a computational point of view, we require an additional evaluation of the tangent vector, while typically obtaining far better approximations of the underlying curve (see

Algorithm 3 Improved Euler Algorithm on $\mathbb{R}^n$

Require: $f(\cdot) : \mathbb{R}^n \to \mathbb{R}$
Require: $h > 0$
set $y_0 \in \mathbb{R}^n$
$t \leftarrow 0$
repeat
  $\upsilon_1 \leftarrow \nabla f(y_t)$
  $\upsilon_2 \leftarrow \nabla f\big(y_t + \frac{1}{2}h\upsilon_1\big)$
  $y_{t+h} \leftarrow y_t + h\upsilon_2$
  $t \leftarrow t + h$
until convergence

Fig. 3.5). Moreover, this simple improvement of Euler's method causes the approximation error to decrease quadratically with h [63].

For the problem of optimizing an objective functional $f : \mathbb{R}^n \mapsto \mathbb{R}$, the improved Euler approach can be summarized by Alg. 3.

Runge-Kutta Methods

Although the improvement of Euler's method yields better approximations of $y(t)$ than Euler's method, we study further improvements of the accuracy of numerical integration approaches by considering more sophisticated extensions such as Runge-Kutta methods.

The general idea of Runge-Kutta integration is to decompose the update direction in terms of a set of basic motions $\upsilon_1, \ldots, \upsilon_s \in \mathbb{R}^n$, such that the update of point y_t in iterate t writes as

$$y_{t+1} = y_t + h \sum_{i=1}^{s} b_i \upsilon_i \ ,$$

where $b_1, \ldots, b_s \in \mathbb{R}$ are fixed constants with $\sum_i b_i = 1$. Hence, the update of y_{t+1} depends on the decomposition in terms of $\upsilon_1, \ldots, \upsilon_s$. Motivated by the improvement of Euler's method, we let $\upsilon_1 = A(ht, y_t)$ be the slope of y_t and further basic motions be a linear combination of the previous ones, such that

$$\upsilon_i = A\Big(t + c_i h, y_i + h \sum_{j=1}^{i-1} a_{ij} \upsilon_j\Big) \ ,$$

where c_i refers to the step length and $a_{ij} \geq 0$ specify the coefficients of the combination that are (similar to b_i) characteristic for the corresponding approach and the degree of the approximation error [63]. Typically, the parameters a_{ij}, b_i, c_i are organized in terms of table, i.e. the so-called Butcher tableau:

$$\begin{array}{c|ccccc} c_1 & & & & & \\ c_2 & a_{2,1} & & & & \\ \vdots & \vdots & \ddots & & & \\ c_s & a_{s,1} & a_{s,2} & a_{s,3} & \ldots & a_{s,s} \\ \hline & b_1 & b_2 & b_3 & \ldots & b_s \end{array}$$

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array} \qquad \begin{array}{c|ccc} 0 & & & \\ \frac{1}{3} & \frac{1}{3} & & \\ \frac{2}{3} & 0 & \frac{2}{3} & \\ \hline & \frac{1}{4} & 0 & \frac{3}{4} \end{array} \qquad \begin{array}{c|cc} 0 & & \\ \frac{1}{2} & \frac{1}{2} & \\ \hline & 0 & 1 \end{array} \qquad \begin{array}{c|c} 0 & \\ \hline & 1 \end{array}$$

Table 3.1: Butcher tableaux for Runge-Kutta-MK integration methods of fourth and third (left and second to left) order as well as for the improved Euler method (second to right) and the standard Euler approach (right) .

Choosing the parameters a_{ij}, b_i, c_i for $i, j = 1, \ldots, s$ properly, we obtain a numerical integration approach whose approximation error decreases with h^s [63], where s refers to the order of the integration scheme. Runge-Kutta integration methods, where the corresponding Butcher tableau evolves in triangular form is typically called explicit Runge-Kutta [108], in contrast to implicit approaches where the Butcher tableau might be filled completely. As Euler's method as well as its improved counterpart are special instances of explicit Runge-Kutta schemes, we can characterize these approaches in terms of the corresponding Butcher tableaux as illustrated in Tab. 3.1.

Even though implicit approaches are typically more stable for varying step sizes h, determining the update directions $v_1, \ldots, v_s$ is complicated in general and approximate algorithms have to be applied when extending the concepts to the group of Euclidean transformations [91]. Thus, we do not consider implicit approaches to ordinary differential equations in this work.

3.3.2 Runge-Kutta Crouch-Grossman Methods

In order to extend the principle of Runge-Kutta type integration to the curved space of Euclidean transformations, we make use of the special properties of $\mathsf{SE}(3)$ analyzed in Sec. 3.2.

According to (3.5), the ordinary differential equation (3.13) at $Y(t) \in \mathsf{SE}(3)$, can be represented by

$$\dot{Y}(t) = A\big(t, Y(t)\big) = \Upsilon(t) Y(t) \,,$$

where $\Upsilon(t) \in \mathfrak{se}(3)$. Due to the retraction properties of the exponential map [69] and similar to the Euclidean case, the smooth curve $Y(t)$ can be stated as a linear approximation around $Y(t=0) = Y_0$ by

$$Y(t) \approx \exp\big(t\Upsilon(0)\big) Y_0 \tag{3.14}$$

that is known as the exponentiation of the vector field. Thus, integrating vector fields according to (3.14) extends Euler's algorithm in $\mathbb{R}^n$ to the curved space of Euclidean transformations.

Similar to the Euclidean case, integration techniques based on successive linearization according to (3.14) typically yield poor approximations of the underlying curve, however. Hence, higher order methods have to be studied. Consequently, we have to cope with the decomposition of the update in terms of basis directions that evolve on the curved space of Euclidean transformations.

$$
\begin{array}{c|ccccc}
0 & & & & & \\
\frac{1458}{1783} & \frac{1458}{1783} & & & & \\
\frac{743}{1925} & \frac{1039}{3247} & \frac{97}{1470} & & & \\
\frac{368}{1135} & \frac{997}{1082} & \frac{1167}{2335} & -\frac{475}{433} & & \\
\frac{406}{463} & \frac{173}{487} & \frac{751}{3141} & \frac{547}{393} & -\frac{680}{613} & \\
\hline
 & \frac{407}{2969} & -\frac{135}{7349} & \frac{543}{734} & -\frac{267}{1400} & \frac{696}{2095}
\end{array}
\qquad
\begin{array}{c|ccc}
0 & & & \\
\frac{3}{4} & \frac{3}{24} & & \\
\frac{17}{24} & \frac{119}{216} & \frac{17}{108} & \\
\hline
 & \frac{13}{51} & -\frac{2}{3} & \frac{24}{17}
\end{array}
$$

Table 3.2: Butcher tableaux for numerical Runge-Kutta Crouch-Grossman integration methods of fourth and third order (left, right) respecting the geometry of the space of Euclidean transformations.

A reasonable strategy is to represent the update in terms of direct motions on the manifold $\mathsf{SE}(3)$ that are formed by exponentiating basis motions $\Upsilon_1, \ldots, \Upsilon_s$ in the corresponding algebra. This is roughly the idea of Crouch and Grossman [34]. Using the notations of the Euclidean case, the update of Y_t using a set of update directions evolving on $\mathsf{SE}(3)$ reads as

$$Y_{t+1} = \exp\left(hb_s\Upsilon_s\right)\exp\left(hb_{s-1}\Upsilon_{s-1}\right)\ldots\exp\left(hb_1\Upsilon_1\right)Y_t\ , \tag{3.15}$$

where h denotes the step size and $b_1, \ldots, b_s$ refer to the characteristic coefficients derived from Butcher's tableau. Hence, for a given set of coefficients, we can adapt the explicit Runge-Kutta approach in the Euclidean space to the set of Euclidean transformations as

$$\begin{aligned} Y^{(i)} &= \exp\left(ha_{ii}\Upsilon_i\right)\ldots\exp\left(ha_{i1}\Upsilon_1\right)Y_t\ , \\ \Upsilon_i &= A\big(t + c_ih, Y^{(i)}\big)\ , \end{aligned}$$

for all $i = 1, \ldots s$, followed by the update of Y_t by (3.15). This method is commonly referred to Runge-Kutta Crouch-Grossman integration. The properties of the exponential map guarantee that each Y_{t+1} lies on the curved spaced defined by the group of Euclidean transformations.

Similar to the Euclidean case, the accuracy of this numerical integration approach depends on the coefficients b_i, c_i and a_{ij}. However, due to the fact that the curves of two different flows do not commute in general, i.e.

$$\exp(ha_{ij}\Upsilon_j)\exp(ha_{ik}\Upsilon_k) \neq \exp\left(h(a_{ij}\Upsilon_j + a_{ik}\Upsilon_k)\right)\ ,$$

the coefficients specifying Runge-Kutta approaches in the Euclidean space cannot be adapted to the space of Euclidean transformations directly. In contrast, a fourth-order integration method cannot be implemented in four stages, but requires five stages [92]. For seek of completeness, the corresponding Butcher tableaux for third and fourth order Runge-Kutta Crouch-Grossman integration are presented in Tab. 3.2.

3.3.3 Runge-Kutta Munte-Kaas Methods

In this section, we finally describe a class of numerical integration methods that allow to compute the curve on $\mathsf{SE}(3)$ following the vector field defined by (3.13), while being a direct counterpart to the Euclidean case in terms of using similar characteristic coefficients in

Butcher's tableau for obtaining higher order integration results [92]. The major difference to Runge-Kutta Crouch-Grossman methods is that decomposition of the basic motions will be considered on the Lie algebra rather than on the Lie group such that the basic update equation is given by

$$Y_{t+1} = \exp\left(\sum_{i=1}^{s} b_i \Upsilon_i\right) Y_t \ . \tag{3.16}$$

However, as the ordinary differential equation (3.13) naturally evolves on the Lie group, the crucial step will be pulling the equation back to the Lie algebra.

Theorem 3.3. *[69] For small $t \geq 0$, the solution of $\dot{Y} = \Upsilon(t)Y$ is given by*

$$Y(t) = \exp\big(\Theta(t)\big) Y_0 \ ,$$

where $\Theta \in \mathfrak{se}(3)$ satisfies the differential equation

$$\dot{\Theta}(t) = \operatorname{dexp}^{-1}_{\Theta(t)}\big(\Upsilon(t)\big) \ , \quad \Theta(0) = 0 \tag{3.17}$$

and the dexp^{-1} *refers to the inverse derivative of the matrix exponential.*

This definition yields a differential equation in the Lie algebra. As the Lie algebra is a vector space, one can apply traditional Runge-Kutta integration techniques with its corresponding coefficients (Tab. 3.1), yielding the class of Runge-Kutta Munthe-Kaas approaches [85]. Given the characteristic parameters a_{ij}, c_i, b_i, the differential equation (3.17) can be solved by computing for all $i = 1, \ldots, s$:

$$\begin{aligned}
\Theta_i &= \sum_{j=1}^{i-1} a_{ij} \Upsilon_j \\
\Lambda_i &= hA\big(t + c_i h, \exp(\Theta_i) Y_n\big) \\
\Upsilon_i &= \operatorname{dexp}^{-1}_{\Theta_i}(\Lambda_i)
\end{aligned}$$

followed by an update of $Y(t)$ according to (3.16). The inverse of the derivative of the exponential mapping that is required to compute (3.17) is given by the inverse function of dexp specified by Def. 3.11. In concrete terms, it reads as

$$\operatorname{dexp}^{-1}_{\Phi}(\Psi) = \sum_{j=0}^{\infty} \frac{B_j}{j!} \operatorname{ad}^j_{\Phi}(\Psi) \ ,$$

where $\operatorname{ad}^j_{\Phi}(\Psi) = [\Phi, \operatorname{ad}^{j-1}_{\Phi}(\Psi)]$ and $\operatorname{ad}^0_{\Phi}(\Psi) = \Psi$ and B_j denote the Bernoulli numbers [1]. Although there are closed form representations of dexp^{-1} that allow fast computation [92], for obtaining a numerical integration scheme of approximation order s, it is sufficient to evaluate dexp^{-1} only up to the $(s-1)$-th term [69]. This allows to directly extend the improvement of Euler's method to a numerical integration approach evolving on the group of Euclidean transformations as specified in Alg. 4.

A step-by-step comparison of Alg. 3 and Alg. 4 reveals the modifications necessary to transfer the numerical integration scheme of improving Euler's algorithm to the manifold $\mathsf{SE}(3)$.

Algorithm 4 Improved Euler Algorithm on $\mathsf{SE}(3)$

Require: $f(\cdot) : \mathsf{SE}(3) \to \mathbb{R}$
Require: $h > 0$
set $Y_0 \in \mathsf{SE}(3)$
$t \leftarrow 0$
repeat
  $G \leftarrow \frac{\partial}{\partial Y} f\big(Y_t\big)$
  $\Theta_1 \leftarrow \frac{1}{2} h \pi_{Y_t}(G)$
  $G \leftarrow \frac{\partial}{\partial Y} f\big(\exp(\Theta_1) Y_t\big)$
  $\Theta_2 \leftarrow h \pi_{\exp(\Theta_1) Y_t}(G)$
  $Y_{t+h} \leftarrow \exp(\Theta_2) Y_t$
  $t \leftarrow t + h$
until convergence

Usual gradient computations of the objective functional have to be interleaved with projections onto tangent spaces, because for objective functions defined on manifolds, the gradient actually is a vector field in terms of (3.13), cf. [82]. Moreover, addition is replaced by the group operation in terms of matrix multiplication. This modification reflects that shortest paths along a prescribed direction are straight lines in the Euclidean space, but smooth curves when defined on manifolds.

Compared to first order numerical integration such as Euler's method, higher order integration schemes similar to the Euclidean case yield far more accurate results when reconstructing curves induced by smooth vector fields on manifolds (see e.g. Fig. 3.6). However, better results with respect to more accurate approximations come at the cost of multiple evaluations of the vector field (3.13) per iteration. In terms of function optimization, this means to have multiple gradient evaluations that can be expensive to compute in many applications. Finding the compromise between accurate approximations of higher order Runge-Kutta approaches and costs of multiple gradient evaluations is difficult in general and will not be further addressed in this work.

Additionally, as traditional numerical integration approaches are only concerned with solving ordinary differential equations by means of (3.13), applying such methods to function optimization yield algorithms that only use first order information of the objective. This causes the optimization algorithm to be inefficient in terms of speed of convergence, in general.

Apart from these limitations, algorithms for function optimization based on numerical integration techniques typically have large regions of attraction as long as the step-size is sufficiently small. Hence, they are a good choice in order to numerically evaluate the accuracy of a given objective functional, see Sec. 5.2.1.

3.4 Newton's Algorithm

To obtain a minimizer of a smooth function $f\colon \mathbb{R}^n \to \mathbb{R}$ in a fast and accurate way, Newton's method is typically the method of choice because it converges quadratically provided the initial point $x_0 \in \mathbb{R}^n$ is sufficiently close to the local minimum.

In this section, we briefly recapitulate Newton's algorithm in the Euclidean space and work out two algorithms for geometric optimization that utilize second-order information

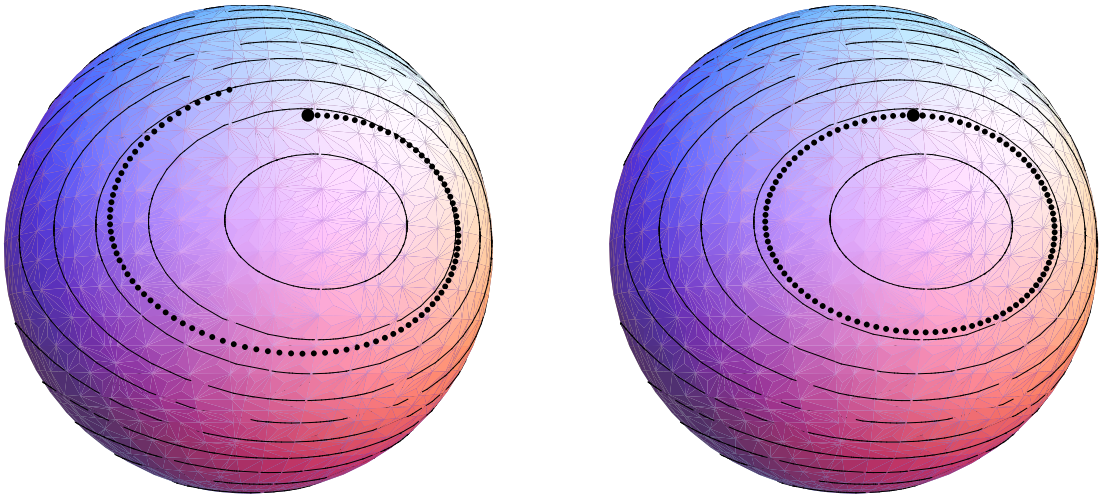


Figure 3.6: Simulation of integration of a toy vector field evolving on a sphere. The solid lines depict prototypical flow curves. While standard forward Euler methods fail to reconstruct the flow accurately, simple variations such as improving Euler's method according to Alg. 4 yield a proper approximation of the flow curve.

of an objective functional defined on the group of rigid body transformations, based on [94] and another variant suggested by ourselves. For the mathematical background, we refer to e.g. [37, 41, 82].

3.4.1 Newton's Method in $\mathbb{R}^n$

Based on a second-order approximation around $x_0 \in \mathbb{R}^n$, i.e. a Taylor series truncated after its second term, an objective functional $f\colon \mathbb{R}^n \to \mathbb{R}$ can be approximated locally by

$$f(x) \approx f(x_0) + (\nabla f_{x_0})^\top (x - x_0) + \frac{1}{2}(x - x_0)^\top H_{x_0}(x - x_0) \ ,$$

where $\nabla f_{x_0}, H_{x_0}$ denote the gradient and the Hessian of f evaluated at x_0, respectively.

A critical point of f is a point $x^* \in \mathbb{R}^n$ such that $\nabla f(x^*) = 0$. Thus, to determine x^* approximately, it is sufficient to consider the gradient of the second order approximation of f with respect to x, leading to the characteristic linear system

$$H_{x_0} y = -\nabla f_{x_0} \tag{3.18}$$

that is to be solved numerically and indicates the next iterate that reads as

$$x = x_0 + y \ . \tag{3.19}$$

Subsequently, applying this second order approximation in x, refines the estimate of the optimal configuration, provided that the initializer x_0 is sufficiently close to the local optimum [35]. In its simplest case, i.e. $f : \mathbb{R} \mapsto \mathbb{R}$, Newton's method can be illustrated by Fig. 3.7.

However, in order to apply this scheme to the minimization of a functional $f : \mathsf{SE}\,(3) \mapsto \mathbb{R}$, we have to take into account that the domain of definition is a curved space.

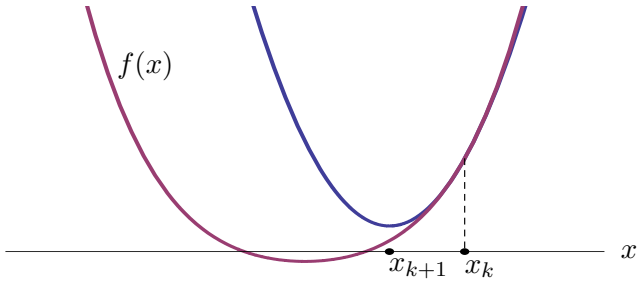


Figure 3.7: Sketch of Newton's method for an objective functional $f : \mathbb{R} \mapsto \mathbb{R}$. Based on an initial value x_k, Newton's algorithm subsequently minimizes a second-order approximation (blue) of $f(x)$ at x_k, by solving the characteristic linear equation (3.18) numerically, yielding an update step to x_{k+1}.

3.4.2 Newton Optimization by Motion Approximation

To motivate the subsequent steps, let us reconsider the exponential mapping $\exp : \mathfrak{se}(3) \mapsto \mathsf{SE}(3)$ in matrix representation that is given by

$$Y = \exp(\Phi) = \sum_{k=0}^{\infty} \frac{\Phi^k}{k!} ,$$

where $Y \in \mathsf{SE}(3)$ and $\Phi \in \mathfrak{se}(3)$. Accordingly, it makes sense to consider the local approximations

$$Y_{lin} \approx I + \Phi \tag{3.20a}$$

$$Y_{quad} \approx I + \Phi + \frac{1}{2}\Phi^2, \tag{3.20b}$$

respectively, as suggested by Pottmann et al. [94], and to determine the optimal tangent vector Φ. By inserting the approximations (3.20a) and (3.20b) into $f(Y)$, and by expanding Φ with respect to the basis $\{\mathcal{L}_k\}_{k=1,\dots,6}$ introduced in (3.7), the objective function $f(Y)$ is restricted to the 6-dimensional vector space $\mathcal{T}$ in terms of the coefficients $(\phi_1, \dots, \phi_6)^\top$ as variables.

As a result, the linear system (3.18) defining the Newton iteration is replaced by (we keep the symbols H and ∇f for simplicity)

$$H(\phi) = -\nabla f, \tag{3.21}$$

where $(\nabla f)_i = \frac{\partial}{\partial \phi_i} f$ and $H_{ij} = \frac{\partial^2}{\partial \phi_i \partial \phi_j} f$ evaluated at $\phi = 0$.

As (3.20a) and (3.20b) are local approximations of the Euclidean group, the solution $\Phi = \sum_k \phi_k \mathcal{L}_k$ of the linear system (3.21) will not be an element of $\mathsf{SE}(3)$ in general such that we cannot apply (3.19) with respect to the group operation. Rather, the Newton update $Y \in \mathsf{SE}(3)$ is determined by inserting Φ into the exponential map, specified by Thm. 3.1. Hence, the entire Newton algorithm is specified by Alg. 5.

3.4.3 Intrinsic Newton Updates

Instead of restricting first the objective function f to the tangent space $\mathcal{T}$ through the local manifold approximations (3.20) and then computing Newton updates by solving (3.21),

Algorithm 5 Newton's Method Based on Motion Approximation

Require: $f : \mathsf{SE}(3) \to \mathbb{R}$, $Y_0 \in \mathsf{SE}(3)$

$k = 0$

repeat

Compute

$$\begin{aligned}(\nabla f)_i &= \frac{\partial}{\partial \phi_i} f\big(Y_{approx} Y_k\big) \ , \\ H_{ij} &= \frac{\partial^2}{\partial \phi_i \partial \phi_j} f\big(Y_{approx} Y_k\big) \ ,\end{aligned}$$

where $Y_{approx} = Y_{lin}, Y_{quad}$, respectively

Obtain ϕ by solving

$$H\phi = -\nabla f \ .$$

Set

$$Y_{k+1} = \exp(\Phi) Y_k \ ,$$

with $\Phi = \sum_{l=1}^{6} \phi_l \mathcal{L}_l$

$k = k + 1$

until convergence

we may base the Newton iteration directly on the intrinsic gradient and Hessian of the manifold $\mathsf{SE}(3)$.

This means that the linear system (3.18) in the Euclidean case is replaced by the linear system defined by the variational equation

$$\langle \overline{\nabla}_\Phi (\nabla f), \Psi \rangle = -\langle \nabla f, \Psi \rangle \ , \quad \forall \Psi \in \mathcal{T} \ , \tag{3.22}$$

with the gradient ∇f given by (3.11) and the Hessian defined in (3.12). While system (3.22) is slightly more expensive to solve than (3.21), it better reflects the geometry of the underlying manifold. We will consider this aspect in more detail in the following subsection and demonstrate favorable properties of (3.22) also in the evaluation part of this work (Sec. 5.2.3).

As in the case of (3.21), the tangent vector Φ solving (3.22) does not directly result in a Euclidean transformation Y as Newton update. Rather to compute the update according to (3.19) with respect to group operations in $\mathsf{SE}(3)$, we have to apply the exponential mapping

$$Y = \exp(\Phi)$$

of Thm. 3.1, too. The complete Newton algorithm using intrinsic gradient and Hessian information is summarized by Alg. 6.

3.4.4 Local vs. Intrinsic Approximation

While both schemes (3.21) and (3.22) require to solve linear systems in each iteration as well as retracting the obtained solution back to the manifold, there are major differences in terms of convergence properties. We address this issue in this section and take it up again in connection with discussing experimental results in Chap. 5, see in particular Sec. 5.2.3.

Algorithm 6 Newton Algorithm Exploiting the Manifold Structure

Require: $f : \mathsf{SE}(3) \to \mathbb{R}$, $Y_0 \in \mathsf{SE}(3)$

$k = 0$

repeat

Obtain $\Phi \in \mathfrak{se}(3)$ such that

$$\langle \overline{\nabla}_\Phi(\nabla f), \Psi \rangle = -\langle \nabla f, \Psi \rangle \ , \quad \forall \Psi \in \mathcal{T} \ ,$$

where

$$(\nabla f) = \pi(\partial f) \ ,$$
$$(\partial f)_{ij} = \frac{\partial}{\partial Y_{ij}} f\big(YY_k\big) \ ,$$

and $\langle \overline{\nabla}_\Phi(\nabla f), \Psi \rangle$ is given by (3.12)

Set

$$Y_{k+1} = \exp(\Phi) Y_k$$

$k = k + 1$

until convergence

Recall from Chap. 2 that the objective function to be studied in this work reads

$$f(Y) = -\sum_{i=1}^{n} \log \left(\frac{1}{m} \sum_{j=1}^{m} \exp\big(-h_{ij}(Y)\big) \right) ,$$

where $h_{ij}(Y) = \frac{1}{\sigma^2} \left\| u_i - Rv_j - t \right\|_2^2$ and $Y \in \mathsf{SE}(3)$.

Approximating the rigid body transformation by truncating the exponential mapping after the linear term (3.20a) yields a redefinition of h_{ij} such that optimization of f is restricted to the tangent space $\mathcal{T}$. As this approach provides an accurate approximation only within a small neighborhood around the current iterate, however, convergence to the correct local optimum is unlikely if it lies outside this neighborhood [94].

In contrast, second order truncation (3.20b) provides a more accurate approximation of the manifold $\mathsf{SE}(3)$ locally. On the other hand, inserting the quadratic approximation into h_{ij} maps $Rv_j + t$ to

$$v_j + \Phi_t + \Phi_R v_j + \frac{1}{2} \Phi_R \big(\Phi_t + \Phi_R v_j \big) .$$

Using the fact that Φ_R is skew symmetric, the latter part rewrites as

$$\frac{1}{2} \big(\Phi_R \Phi_t + (\phi^\top v_j)\phi - (\phi^\top \phi) v_j \big) ,$$

where ϕ are the coefficients of the basis expansion $\Phi_R = \sum_k \phi_k \mathcal{L}_k$.

As a consequence, when the rotation components of Newton updates happen to become large in magnitude, the non-convexity of the objective function due to the quadratic terms involved may cause Newton updates to step into wrong directions. This will be confirmed by numerical experiments in Chap. 5.

This argument can be underlined by considering the Rodrigues' formula, the closed form expression of the exponential map

$$R = I + \Phi_R \frac{\sin(\|\Phi_R\|)}{\|\Phi_R\|} + \Phi_R^2 \frac{1 - \cos(\|\Phi_R\|)}{\|\Phi_R\|^2} \ .$$

Approximating the trigonometric function by its first and second order Taylor expansion in $\|\Phi_R\|$, given by

$$\begin{aligned} \sin(\|\Phi_R\|) &\approx \|\Phi_R\| \ , \qquad & \cos(\|\Phi_R\|) &\approx 1 \ , \\ \sin(\|\Phi_R\|) &\approx \|\Phi_R\| \ , \qquad & \cos(\|\Phi_R\|) &\approx 1 - \frac{1}{2}\|\Phi_R\|^2 \ , \end{aligned}$$

and insertion into Rodrigues' formula directly results in (3.20a) and (3.20b), respectively. Thus, with increasing $\|\Phi_R\|$ the approximation fails to be accurate. Moreover, as this approximation also affects the translation part, large magnitudes in rotation affects the accuracy in t.

Another issue concerns the choice of the metric. While we suggest the canonical metric in the ambient space [41], embeddings of the Euclidean transformations into $\mathbb{R}^6$ and using the corresponding metric, i.e. the standard inner product in $\mathbb{R}^6$, results in a different scaling of the rotational part.

Moreover, representing Φ in terms of its basis expansion, first and second-order approximations yield the restriction of $f : \mathsf{SE}(3) \mapsto \mathbb{R}$ to $f : \mathbb{R}^6 \mapsto \mathbb{R}$, where second-order derivatives are symmetric in the latter Euclidean space, i.e. $\frac{\partial^2}{\partial\phi_i\partial\phi_j} f = \frac{\partial^2}{\partial\phi_j\partial\phi_i} f$. As in general the Lie bracket of two elements $\mathcal{L}_i, \mathcal{L}_j \in \mathfrak{se}(3)$ does not vanish, however, using standard first and second-order derivatives only yields approximations to the correct Hessian. Thus, if the components of the transformation become large in magnitude, the resulting approximation of the Hessian in (3.21) becomes worse, whereas (3.22) is based on (3.12) that includes corrective terms and thus better reflects the geometry of the underlying space. Our numerical evaluation discussed in Sec. 5.2.3 demonstrates that this difference is relevant to applications.

3.5 Summary and Further Work

Summary

To conclude, in this chapter we investigated two conceptually different approaches to determine the optimizer of a smooth objective functional that evolves on the curved space of Euclidean transformations. In contrast to algorithms that cope with the optimization problem in the ambient space and subsequently project temporal solutions to the curved domain of definition, the approaches presented in this chapter fully exploit the geometric properties of the underlying group of transformations specified by Sec. 3.2 and typically yield more stable results.

As the group of Euclidean transformations naturally evolves in terms of an ordinary differential equation, at first, we investigated in Sec. 3.3 optimization techniques based on geometric numerical integration that extend Runge-Kutta type approaches to the manifold of rigid body transformations. By computing a path in $\mathsf{SE}(3)$ induced by the negative gradient vector field of an objective functional, we are able to determine a locally optimal

configuration on the group of Euclidean transformations, provided that the step size is sufficiently small.

Increasing the step sizes typically yield less iterates until convergence at the cost of requiring gradient information at more points on the manifold in order to ensure stable convergence. As evaluation of the gradient is expensive in many applications, finding the optimal trade-off between large step sizes and few gradient evaluations is difficult and is to be addressed in further work.

To cope with the critical determination of step sizes, we secondly investigated in Sec. 3.4 a novel Newton-like algorithm that fully exploits the properties of the underlying manifold while using up to second order information of the objective functional. We analyzed relations to state-of-the-art approaches and commented on benefits and drawbacks of this approach.

Further Work

The proposed Newton-like algorithm typically determines the optimal configuration within few iterations provided the initial configuration is sufficiently close to the optimizer as will be confirmed by numerical experiments in Sec. 5.2.3. However, as the ability of converging to the optimal configuration depends on the objective functional, we want to estimate the size of the region of attraction [35] in further work, in order to obtain a criterion that guarantees convergences to an optimal configuration beforehand.

Additionally, to increase the region of attraction of Newton's algorithm it is common to introduce additional step size regularization techniques [133] to cope with poor approximations of the objective. Another approach is to consider trust region algorithms [2] that slightly complicate the update step while guaranteeing stable convergence. Such approaches have to be addressed in further work as well.

Finally, as for the specific task of point set alignment the objective functions commonly used (see Chap. 2) are highly non-convex, state-of-the-art approaches [70, 94, 110, 111, 135] as well as the optimization algorithms presented in this chapter converge to the optimal solution locally only. Large amounts of noise and clutter as well as natural occlusion further complicate the problem, such that accurate initializations are mandatory. This problem will be addressed in Chap. 4.

Chapter 4

Discrete and Convex Programming for Initialization Estimation

4.1 Overview

4.1.1 Introduction and Motivation

Detecting arbitrary but known objects in a recorded scene by means of obtaining rough estimates of their poses and intrinsic parameters such as deformation is a challenging problem and an important field of research in computer vision that includes a wide range of applications [59, 74, 101, 132].

The most intuitive way to address such problems is to apply so-called template matching strategies, i.e. inspecting the scene for specific structures with respect to fixed configurations such as scale or transformation. This is closely related to the basis pursuit principle [38], where the goal is to reconstruct the scene by few characteristic basis functions, drawn from a large pool of candidates. In the very simple scenario of 1D signal reconstruction this idea can be sketched by Fig. 4.1.

In this chapter, we extend the basic ideas of template matching and sparse reconstruction in order to obtain rough estimates of the number of rigid model instances as well as of the corresponding Euclidean transformations that align the set of objects to 3D measurements, as illustrated in Fig. 4.2 for the prototypical 2D setting. This is primarily based on the reformulation of reconstructing the scene by a collection of object instances in different positions in terms of a binary least-squares objective function including a regularizer enforcing sparsity. This objective quantitatively encodes coverings of the scene with multiple but few model instances at different poses. Due to the large size of the resulting optimization problem, we consider different optimization algorithms [7, 16, 88] to accurately determine the optimal configuration and to meet the time constraints of most industrial applications.

4.1.2 Related Work and Contribution

To find known rigid objects in a recorded scene, i.e. to obtain initial estimates of the objects' pose, a natural approach is to identify parts of the model like cones, tubes, lines, etc., and to infer the objects' pose accordingly [13, 29]. In view of self occlusions, noise, and the ability to uniformly deal with a large variety of objects, basing the approach on the accurate detection of a limited number of specific parts is less attractive, however. Instead, more recent work [8, 31, 48, 53, 71, 109] focused on the extraction of local salient features from scene and model. Feature extraction and correspondence is quite difficult to establish, however, especially if objects exhibit symmetries as commonly occur in industrial settings, and if the samples are noisy and sparsely distributed.

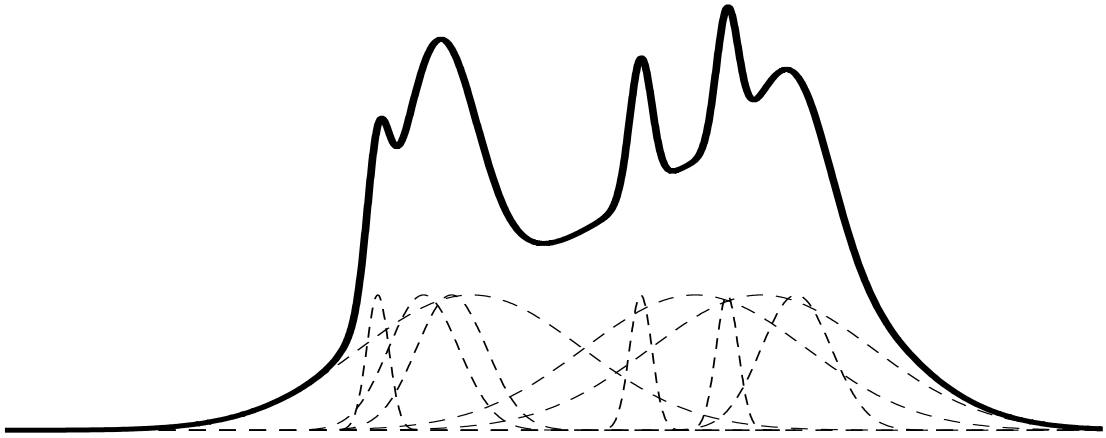


Figure 4.1: Sketch of the sparse signal recovery problem. A given input signal (thick line) is approximated by a linear combination of only few basis functions (dashed lines). The selection of these basis functions is accomplished by solving for a sparse coefficient vector by convex programming. In this chapter, we model the problem of multiple object detection as a sparse signal recovery problem – see Fig. 4.2.

Another established line of research in this context concerns hypothesis generation and verification techniques [5, 46, 130] to obtain rough estimates of the pose [105]. Recent works [58, 129] include accurate data structures to speed up the recognition process at the cost of exhaustive pre-computation.

To this end, we also refer to randomized algorithms such as particle filtering [52, 106, 118] to estimate the pose of arbitrary objects in a recorded scene. However, as these approaches are highly non-deterministic, there are no guarantees to determine the optimal configuration at all.

Moreover, all these approaches are designed to generate hypotheses about *single* object instances matching the scene in general. Consequently, concerning applications with *multiple* object instances, iterative "search and pick" approaches have to be applied, where every incorrect detection affects the entire subsequent process. A prototypical illustration of this issue is given by Fig. 4.3.

In this chapter, we consider a novel approach that *jointly* estimates the pose of *multiple* object instances and resolves conflicting hypotheses through *non-local* contextual processing. Furthermore, we *adaptively* prune the corresponding parameter space based on the given data in order to drastically reduce the otherwise huge problem size in an online manner. Both objectives are accomplished by convex optimization.

Convex models and programming pervade most disciplines and current work on empirical data processing, including reasoning with dictionaries [26], compressed sensing [38], graphical models and inference [126], and machine learning [10]. Discrete and continuous graph cuts [19, 25] and numerous applications provide prominent examples in the field of computer vision. The relevance of globally optimal inference for model evaluation and the guidance of convex modeling for the relaxation of more intricate models can hardly be overestimated. Accordingly, algorithms for efficiently coping with large problem sizes attract more interest in applied research.

In this chapter, we aim at taming the optimization of a highly non-convex objective

Figure 4.2: Extending the principle of sparse signal recovery (see Fig. 4.1) to the problem of 3D template matching – here in 2D for illustration – amounts to approximate the scene (left) by a small subset selected from a large collection of candidates (right panel). Again this can be achieved by convex programming.

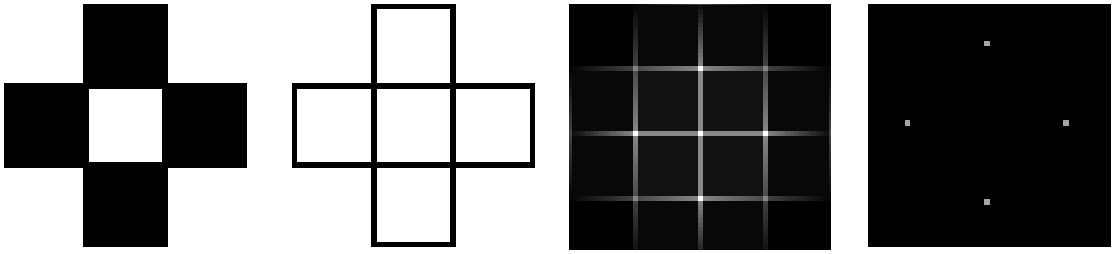
function for the registration of noisy unstructured point sets by detecting in parallel multiple objects together with rough estimates of the underlying Euclidean transformation in a preprocessing step through large-scale convex programming. By inspecting and evaluating the optimality conditions, simple and efficiently computable criteria are obtained that can be applied to any problem instance in order to drastically reduce the problem size in an online fashion. For numerically solving the remaining and still large optimization problem, we competitively evaluate different state-of-the-art approaches to sparse convex programming [7, 16, 88]. Results of this chapter have been partially published in [21].

4.1.3 Organization

The remainder of this chapter is organized as follows. In Sec. 4.2, we mathematically model the problem of estimating the rough positions of multiple object instances for e.g. bin-picking applications as a large-scale binary optimization problem. Specifically, by using a binary least-squares objective function including a regularizer enforcing sparsity that quantitatively encodes coverings of the scene with multiple but few model instances at different poses, we can directly incorporate constraints related to the imaging processes.

In Sec. 4.3, we consider different possibilities to cope with the large size of the resulting problem. By investigating necessary optimality conditions of the optimization problem, we obtain high quality criteria to reduce the underlying search space by fixing variables to their optimal value beforehand. Moreover, we consider different relaxations of the binary large-scale objective in order to apply different convex and continuous programming techniques in Sec. 4.4. To this end, we analyze three different optimization schemes and investigate their applicability to such large-scale problems.

As the optimal configurations obtained by relaxing the objective are not binary in general, we consider in Sec. 4.5 different post-processing techniques to approximately infer Euclidean transformations that are related to the optimal binary solution of the large-scale optimization problem. Finally, we conclude in Sec. 4.6, summarize the results, and point out further work.



(a) (b) (c) (d)

Figure 4.3: A simplified sketch of the scenario addressed in this chapter where our goal is to detect objects (black boxes) based on the given edge images (b). While the result of the Hough-Transform [5] (c) is due to imaging constraints typically inaccurate, i.e. we obtain a vote for a box in the center where no box is placed, the novel method proposed in this chapter (d) accurately detects the four instances of the object. Additionally, due to the joint estimation it does not hallucinate non-existing structures such as the middle object.

4.2 Sparse Reconstruction Problem

Given a set of point measurements $\{u_i,\, i = 1, \ldots, m\} \subset \mathbb{R}^3$ of the scene, and let $\mathcal{O}$ be a model description, i.e. a CAD-file or a sample scan. We wish to estimate the positions of k objects that are assumed to be randomly assembled in the scene in terms of Euclidean transformations $\{Y_j,\, j = 1, \ldots, k\} \subset \mathsf{SE}(3)$. Since k is unknown in general and the space of rigid body transformations is uncountable, this task is involved. To this end, we adopt the basis pursuit approach [26], as illustrated in Fig. 4.1 for the original setting and for our setting in Fig. 4.2.

We act on a few plausible assumptions:

1. All objects in the scene are instances of the same model,
2. imprecise prior knowledge about object poses exists, e.g. objects appear face up, and
3. each object in the scene fits at least a single scene measurement [58].

Based on these assumptions, we can discretize the space of Euclidean transformations to obtain a set of possible object poses $\mathcal{S} = \{Y_j,\, j = 1, \ldots, n\}$, where n is very large.

In spite of these assumptions, the resulting dependencies between scene measurements and possible object poses is quite involved, as sketched in Fig. 4.4. Multiple model configurations are likely to cause the same scene sample, while a scene sample is only allowed to correspond to a single object, due to the imaging process. On the other hand, multiple scene samples typically belong to the same object instance, and the corresponding numbers of samples are unknown.

Next, we derive an objective criterion for fitting multiple object instances with various individual poses to given scene measurements taking into account the dependencies discussed above.

4.2.1 Objective Function

The Euclidean distance of a scene sample u_i to a candidate $\mathcal{O}_{Y_j}$, i.e. an instance of the object of interest in pose Y_j, is denoted by $d(u_i, \mathcal{O}_{Y_j})$. As an object in general is given

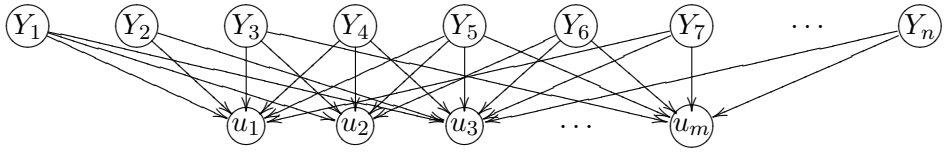


Figure 4.4: Sketch of dependencies between scene samples u_i, $i = 1, \dots m$ and possible object instances with poses Y_j, $j = 1, \dots, n$, where in general $m \ll n$. The graph is highly connected as multiple samples can depend on multiple object configurations.

as a reference scan, i.e. a point cloud, or in case of a CAD file as a set of geometric parts like circles, lines, triangles, and tubes, partitioning the object representation into parts $\mathcal{P}^l$, $l = 1, 2, \dots$, turns the Euclidean distance into

$$d(u_i, \mathcal{O}_{Y_j}) = \min_l d(u_i, \mathcal{P}^l_{Y_j}) \ , \tag{4.1}$$

where $\mathcal{P}^l_{Y_j}$ refers to the l-th object part in configuration Y_j.

Assuming geometric simplicity of the object parts, $d(u_i, \mathcal{P}^l_{Y_j})$ can often be evaluated in closed form. In the case of an object specified by a reference scan, i.e. a set of samples $v_l, l = 1, \dots$, the distance of scene measurement u_i to an object point v_l simplifies to the Euclidean distance

$$d(u_i, \mathcal{P}^l_{Y_j}) := \|u_i - Y_j(v_l)\|_2 \ ,$$

where $Y_j(v_l)$ denotes the Euclidean transformation of v_l by Y_j. Moreover, in this scenario, evaluating the distance function (4.1) can be done efficiently using careful implementations such as pre-computed look-up tables [84] or search trees [104].

Based on the distance measure (4.1), we impose the condition that a scene sample u_i votes for an object instance $\mathcal{O}_{Y_j}$ only if its distance is small within a local neighborhood. Using the indicator variables

$$\eta_{ij} = \begin{cases} 1 \ , & \text{if } d(u_k, \mathcal{O}_{Y_j}) \leq \delta \ , \ \forall u_k \in \mathcal{N}(u_i) \ , \\ 0 \ , & \text{else} \ , \end{cases}$$

where $\delta > 0$ is a user parameter and $\mathcal{N}(u_i)$ denotes a local neighborhood of u_i computed in a preprocessing step, we define the similarity measure $a_{ij} \in [0, 1]$ between u_i and $\mathcal{O}_{Y_j}$ as

$$a_{ij} = \exp\left(-\frac{1}{\sigma} d(u_i, \mathcal{O}_{Y_j})\right) \eta_{ij} \ ,$$

where $\sigma > 0$ controls the sensitivity to noise and $d(u_i, \mathcal{O}_{Y_j})$ is the distance function (4.1). Possible choices for $\mathcal{N}(u_i)$ include the neighborhood with respect to the Euclidean distance, i.e. $u_k \in \mathcal{N}(u_i)$ if $\|u_k - u_i\|_2 < \varrho$, as well as measures based on underlying graph structures [61]. In this work we consider neighborhoods based on the Euclidean distance defined by user parameter ϱ.

Let $x \in \{0, 1\}^n$ collect the indicator variables x_j representing the presence of object instance $\mathcal{O}_{Y_j}$ in the scene. The term $a_{ij} x_i$ then indicates how likely observation u_i belongs to $\mathcal{O}_{Y_j}$. Due to the fact that each object instance fits at least a single scene sample, unique "explanations" for each observation lead to the constraint

$$\sum_j a_{ij} x_j = 1 \ , \quad \forall \ i = 1, \dots, m \ .$$

Because measurements are typically contaminated by noise and a fraction of the scene samples correspond to background structures, however, we sum up the squared residuals of this constraint for each scene sample and obtain the objective function

$$\|Ax - e\|_2^2 \ , \tag{4.2}$$

where $A \in \mathbb{R}^{m \times n}$, $m \ll n$, defines a large underdetermined system and $e = (1, 1, \ldots, 1)^\top$ denotes a vector of ones.

4.2.2 Sparseness Prior

Ruling out conflicting object instances that may have caused the same observation amounts to obtain a minimizer x of (4.2) with minimal support in terms of the (pseudo) ℓ_0-norm

$$\|x\|_0 = \big|\, \{x_j,\, x_j \neq 0\} \,\big| \ ,$$

where $|\cdot|$ denotes the cardinality of a finite set, cf. [39].

Hence, selecting the minimal number of objects, such that the objects fit the scene measurements accurately in terms of (4.2), i.e. $\|Ax - e\|_2^2 \leq \varepsilon$ with $\varepsilon \geq 0$, amounts to solve the optimization problem [39, 120]

$$\min_{x \in \{0,1\}^n} \|x\|_0 \qquad \text{s.t. } \|Ax - e\|_2^2 \leq \varepsilon \ ,$$

which can be restated as

$$\min_{x \in \{0,1\}^n} h(x) \ , \qquad h(x) = \mu \|x\|_0 + \|Ax - e\|_2^2 \ , \tag{4.3}$$

where $\mu > 0$ denotes an appropriate Lagrangian regularization parameter defined by the user.

In the literature, optimization problems such as (4.3) are known as subset selection problems [83, 87, 120], where in the presence of a fixed set of non-zero elements, obtaining the optimal configuration amounts to solve a least squares problem. However, selecting the index set corresponding to the non-zero elements of x is a complex combinatorial problem that is NP-hard [87].

4.2.3 The Compressed Sensing Point of View

Obtaining accurate relaxations to (4.3) such that the solution of the relaxed problem equals the optimal configuration of (4.3) is the primary concern in the field of compressed sensing [38]. In this section, we collect different properties of the underlying system in order to obtain results that justify to consider relaxations for finding an optimal configuration of (4.3).

There is a tremendous amount of theoretical work focusing on the uniqueness and sparsity of relaxed solutions of the ideal noise-free problem, where convex relaxations to

$$\min \|x\|_0 \qquad \text{s.t. } Ax = b$$

are considered, we refer to [93, 119] and the references therein. Although the results yield strict statements on the ability to reconstruct solutions of the original non-convex problem, they are difficult to check and do not apply to real-world settings such as (4.3), in general.

A more realistic setup is to consider a measured signal b of the form $b = Ax + w$, where w is an unknown noise vector and x is supposed to be sparse. While the problem of obtaining x is very ill-posed in general, it turns out that, due to the sparsity, we can accurately approximate x using convex relaxation [39, 120].

Let us assume that $\|w\|_2^2 \leq \varepsilon$. Obtaining a sparse x that reconstructs b up to the presence of noise amounts to solve

$$\min \|x\|_0 \qquad \text{s.t. } \|Ax - b\|_2^2 \leq \varepsilon \,. \tag{4.4}$$

This problem is intractable due to the exponentially large size of the search space and the non-convexity of $\|\cdot\|_0$. To cope with these issues, one might replace the pseudo ℓ_0-norm with its "closest" convex relaxation, i.e. the ℓ_1-norm and obtain the convex problem

$$\min \|y\|_1 \qquad \text{s.t. } \|Ay - b\|_2^2 \leq \delta \,,$$

where $\delta \geq 0$ is related to ε [120]. Using an appropriate Lagrangian multiplier $\gamma \geq 0$, the optimization problem can be restated as

$$\min \gamma \|y\|_1 + \frac{1}{2}\|Ay - b\|_2^2 \tag{4.5}$$

and forms the convex relaxation of the non-convex optimization problem (4.4). The advantage of this formulation is that it can be solved in polynomial time in general [18]. However, it is not clear whether the optimal solution of (4.5) coincides with the optimal configuration of (4.4).

To show the relation between x^* and y^*, the optimal solutions of (4.4) and (4.5), respectively, we introduce a few concepts from the field of sparse approximation. For this purpose we assume in the following that the columns of $A = (a_1, \ldots, a_n)$ are normalized, i.e. ${a_i}^\top a_i = 1$ for all $i = 1, \ldots, n$. Although this assumption does not hold for most real world applications, it enables us to gain insight into the properties of the underlying problem.

Definition 4.1. *Let $x \in \mathbb{R}^n$. The support of x is given by the set of non-zero indices:*

$$\operatorname{supp}(x) = \{i |\, i \in \{1, \ldots, n\}, x_i \neq 0\}$$

Let Λ index a subset of the columns of A that are linearly independent and denoted by A_Λ. The projection [120] of a given signal b to the span of A_Λ is given by $c_\Lambda = A_\Lambda \left(A_\Lambda^\top A_\Lambda\right)^{-1} A_\Lambda^\top b$.

Definition 4.2. *Let Λ index a set of linearly independent columns in A. The exact recovery coefficient* $\operatorname{erc}(\Lambda)$ *is given by*

$$\operatorname{erc}(\Lambda) = 1 - \max_{i \notin \Lambda} \left\| \left(A_\Lambda^\top A_\Lambda\right)^{-1} {A_\Lambda}^\top a_i \right\|_1 \,.$$

The exact recovery coefficient $\operatorname{erc}(\Lambda) \geq 0$ encodes the difference between columns in A_Λ and the remaining ones. It yields a sufficient condition to recover the optimal representation of an exactly sparse signal [119].

Theorem 4.1 ([120]). *Let Λ index a linearly independent collection of columns of A for which $\operatorname{erc}(\Lambda) \geq 0$. Suppose that for given b, the ℓ_2 best approximation over Λ satisfies*

$$\|A^\top (b - c_\Lambda)\|_\infty \leq \gamma \operatorname{erc}(\Lambda) \,,$$

then the optimal y^* solving (4.5) with parameter γ satisfies

1. $\operatorname{supp}(y^*) \subseteq \Lambda$,
2. $\|y^* - x_\Lambda^*\|_\infty \leq \gamma \|\left(A_\Lambda^\top A_\Lambda\right)^{-1}\|_\infty$,
3. $\forall\, i \in \Lambda \,, |(x_\Lambda^*)_i| > \gamma \|\left(A_\Lambda^\top A_\Lambda\right)^{-1}\|_\infty \Rightarrow i \in \operatorname{supp}(y^*)$,
4. y^* *is unique* ,

where x_Λ^* denotes the optimal configuration of (4.4) on Λ.

In other words, the solution of the convex relaxed problem is unique and is not far from the optimal configuration of the non-convex optimization problem (4.4). While Thm. 4.1 yields a criterion that allows to justify the relaxation to a convex problem, checking this hypothesis requires a careful choice of the index set Λ. To cope with this issue, i.e. to derive a criterion that is less influenced by the choice of Λ, we consider the following.

Theorem 4.2 ([120]). *Let Λ index a linearly independent collection of columns of $A \in \mathbb{R}^{m \times n}$ with $|\Lambda| \leq k$. Let $\rho = \max_{i \neq j} |a_i^\top a_j|$, for $i, j = 1, \ldots, n$, be the coherence of A and $\rho k \leq \frac{1}{2}$. Suppose that for given b, the ℓ_2 best approximation over Λ satisfies*

$$\|A^\top (b - c_\Lambda)\|_\infty \leq \gamma \frac{1 - (2k-1)\rho}{1 - (k-1)\rho} \,,$$

then the optimal y^* solving (4.5) with parameter γ satisfies

1. $\operatorname{supp}(y^*) \subseteq \Lambda$,
2. $\|y^* - x_\Lambda^*\|_\infty \leq \frac{\gamma}{1-(k-1)\rho}$,
3. $\forall\, i \in \Lambda \,, |(x_\Lambda^*)_i| > \frac{\gamma}{1-(k-1)\rho} \Rightarrow i \in \operatorname{supp}(y^*)$,
4. y^* *is unique* ,

where x_Λ^* denotes the optimal configuration of (4.4) on Λ.

While this condition is less influenced by the choice of Λ, it remains difficult to obtain the optimal Λ that guarantees to recover a solution of (4.4) when the size of the support of the optimal configuration increases. In case one obtains a Λ that guarantees optimal reconstruction for a large support of x, the columns in the dictionary matrix A are typically highly distinctive such that even greedy methods can solve the problem accurately [39, 119].

In real world settings, however, the columns of A are typically highly overlapping such that the coherence is large and the size of the collection of linear independent columns of A has to be small in general. Hence, the criteria obtained by Thm. 4.1 and Thm. 4.2 only hold for very sparse coefficient vectors [39]. Moreover, as the columns of A are not normalized in settings such as (4.3), adapting these results to our configuration is not straightforward in general, because normalization of the columns of A yields loosing the meaning of the entire setup.

Nonetheless, despite that the results obtained cannot be adapted to real world settings in general as they are very conservative and difficult to check, they highly motivate to consider convex relaxations of the binary optimization problem (4.3) that is to be studied in the following.

4.3 Continuous and Convex Relaxation

To solve optimization problem (4.3), the most naive approach would be to sift through all possible 2^n binary combinations in a brute-force manner. However, with increasing problem size, i.e. increasing n, this approach becomes intractable quickly.

Motivated by the results obtained in Sec. 4.2.3, in the following we consider more sophisticated approaches to determine the solution of (4.3). In detail, we analyze the binary, sparse reconstruction problem and derive efficient preprocessing techniques to fix variables prior to optimization. Moreover, we consider a relaxation to (4.3) that yields a convex objective function defined on a convex domain and consequently can be optimized by standard convex optimization approaches [18].

4.3.1 Problem Analysis

Besides the non-convexity and the discrete underlying domain of (4.3), determination of the optimal configuration is complicated due to the large problem size. In order to ensure feasibility using standard optimization techniques [18], we have to reduce the size of n drastically. Based on necessary optimality criteria, we can fix variables early based on the following.

Theorem 4.3. *Let $x^* \in \{0,1\}^n$ be the global minimizer of the objective function $f(x) = \mu \|x\|_0 + \|Ax - e\|_2^2$. Moreover, let x_k^* denote the k-th element of x^*, $A = (a_1, \ldots, a_n)$, and $a_{ij} \geq 0$, $\forall i = 1, \ldots, m$, $j = 1, \ldots, n$. Then, for all $k \in \{1, \ldots, n\}$, $x_k^* = 0$ if*

$$-\mu + 2(e^\top a_k) - {a_k}^\top a_k < 0 \,. \tag{4.6}$$

Proof. Assume $x_k^* \neq 0$. Due to global optimality

$$f(x) \geq f(x^*) \,,$$

holds true for all $x \in \{0,1\}^n$, in particular for $\widetilde{x}$ given by $\widetilde{x}_j = x_j^*$, $\forall j \neq k$, and $\widetilde{x}_k = 0$. By inserting $\widetilde{x}$ into the global optimality condition and substituting f with its definition, we obtain

$$\mu \|\widetilde{x}\|_0 + \|A\widetilde{x} - e\|_2^2 \geq \mu \|x^*\|_0 + \|Ax^* - e\|_2^2 \,.$$

By construction, x^* and $\widetilde{x}$ are equal except for a single entry. As $x_k^* \in \{0,1\}$ and $x_k^* \neq 0$ by assumption, $\|\widetilde{x}\|_0 = \|x^*\|_0 - x_k^*$. Thus, the optimality condition simplifies to

$$-\mu x_k^* + 2(e^\top a_k)x_k^* - 2x_k^* \left({a_k}^\top A x^*\right) + \left({a_k}^\top a_k\right) x_k^* x_k^* \geq 0 \,.$$

By division of x_k^*, this is equivalent to

$$-\mu + 2(e^\top a_k) - 2{a_k}^\top A x^* + \left({a_k}^\top a_k\right) x_k^* \geq 0 \,.$$

Since $a_{ij} \geq 0$, the left-hand side can be bounded from above by

$$-\mu + 2(e^\top a_k) - \left({a_k}^\top a_k\right) x_k^* \,.$$

Then, due to the assumption $x_k^* \neq 0$ and $x_k^* \in \{0,1\}$, we finally obtain

$$-\mu + 2(e^\top a_k) - {a_k}^\top a_k \geq 0 \,,$$

contradicting (4.6). □

Condition (4.6) allows us to set free variables to zero by simply inspecting single columns of the dictionary matrix A. As we will see in Chap. 5, applying Thm. 4.3 reduces the size of the search space and consequently speeds up the entire optimization procedure dramatically.

Similarly, we can derive a criterion that allows to determine variables that are guaranteed to equal one in the globally optimal solution of (4.3).

Theorem 4.4. *Let $x^* \in \{0,1\}^n$ be the global minimizer of the objective function $f(x) = \mu \|x\|_0 + \|Ax - e\|_2^2$. Moreover, let x_k^* denote the k-th element of x^*, $A = (a_1, \ldots, a_n)$, and $a_{ij} \geq 0$, $\forall i = 1, \ldots, m$, $j = 1, \ldots, n$. Then, for all $k \in \{1, \ldots, n\}$, $x_k^* = 1$ if*

$$\mu - 2(e^\top a_k) + 2(e^\top A^\top a_k) - \left(a_k{}^\top a_k\right) < 0 \tag{4.7}$$

Proof. Assume $x_k^* = 0$. Due to global optimality

$$f(x) \geq f(x^*) \,,$$

holds true for all $x \in \{0,1\}^n$, in particular for $\widetilde{x}$ given by $\widetilde{x}_j = x_j^*$, $\forall j \neq k$, and $\widetilde{x}_k = 1$. By inserting $\widetilde{x}$ into the global optimality condition and substituting f with its definition, we obtain

$$\mu \|\widetilde{x}\|_0 + \|A\widetilde{x} - e\|_2^2 \geq \mu \|x^*\|_0 + \|Ax^* - e\|_2^2 \,.$$

By construction, x^* and $\widetilde{x}$ are equal except for a single entry. As $x_k^* \in \{0,1\}$ and $x_k^* = 0$ by assumption, $\|x^*\|_0 = \|\widetilde{x}\|_0 - \widetilde{x}_k$. Thus, the optimality condition simplifies to

$$\mu\widetilde{x}_k - 2(e^\top a_k)\widetilde{x}_k + 2\widetilde{x}_k \left(a_k{}^\top A\widetilde{x}\right) - \left(a_k{}^\top a_k\right)\widetilde{x}_k\widetilde{x}_k \geq 0 \,.$$

By division of $\widetilde{x}_k$, this is equivalent to

$$\mu - 2(e^\top a_k) + 2a_k{}^\top A\widetilde{x} - \left(a_k{}^\top a_k\right)\widetilde{x}_k \geq 0 \,.$$

Since $a_{ij} \geq 0$ and $\widetilde{x}_k = 1$ by definition, the left-hand side can be bounded from above by

$$\mu - 2(e^\top a_k) + 2(e^\top A^\top a_k) - \left(a_k{}^\top a_k\right) \geq 0 \,,$$

contradicting (4.7). □

As Thm. 4.4 allows to identify elements of the optimal configuration that are guaranteed to equal one, a reasonable strategy to determine the global optimizer of (4.3) would be to apply Thm. 4.3 and Thm. 4.4 iteratively.

However, as $e^\top A^\top a_k = \sum_{i=1}^n a_i{}^\top a_k$ refers to the sum of inner products of a_k with all other columns in the dictionary matrix A, (4.7) only holds if the columns of A are very distinctive. As this does not apply to the real world setting considered in this work, Thm. 4.4 does not reduce the amount of free variables at all.

Although fixing free variables in a preprocessing step simplifies the problem, there is no guarantee that the remaining binary and non-convex optimization problem can be solved efficiently by using brute-force or greedy methods. Hence, we need to consider more sophisticated and efficient techniques for determining a solution of (4.3).

4.3.2 Continuous Relaxation

As the discrete optimization problem (4.3) is computationally intractable in general, we consider the relaxed, continuous optimization problem

$$\min_{x\in[0,1]^n} \mu\|x\|_0 + \|Ax - e\|_2^2 \,. \tag{4.8}$$

Clearly, the optimal binary configuration of the relaxed continuous problem coincides with the solution of (4.3). However, there is no guarantee that the globally optimal solution of (4.8) is binary. To cope with this issue, i.e. to force the results to be binary, one can apply subsequent branch and cut techniques [131], use binarization constraints [4], or consider other post-processing steps. As branch and bound typically requires to solve (4.8) iteratively for different constraint sets, and additional binarization constraints further complicate the optimization, we prefer to apply post-processing steps considered in Sec. 4.5 in order to infer the optimal binary configuration of (4.8).

4.3.3 Convex Relaxation

While relaxing the binary domain to a continuous convex set simplifies the problem, the non-convexity and non-smoothness of the pseudo ℓ_0-norm remains.

A reasonable strategy to cope with this issue is to use an approximation of the pseudo ℓ_0-norm [20] by means of the concave functional

$$e^\top \left(e - e^{-\sigma|x|}\right) \approx \|x\|_0 \,,$$

where $|x|$ refers to the element wise absolute value of x and σ denotes the parameter controlling the smoothness of the approximation, see Fig. 4.5.

Inserting this functional into optimization problem (4.8), the objective function can be approximated by

$$e^\top \left(e - e^{-\sigma|x|}\right) + \|Ax - e\|_2^2 \,.$$

While this objective function forms a differentiable composition of a convex and a concave function on $[0,1]^n$, we can obtain a local optimizer efficiently using the algorithm proposed by An and Tao [4]. To obtain the globally optimal configuration, however, accurate initializers or iterative optimization with varying starting points are mandatory.

To obtain a relaxation that copes with the issues analyzed above, we consider the p-norm

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{\frac{1}{p}}$$

that is convex if and only if $p \geq 1$ [18]. Hence, the convex p-norm closest to the ℓ_0-norm is the ℓ_1-norm (see Fig. 4.5), for which the relaxation of (4.8) is convex and reads as

$$\min_{x\in[0,1]^n} \mu\|x\|_1 + \|Ax - e\|_2^2 \,. \tag{4.9}$$

Due to the results of Sec. 4.2.3, the distance of the optimizers of (4.8) and (4.9) can be expected to be small and the optimal configurations have nearly the same support for

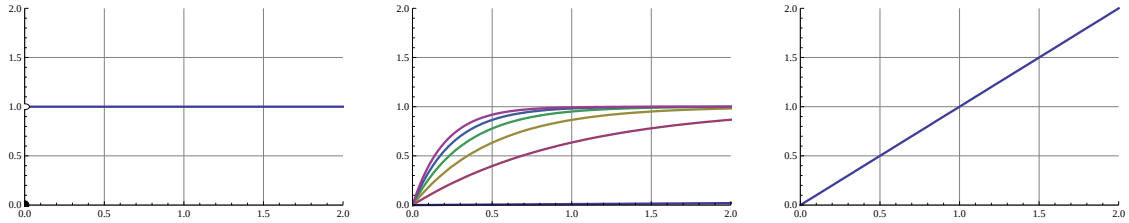


Figure 4.5: Visualization of the different regularizers for optimization. While the pseudo ℓ_0-norm (left) is neither smooth nor convex, using an approximation of the ℓ_0-norm (middle) for different control parameters typically preserves the shape of ℓ_0 while letting the corresponding optimization problem being non-convex. Using the "closest" convex approximation (right) in contrast, i.e. the ℓ_1-norm, dramatically simplifies the underlying optimization problem (4.8) by means of convexifying the objective.

simplified scenarios. Although the requirements to guarantee these results do not hold for our application in general, experiments will show that the optimal configuration of (4.9) yields sufficiently good results.

In the sequel, we study three different optimization procedures for finding the global optimizer of (4.9).

4.4 Convex Optimization

Determining the optimizer of (4.9) amounts to solve a convex quadratic optimization problem subject to box constraints, i.e. $x \in [0,1]^n$.

Problems of such kind are well studied in literature, see for instance [18, 100], and are typically solved using interior-point solvers [18]. However, due to the problem's size of (4.9) such approaches cannot be applied in our case. In the following, we investigate three different approaches to *sparse* convex programming based on Birgin et al.'s spectral project gradients algorithm [16], Nesterov's approach [88], and Beck and Teboulle's fast iterative shrinkage-thresholding algorithm [7].

Even though these algorithms only use first order information of the objective, they optimally fit into the framework of large-scale optimization. Moreover, they yield accurate results with respect to convergence to the optimizer that ensures sufficiently short processing times and thus enables applicability to industrial applications with tight timing constraints.

4.4.1 Spectral Projected Gradients

Instead of optimizing an objective functional itself, it is common to consider local, quadratic Taylor approximations of f in order to obtain an optimal descent direction. In a very similar way, the spectral projected gradients algorithm approximates f around x_k by

$$f(x_k) + (\nabla f(x_k))^\top (x - x_k) + \frac{1}{2}(x - x_k)^\top H(x_k)(x - x_{k-1}) \ ,$$

where H denotes the Hessian of f evaluated at x_k that is approximated by the simplified structure $H = \lambda^{-1} I$ with $\lambda \geq 0$.

In general, the Hessian can be computed using finite differences, such that for all x sufficiently close to x_k, it satisfies the secant equation

$$H(x_k)\big(x - x_k\big) = \nabla f(x) - \nabla f(x_k) \ ,$$

yielding the optimal λ in a least squares sense as

$$\lambda^{-1} = \frac{{s_k}^\top y_k}{{s_k}^\top s_k} \ .$$

where $s_k = x - x_k$ and $y_k = \nabla f(x) - \nabla f(x_k)$.

Thus, for any given value of λ, we can compute an approximation of the local optimizer x_{k+1} by minimizing the simplified second order Taylor series, that yields the update

$$x_{k+1} = x_k - \lambda \nabla f(x_k)$$

known as the Barzilai-Borwein method [97].

However, the approximation of the local optimizer is rather poor, as the Hessian does not fit the simplified form $H = \lambda^{-1} I$ in general. Moreover, there is no guarantee that the approximation of the optimal configuration is element of the domain of definition at all. To cope with these issues, we need to introduce few further modifications.

In order not to be too sensitive to poor approximations of λ, we simply introduce safeguard parameters $\lambda_{\min}, \lambda_{\max} > 0$ that ensure that $\lambda_{\min} \leq \lambda \leq \lambda_{\max}$ by setting

$$\lambda = \min\left\{\lambda_{\max}, \max\left\{\lambda_{\min}, \frac{{s_k}^\top s_k}{{s_k}^\top y_k}\right\}\right\} \ .$$

Additionally, by using a non-monotone line-search strategy [60] that consists of determining the optimal $\alpha \in [0, 1]$ such that

$$f(x_k + \alpha d_k) \leq \max_{0 \leq j \leq m} f(x_{k-j}) + \alpha\gamma\delta \ ,$$

we can determine the optimal step-size even if the update direction yields a poor approximation of the best decent direction, where d_k denotes an update direction, $f(x_{k-j})$ refers to the value of the objective function at iterate x_{k-j}, and the parameters $\gamma, \delta \in \mathbb{R}$ control the degree of non-monotonicity.

The basic idea behind using non-monotone line-search strategies is the following. If the objective function is at most quadratic in certain variables, a single Newton step is typically sufficient to determine the optimal configuration of these unknowns. Nevertheless, it might happen that the value of the objective increases in total. Additionally, since function evaluations, especially in the large-scale setting, are usually computationally expensive, the ability to temporarily increase the objective value typically yields less function evaluations when considering line-search algorithms such as Alg. 8, below.

Finally, to guarantee that each iterate is element of the feasible convex set, i.e. $x_{k+1} \in [0, 1]^n$ in our case, instead of searching in the direction of $\lambda \nabla f(x_k)$ as is the case for classical

Algorithm 7 Spectral Projected Gradients Algorithm [16]

Require: $x_0 \in \mathbb{R}^n$, $m \geq 1$, $0 < \lambda_{\min} < \lambda_{\max}$, $\gamma \in (0,1)$,
$0 < \sigma_1 < \sigma_2 < 1$, $\lambda_0 \in [\lambda_{\min}, \lambda_{\max}]$

```
x_0 = π_Ω(x_0)
k = 0
repeat
  d_k = π_Ω(x_k − λ_k ∇f(x_k)) − x_k
  Compute a_k using Alg. 8
  x_{k+1} = x_k + α_k d_k
  s_k = x_{k+1} − x_k
  y_k = ∇f(x_{k+1}) − ∇f(x_k)
  if ⟨s_k, y_k⟩ ≤ 0 then
    λ_{k+1} = λ_max
  else
    λ_{k+1} = min{λ_max, max{λ_min, ⟨s_k, s_k⟩ / ⟨s_k, y_k⟩}}
  end if
  k = k + 1
until convergence
return x* = x_k
```

Newton-like schemes, we consider the update direction d_k that results from a projection of the local optimizer onto the feasible set Ω, given by

$$d_k = \pi_\Omega(x_k - \lambda \nabla f_{x_k}) - x_k \ ,$$

where $\pi_\Omega : \mathbb{R}^n \mapsto \Omega$ refers to the orthogonal projector.

Combining these extensions with the simplified Newton-like optimization scheme yields the spectral projected gradient algorithm [16] that is summarized in Alg. 7.

Due to the included line-search, the algorithm typically decreases the value of the objective at least each m-th iteration. Additionally, under reasonable assumptions such as f is finite in $\mathbb{R}^n$, Alg. 7 converges to a local minimizer of f on closed convex set $\Omega \subseteq \mathbb{R}^n$ after a finite number of iterations, cf. [16, 127].

In our case, the objective f is also convex such that the optimizer determined by Alg. 7 is the globally optimal configuration of f. Moreover, as Alg. 7 forms a quasi-Newton optimization scheme, it provides fast convergence and due to the exclusive requirements of first order information of f and simple vector calculus, it directly fits into the framework of large-scale optimization.

4.4.2 Nesterov's Algorithm

Although Alg. 7 converges to global optimizer of (4.9) in a finite number of iterations, it is a-priori not clear how many iterations are required to provide a certain accuracy of the optimal configuration. Many industrial applications, however, require such information in order to guarantee fixed run-time constraints. To cope with this issue in the subsequent we consider a different approach to solve (4.9) based on Nesterov's algorithm [88].

This optimization procedure is based on the Lipschitz continuity of the gradient of the objective that is given by

$$\|\nabla f(x) - \nabla f(y)\|_2 \leq L \|x - y\|_2 \ ,$$

Algorithm 8 Line Search required by Alg. 7

$f_{\max} = \max\{f(x_{k-j}), 0 \leq j \leq \min\{k, m-1\}\}$
$x_+ = x_k + d_k$
$\delta = \langle \nabla f(x_k), d_k \rangle$
$\alpha = 1$
while $f(x_+) > f_{\max} + \alpha\gamma\delta$ **do**
  $\alpha_{tmp} = -\frac{1}{2}\frac{\alpha^2\delta}{f(x_+) - f(x_k) - \alpha\delta}$
  if $\sigma_1 \leq \alpha_{tmp} \leq \sigma_2\alpha$ **then**
    $\alpha = \alpha_{tmp}$
  else
    $\alpha = \frac{1}{2}\alpha$
  end if
  $x_+ = x_k + \alpha d_k$
end while
return $\alpha_k = \alpha$

for a fixed $L \geq 0$ and holds for any $x, y \in \Omega$. In the following, we will see that the objective (4.9) satisfies this property where $\Omega = [0,1]^n$.

To show that $f(x) = \mu\|x\|_1 + \|Ax - b\|_2^2$ is Lipschitz continuous in the gradient, we consider

$$\begin{aligned}\|\nabla f(x) - \nabla f(y)\|_2 &= 2\left\|A^\top A(x-y)\right\|_2 \\ &\leq 2\left\|A^\top A\right\|_2 \|x-y\|_2 ,\end{aligned}$$

where setting $L = 2\|A^\top A\|_2$ directly yields the Lipschitz constant of ∇f.

Given an objective function that is Lipschitz continuous in the gradient, we can, for any $x \in \Omega$, derive an upper bound of f at $y \in \Omega$ given by

$$f(y) \leq f(x) - (x-y)^\top \nabla f(x) + \frac{1}{2}L\|x-y\|_2^2 .$$

Hence, instead of minimizing f directly, it is reasonable to consider minimizing the upper bound that yields the optimization problem

$$\arg\min_{y \in \Omega} f(x) - (x-y)^\top \nabla f(x) + \frac{1}{2}L\|x-y\|_2^2 \tag{4.10}$$

and bounds the deviation from the current iterate x. Solving optimization problem (4.10) can be translated to an equivalent optimization problem

$$\arg\min_{y \in \Omega} \frac{1}{2}L\|y - \big(x - \frac{1}{L}\nabla f(x)\big)\|_2^2 ,$$

which can be solved by computing the orthogonal projection of $x - \frac{1}{L}\nabla f(x)$ to the feasible set Ω. As in our case $\Omega = [0,1]^n$, the projection can be implemented efficiently. Optimization procedures that iteratively project the scaled gradient to the feasible set Ω are known as projected gradient algorithms [18].

Algorithm 9 Nesterov's Optimization Procedure for Smooth Functions [88]

Require: $x_0 \in \Omega$, $d(x) : \Omega \mapsto \mathbb{R}$
  for $k \geq 0$ **do**
    $y_k = \arg\min_{y \in \Omega} \langle \nabla f(x_k), y - x_k \rangle + \frac{1}{2} L \|y - x_k\|_2^2$
    $z_k = \arg\min_{z \in \Omega} \frac{1}{\sigma} L d(z) + \sum_{i=0}^{k} \frac{i+1}{2} \langle \nabla f(x_i), z - x_i \rangle$
    $x_{k+1} = \frac{2}{k+3} z_k + \frac{k+1}{k+3} y_k$
  end for
  return $x^* = y_k$

Additionally, to take into account previous iterates in order to better approximate the objective function, we consider the relation

$$f(y_k) \leq \frac{L}{a_k \sigma} d(x) + \frac{1}{a_k} \sum_{i=0}^{k} \alpha_i \big(f(x_i) + \langle \nabla f(x_i), x - x_i \rangle \big) \tag{4.11}$$

where $d(x)$ denotes an appropriate convex prox-function [88] for which $d(x) \geq \frac{1}{2}\sigma \|x - x_0\|^2$ for any $x \in \Omega$, $\alpha_i \geq 0$ refer to user defined parameters and $a_k = \sum_{i=1}^{k} \alpha_i$. This relation roughly reflects the fact that for each sequence of iterates, one wishes to obtain a configuration y_k whose objective value is lower than a convex combination of lower bounds shifted by the prox-function.

Let z_k be the optimal configuration that minimizes the upper bound on $f(y_k)$ in (4.11). Similar to the solution of (4.10), z_k can be computed by projecting the weighted cumulative gradients to Ω. Then, by choosing $\alpha_k = \frac{k+1}{2}$ and $\tau_k = \frac{2}{k+3}$, the relation (4.11) is satisfied if

$$x_{k+1} = \tau_k z_k + (1 - \tau_k) y_k \ ,$$

where y_{k+1} denotes the minimizer of (4.10). Moreover, a sequence of points $\{y_k\}_{k=0}^{\infty}$, $\{x_k\}_{k=0}^{\infty}$ generated in such a manner yields the suboptimality bound

$$f(y) - f(x^*) \leq \frac{4 L d(x^*)}{\sigma (k+1)(k+2)} \ ,$$

where x^* is a global minimizer of f and $\sigma > 0$ refers to the convexity parameter of $d(x)$ [88].

Hence, to guarantee an approximation error of at most $\epsilon \geq 0$, $O(\frac{1}{\sqrt{\epsilon}})$ iterations are required. In contrast, for standard projected gradient algorithms the number of iterations is proportional to $\frac{1}{\epsilon}$. The entire approach for minimizing a smooth objective using Nesterov's procedure can be summarized by Alg. 9.

4.4.3 Fast Iterative Shrinkage-Thresholding Algorithm

Although projections to the feasible set Ω are simple in our setting, we consider a recent approach based on Beck and Teboulle's fast iterative shrinkage-thresholding algorithm [7] that further reduces the computational costs of Nesterov's algorithm while providing similar bounds with respect to the number of iterations to obtain a given accuracy.

Algorithm 10 Fast Iterative Shrinkage-Thresholding Algorithm [7]

Require: $x_0 \in \Omega$
  $y_1 = x_0$
  $t_1 = 1$
  for k = 1, ... **do**
    $x_k = \arg\min_{y \in \Omega} \left(\|y - \frac{1}{L} \nabla f(y_k)\| \right)$
    $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$
    $y_{k+1} = \left(1 - \frac{1-t_k}{t_{k+1}}\right) x_k + \left(\frac{1-t_k}{t_{k+1}}\right) x_{k-1}$
  end for
  return $x^* = x_k$

The major difference of this approach to Nesterov's algorithm is that instead of using an accumulated history of past iterates, it smartly chooses intermediate points that only depend on the last two iterates and uses dynamic step-size modifications.

Very similar to Sec. 4.4.2, we assume ∇f to be Lipschitz continuous with constant L. Moreover, instead of minimizing the objective f directly, we also subsequently consider the minimization of the upper bound, given by (4.10). However, in contrast to considering the additional invariant (4.11), we compute the starting point y_{k+1} of each iterate by

$$y_{k+1} = \left(1 - \frac{1-t_k}{t_{k+1}}\right) x_k + \left(\frac{1-t_k}{t_{k+1}}\right) x_{k-1} \,,$$

where $t_1 = 1$ and $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$ such that the complete approach can be given by Alg. 10.

Applying this procedure yields the approximation error to the globally optimal objective value to be given by

$$f(x_k) - f(x^*) \leq \frac{2L\|x_0 - x^*\|_2^2}{(k+1)^2} \,,$$

which is closely related to Nesterov's bound when choosing the prox-function as $d(x) = \frac{1}{2}\sigma\|x - x_0\|_2^2$. However, for this specific choice of d, we can see that Nesterov's approach requires in theory slightly less iterations at the cost of an additional projection.

4.4.4 Bounds of Lipschitz Constants

The benefit of knowing a-priori the number of iterations required to compute a configuration whose value differs less than a fixed threshold from the global optimum, requires to determine the Lipschitz constant L of the gradient, that for our function (4.9) is given by $L = 2\|A^\top A\|_2$, or an upper bound thereof.

Although there are modifications of Alg. 9 and Alg. 10 that do not require explicit knowledge of L, see [7, 89], they require additional line-search-like steps in each iteration that are typically expensive in the large-scale setting. Thus, in the sequel we consider different possibilities to estimate the Lipschitz constant L for our objective functional (4.9) that are based on approximating the maximal absolute eigenvalue $\lambda_{\max}$ of $A^\top A$.

Algorithm 11 Power Iteration

Require: $A \in R^{n \times n}$, $b_0 \in R^n$
Ensure: $\lambda_k b_k = A b_k$, where λ_k is the dominant eigenvalue of A
  $k = 0$
  repeat
    $b_{k+1} = \frac{A b_k}{\|A b_k\|_2}$
    $\lambda_{k+1} = \|A b_{k+1}\|_2$
    $k = k + 1$
  until convergence

Power Iteration

One basic method to numerically determine the absolute of the maximal eigenvalue of the matrix $A^\top A$ is to apply the power method [56] given by Alg. 11.

However, it is also well known that the performance of this approach highly depends on the absolute of the ratio of the largest to the second largest eigenvalue. Moreover, as power iterations cannot guarantee that intermediate results are always larger than $\|A^\top A\|_2$, we have to apply Alg. 11 until convergence in order to correctly estimate the Lipschitz constant of the gradient of f.

Gerschgorin's Disk Theorem

For eigenvalues of a square matrix $C = (c_{ij})$ there is also the widely used theorem of Gerschgorin that bounds the values in intervals as follows.

Theorem 4.5 (Gerschgorin [56])**.** *Let $C = (c_{ij}) \in \mathbb{R}^{n \times n}$. Then each eigenvalue λ of C lies in one of the disks*

$$D_i = \left\{ \lambda \middle| \|\lambda - c_{ii}\|_2 \leq r_i, r_i = \sum_{\substack{j=1 \\ j \neq i}}^{n} |c_{ij}| \right\}, \qquad i = 1, \ldots, n \; .$$

As in our case $A^\top A$ is symmetric and quadratic, all eigenvalues of $A^\top A$ are non-negative by definition. Moreover, as in our special case $A_{ij} \geq 0$, we can obtain an upper bound on the maximal eigenvalue according to Thm. 4.5 by

$$\max_{i=1,\ldots,n} \sum_{j=1}^{n} {a_j}^\top a_i \; ,$$

where $A = (a_1, \ldots, a_n)$.

Trace Operator

Finally, it is possible to derive an upper bound on the maximal absolute eigenvalue of $A^\top A$ using the trace operator [56] that reads as

$$\mathrm{tr}\left(A^\top A\right) = \sum_{i=1}^{n} \lambda_i$$

and directly yields an upper bound on $\lambda_{\max}$ given by

$$\sum_{i=1}^{n} {a_i}^\top a_i \geq \lambda_{\max} \, .$$

While obtaining this bound requires only cheap computations, the accuracy highly depends on the dominance of the largest eigenvalue. Thus, the approximation will yield poor results if there are multiple λ_i close to $\lambda_{\max}$.

In Chap. 5, we experimentally analyze the accuracy of these bounds, their corresponding computational complexity as well as the influence to optimization algorithms such as Nesterov's approach [88] and the fast iterative shrinkage-thresholding procedure [7].

4.5 Postprocessing

Convex relaxation of the original problem (4.3) allows to efficiently determine the optimal configuration at the cost that the global optimum x^* of (4.9) is not an element of $\{0,1\}^n$ in general, but instead has real-valued components $0 < x_i < 1$.

A common strategy to cope with the binarization of x^* is to iteratively solve the relaxed problem (4.9) subject to additional linear constraints [131] such as $x_i \leq 0, x_i \geq 1$, respectively until a binary solution with minimal objective value is found. While this approach guarantees to obtain the optimal binary configuration, in the worst-case scenario all possible binary configurations have to be visited yielding a runtime proportional to 2^n.

As problem (4.9) is of large problem size, application of such branch-and-cut procedures quickly becomes intractable. In the sequel, we consider different approaches to infer the binary solution of the relaxed problem (4.9) in order to obtain a corresponding high-quality discrete configuration yielding the related objects' positions. However, neither of them guarantee to return the optimal discrete solution to (4.3), of course.

4.5.1 k-Means Clustering

Although the variables $x_1, \ldots, x_n$ are partially interdependent, let us assume pairwise independence for all variables. Then, the results of the convex optimization procedure can be viewed as probabilities, where x_i indicates the probability of the presence of an object with pose Y_i. Moreover, the components of x typically form compact clusters in the model-pose space and are well-localized in the image domain as depicted in Fig. 4.6.

Consequently, a clustering post-processing step such as weighted k-means can provide a high-quality solution indicating the objects' pose. The basic idea of weighted k-means clustering [67] is to determine cluster centers $C_1, \ldots, C_k$ that minimize

$$\sum_{j=1}^{k} \sum_{Y_i \in \mathcal{S}_j} x_i d(Y_i, C_j) \, ,$$

where $\mathcal{S}_j \subset \mathsf{SE}\,(3)$ denotes the discrete set of Euclidean transformations that are assigned to cluster center C_j and $d(Y_i, C_j)$ refers to the distance of Y_i to C_j. Similar to the hard-assignment problems introduced in Sec. 2.2.1, obtaining $C_1, \ldots, C_k$ requires proper initial estimates on $\mathcal{S}_j$.

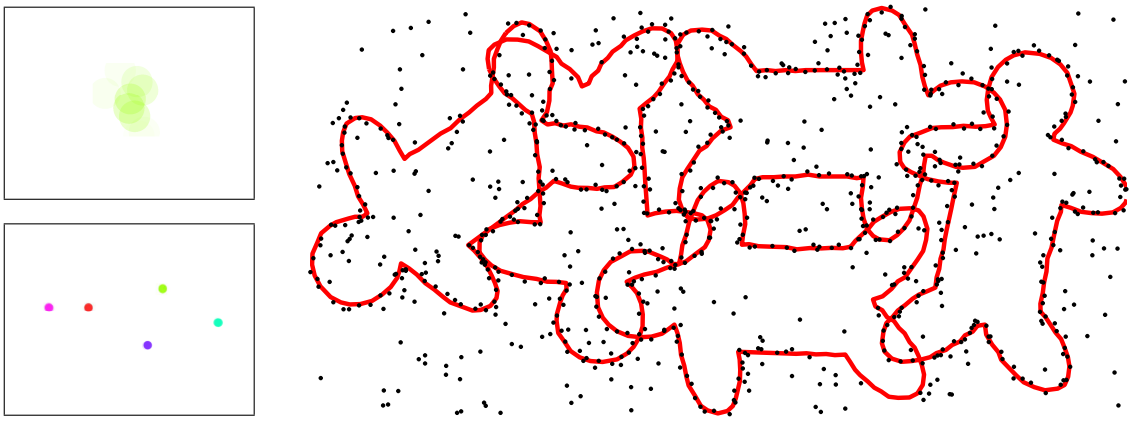


Figure 4.6: Despite a substantial amount of noise and background clutter, poses indicated by x after convex optimization are compactly located (left). A close-up view is shown top left. Dots indicate the center of the model and colors their orientation. The correct object instances in the scene (red shapes) can be determined quickly by a clustering postprocessing step.

Due to the clusters' compactness, however, simple clustering, i.e. assigning Y_i, Y_j to the same cluster $\mathcal{S}_l$ if $d(Y_i, Y_j) < \epsilon$ for an user parameter $\epsilon > 0$, turned out to work very well for computing a reasonable initialization of $\mathcal{S}_1, \dots \mathcal{S}_k$ as well as on the number of cluster centers k, where we used the canonical distance measure [115] on the Lie algebra $\mathfrak{se}(3)$ given by

$$d(Y_i, Y_j) = \left\| \log\left(Y_i^{-1} Y_j\right) \right\| \ , \tag{4.12}$$

where $\log : \mathsf{SE}(3) \mapsto \mathfrak{se}(3)$ denotes the ordinary matrix logarithm and $\|\cdot\|$ refers to the matrix Frobenius norm [56] given by $\|A\| = \sqrt{\mathrm{tr}(A^\top A)}$ and previously used in Chap. 3.

Then, obtaining the cluster center C_j amounts to minimize the smooth objective functional

$$\sum_{Y_i \in \mathcal{S}_j} x_i \left\| \log\left(Y_i^{-1} C_j\right) \right\|$$

for all $j = 1, \dots, k$, separately. This can be done using geometric optimization techniques such as gradient descent algorithms or Newton-like methods, as specified in Chap. 3.

Even though C_j is not an element of the original discretized space of candidate transformations in general, it provides accurate initializations to the pose estimation problem. Additionally setting $x_i = 1$, if the pose Y_i is closest to C_j for all $j = 1, \dots, k$, typically yields proper binary solutions to the original discrete optimization problem (4.3). However, the accuracy of the corresponding result highly depends on the initialization of the cluster centers $C_1, \dots, C_k$ as well as on the number of clusters k and consequently on the compactness of the clusters.

4.5.2 The Mean Shift Procedure

An attractive feature of mean shift clustering [28, 32, 49] is that it does not require a-priori knowledge of the number of clusters. In contrast, it estimates an underlying probability density function using discrete data points and iteratively seeks for a nearby stationary point and thus detects the modes of the density [28].

In what follows, we briefly review the mean shift approach in the Euclidean vector space $\mathbb{R}^n$ and outline its extensions to the space of Euclidean transformations $\mathsf{SE}(3)$.

Mean Shift in $\mathbb{R}^n$

The original mean shift algorithm was first introduced in Fukunaga and Hostetler [49] and later rediscovered by Cheng [28] and Comaniciu and Meer [32]. Given a set of n data points $x_1, \ldots, x_n$, the kernel density estimator in point x is then given by

$$f(x) = \frac{1}{n} \sum_{i=1}^{n} K_\sigma(\|x - x_i\|_2^2) \,,$$

with K_σ denoting a multivariate symmetric kernel, such as the Gaussian kernel introduced in Chap. 2, and σ controls its size. To detect the modes of $f(x)$, it is sufficient to find the zeros of the gradient, i.e. $\nabla f(x) = 0$.

Using the Gaussian kernel given by $K_\sigma(x) = \frac{1}{z_\sigma} \exp\left(-\frac{1}{2\sigma^2} x\right)$ with normalization constant z_σ, the gradient of the kernel density estimator writes as

$$\begin{aligned} \nabla f(x) &= \frac{1}{n\sigma^2} \sum_{i=1}^{n} x_i K_\sigma(\|x - x_i\|_2^2) - \frac{1}{n\sigma^2} x \sum_{i=1}^{n} K_\sigma(\|x - x_i\|_2^2) \\ &= \frac{1}{n\sigma^2} \left(\sum_{i=1}^{n} K_\sigma(\|x - x_i\|_2^2) \right) \left(\frac{\sum_{i=1}^{n} x_i K_\sigma(\|x - x_i\|_2^2)}{\sum_{i=1}^{n} K_\sigma(\|x - x_i\|_2^2)} - x \right) \,, \end{aligned}$$

where the first term is proportional to the kernel density estimate and the second term denotes the so-called mean shift vector. As $f(x) > 0$ by definition, the gradient vanishes for all data points $x_1, \ldots, x_n$ if and only if the corresponding mean shift vector equals 0. This directly yields the simultaneous update

$$x_j^{(k+1)} = \frac{\sum_{i=1}^{n} x_i K_\sigma(\|x_j^{(k)} - x_i^{(k)}\|_2^2)}{\sum_{i=1}^{n} K_\sigma(\|x_j^{(k)} - x_i^{(k)}\|_2^2)} \tag{4.13}$$

for all $j = 1, \ldots, n$ where $x_j^{(k)}$ denotes data sample x_j in the k-th iterate. Moreover, reordering the terms of the gradient yields that the mean shift vector is given by

$$m(x) = \sigma^2 \frac{\nabla f(x)}{f(x)} \,.$$

Hence, the mean shift vector always points towards the direction of the maximum increase of the density estimate such that stepping into direction m for all $x_i, i = 1, \ldots, n$ according to (4.13) yields each sample to converge to stationary points as depicted in Fig. 4.7 for a two dimensional prototypical example. Moreover, normalization of the gradient direction by $f(x)$ causes the algorithm to make large steps through low-density regions while lowering the step-size near local optima. Thus, the mean shift procedure can be interpreted as an adaptive gradient ascent method [32] that is guaranteed to converge to a local mode [28].

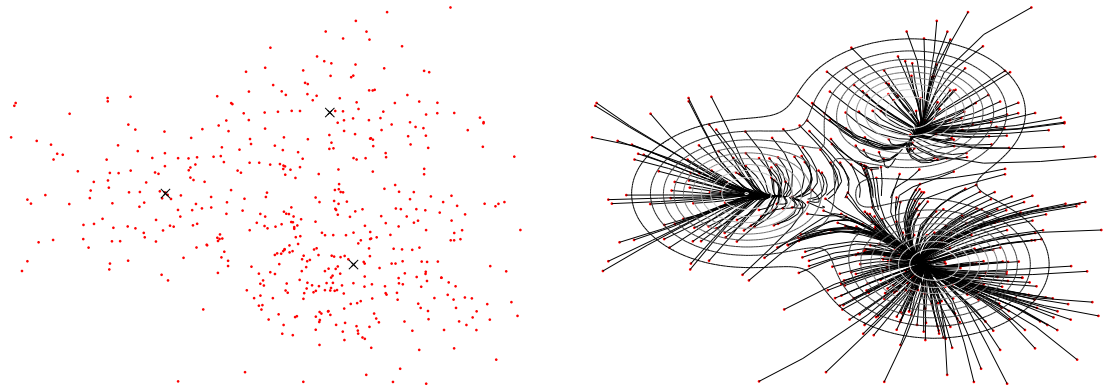


Figure 4.7: Sketch of the mean shift approach in the Euclidean space $\mathbb{R}^2$. Based on samples (red dots) of an underlying probability density function where the modes are indicated by black crosses (left) the mean shift approach let the samples converge along the trajectories to the local stationary point (right), given a proper window parameter h.

Mean Shift in $\mathsf{SE}\,(3)$

To generalize this principle to elements $X_1, \ldots, X_n \in \mathsf{SE}\,(3)$ we have to take the structure of the space of Euclidean transformations into account [115], as the weighted sum of Euclidean transformations is not an element of $\mathsf{SE}\,(3)$ in general.

However, by resorting to tangent vector spaces, the mean-shift operations above can be properly defined on the manifold $\mathsf{SE}\,(3)$, see Sec. 3.2. For further simplicity, we consider the tangent space at $I \in \mathsf{SE}\,(3)$ exclusively and retract the computations around $X \in \mathsf{SE}\,(3)$ to I. Then, the distance between $X, Y \in \mathsf{SE}\,(3)$ can be measured by (4.12). Additionally, the derivative of the squared distance function [115] at $X = I$ with respect to X is given by

$$\nabla d^2(I, X_i) = -2\log(X_i)\ .$$

This lets us compute the derivative of the kernel density estimate given by $f(X) = \frac{1}{n}\sum_{i=1}^{n} K_\sigma\big(d^2(X, X_i)\big)$ with K_σ being a Gaussian kernel as

$$\nabla f(X) = \frac{1}{n\sigma^2}\sum_{i=1}^{n} K_\sigma\big(d^2(X, X_i)\big)\log\left(X^{-1}X_i\right)\ .$$

Analogous to the Euclidean case, this allows to let us define the simultaneous nonlinear mean shift update [115] as

$$X_i^{(k+1)} = \exp\left(m(X_i^{(k)})\right)X_i^{(k)}\ .$$

where $X_i^{(k)}$ denotes the i-th transformation in the k-th iterate and the mean shift vector is given by

$$m(X) = \frac{\sum_{i=1}^{n} K_\sigma\big(d^2(X, Y_i)\big)\log\left(X^{-1}Y_i\right)}{\sum_{i=1}^{n} K_\sigma\big(d^2(X, Y_i)\big)}$$

being an element of $\mathfrak{se}\,(3)$.

Algorithm 12 Generic Randomized Rounding

Require: Optimization problem $\min_{x\in\{0,1\}^n} f(x)$
Ensure: $\bar{x}^* \in \{0,1\}^n$ is feasible optimizer of f with high probability
Solve $x^* = \arg\min_{x\in[0,1]^n} f(x)$
Round the fractional parts of x^* as

$$\begin{aligned} \bar{x}_i^* &= 1 \quad \text{with probability } p(x_i^*) \\ \bar{x}_i^* &= 0 \quad \text{with probability } 1 - p(x_i^*) \end{aligned}$$

where the rounding function p depends on the f.

Similar to the Euclidean case, applying the mean shift approach in SE(3) follows the gradient to stationary points of the underlying density function while respecting the geometric property of the space of Euclidean transformations. However, as is the case with k-means clustering, when using mean shift for postprocessing the results obtained are typically not an element of the original discretized space.

4.5.3 Randomized Rounding

Finally, we study a procedure to infer the optimal binary configuration of (4.9) based on heuristic rounding techniques [12]. Instead of clustering poses to obtain intermediate results $C_1, \ldots, C_k \notin \{Y_1, \ldots, Y_n\}$, randomly rounding the fractional parts of the globally optimal configuration of (4.9) yields a subset of the originally discretized Euclidean pose space.

Let $\bar{x}^*$ be the unknown optimal binary configuration and x^* be the optimal configuration of (4.9). The generic method to obtain non-fractional solutions $\bar{x}^*$ based on x^*, is to randomly round the entries to $\{0,1\}$ according to a given rounding function $p(x)$ as specified by Alg. 12 for the simple case of pairwise independent samples.

Finding an appropriate rounding function $p(x)$ however is involved in general, especially if there are interdependencies between variables. To cope with this issue we can derandomize Alg. 12, i.e. resolve dependencies by fixing variables and lower the dimensionality of the problem in order to apply deterministic algorithms or to simplify the choice of the rounding function.

Derandomization consists of selecting values for $x_1, \ldots, x_n \in \{0,1\}$ sequentially as follows:

1. set $x_1 = 0$ if
$$E\left[f(x)|x_1 = 0\right] \leq E\left[f(x)|x_1 = 1\right]$$
and 1 otherwise
2. for $j \geq 2$, set $x_j = 0$ if
$$E\left[f(x)|x_1, \ldots, x_{j-1}, x_j = 0\right] \leq E\left[f(x)|x_1, \ldots, x_{j-1}, x_j = 1\right]$$
and 1 otherwise,

where we compute the expected value by subsequently generating samples $\bar{y}$ from Alg. 12 using $p(x_i) = x_i$ where x_i corresponds to the fractional solution of the relaxed optimization problem and all $x_1, \ldots, x_{j-1}$ are fixed according to the conditions of prior iterates.

As we are subsequently sampling according to a distribution induced by the optimal continuous result and the expected value of the potential function decreases with each iteration, the objective value of the binary configuration obtained by derandomization does not differ much from the global binary optimum of (4.9), cf. [12].

Applying randomized rounding and the corresponding derandomization procedures however, requires to solve the convex relaxation (4.9) subsequently, i.e. one time for each variable that has to be fixed. Due to the large problem size, however, solving (4.9) is typically numerically inefficient, such that clustering procedures are consequently the better compromise between accuracy of initialization and computational speed of the overall procedure.

4.6 Summary and Further Work

Summary

In this chapter, we investigated a novel approach for obtaining rough estimates of the number of rigid model instances as well as of the corresponding Euclidean transformations that align the set of objects to 3D measurements, by extending the basic ideas of template matching and sparse reconstruction. The results obtained by this approach typically provide accurate initializations for local refinement algorithms such as gradient descent or Newton-like procedures presented in Chap. 3.

The presented procedure jointly estimates the pose of multiple object instances and resolves conflicting hypotheses through non-local contextual processing. The entire approach is primarily based on the reformulation of reconstructing the scene by a collection of object instances in different positions in terms of a binary least-squares objective function including a regularizer enforcing sparsity that is presented in Sec. 4.2. This objective quantitatively encodes coverings of the scene with multiple but few model instances at different poses.

Due to the large size of the resulting optimization problem, we considered different ways to determine the optimizer of the binary problem by relaxing the objective in order to apply convex programming techniques, see Sec. 4.3. Moreover, a detailed analysis of the binary objective revealed that it is possible to fix variables prior to optimization by means of inspecting sufficient optimality conditions and thus to reduce the space of free variables.

For numerically solving the remaining and still large optimization problem, in Sec. 4.4 we competitively evaluated different state-of-the-art approaches to sparse convex programming. As the optimal configuration of the convex problem is fractional in general, we additionally considered in Sec. 4.5 different postprocessing strategies to infer the optimal binary configuration ranging from simple k-means clustering to advanced mean shift approaches exploiting the geometry of the underlying space.

An experimental evaluation of the entire initialization approach for the specific task of bin-picking is given in Chap. 5.

Further Work

Although the presented postprocessing algorithms perform well in our experiments, they provide no guarantees that the obtained solution coincides with the globally optimal config-

uration of the original binary least squares problem. In further work, we want to investigate new strategies that yield tighter bounds.

Moreover, we hope to derive tighter optimality conditions in order to further reduce the space of free variables in a preprocessing step as well as to allow online pruning techniques that fix variables to their final configuration within convex optimization.

Finally, improving the theoretical basis of the relaxations of the non-convex, binary and sparse reconstruction problem to convex formulations for realistic setups such as the bin-picking scenario considered in this work, where there are non-normalized columns, noisy measurements, as well as constraints on the search space, has to be addressed in further work.

Chapter 5
Validation and Applications

Accuracy, robustness, and speed are of primary importance for industrial applications. In this chapter, we therefore experimentally evaluate the proposed approaches for rough initialization estimation presented in Chap. 4 as well as for the geometric refinement processes investigated in Chap. 3 in combination with the objective function of Chap. 2. We apply these algorithms to the special scenario of industrial bin-picking applications.

To this end, we use both computer-generated data allowing for full control of the evaluation and real industrial data as shown in Fig. 1.1. Artificial measurements are generated by reference scans of a simulated scanning device (see Fig. 1.2) or by simply using samples obtained from a given CAD file as depicted in Fig. 5.1. We also consider noise-free computer generated 3D models [123].

Moreover, we compare the algorithms presented in the previous chapters to a range of reimplemented state-of-the-art algorithms including the Iterative Closest Point (ICP) approach [14, 104] based on efficient k-D-tree implementations [11], the fix-point iteration as a special case of Soft Assign [96], an expectation-maximization variant of ICP [57], and the correspondence independent Kernel Correlation [70, 121] algorithm as detailed in Chap. 2. Additionally, we numerically evaluate Newton-like procedures based on local approximations of the smooth manifold of Euclidean transformations [84, 94], cf. Chap. 3.

5.1 Coarse Registration

Finding proper initializations amounts to solve a convex optimization problem as discussed in Sec. 4.3. In the following subsections, we separately discuss the two major issues involved in this connection: the large problem size of the convex relaxation and the conversion of the global optimum to a binary solution in a postprocessing step.

5.1.1 Efficient Initialization

As short processing times are important for many industrial applications, we briefly point out properties of our approach enabling fast on-the-fly computations of some steps of the overall approach.

Concerning the preprocessing based on Thm. 4.3, only the object in position $\mathcal{O}_{Y_k}$ is required to compute the corresponding column vector a_k and to determine if the related indicator variable x_k can be set to zero (i.e. ignored) immediately. Furthermore, each entry of a_k can be computed in parallel and, due to the exclusive requirement of shortest distance evaluations, precomputed distance maps [84] reduce the computation to simple look-up table access.

As a consequence, the remaining costly part is the computation of the local neighborhood for which a range of established algorithms and data structures such as k-D-trees [11] are available.

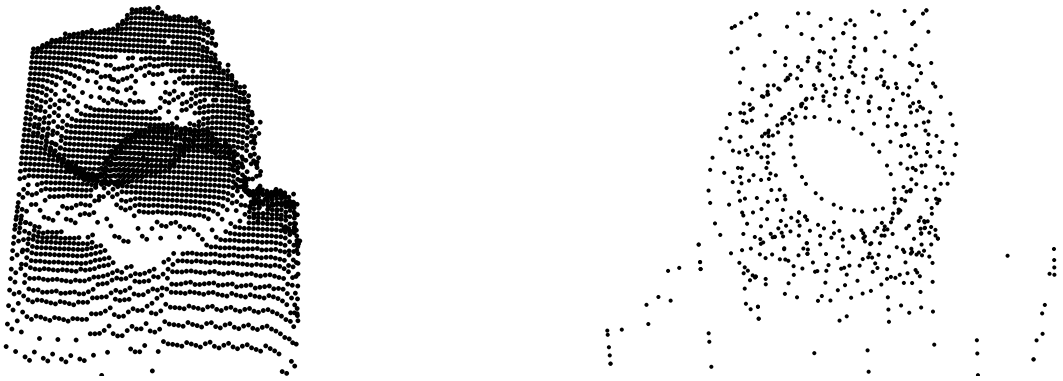


Figure 5.1: Prototypical examples of computer generated 3D point measurements obtained by simulating the scanning device (left) depicted in Fig. 1.2 and by discretizing a given CAD model (right), respectively.

So, realizing the initialization procedure presented in Chap. 4 can be done efficiently while using parallel architectures enables to meet the run-time constraints of many industrial applications.

5.1.2 Convex Optimization

In this section, we numerically evaluate different state-of-the-art approaches previously presented in Sec. 4.4 to solve the convex initialization problem, i.e. we analyze the performance of the spectral projected gradient (SPG) method [16], Nesterov's algorithm [88] and the fast iterative shrinkage-thresholding (FISTA) approach [7]. All algorithms only require evaluations of the objective function and its gradient. Hence, they are suited for large-scale sparse convex programming.

In order to competitively evaluate the performance of SPG, FISTA, and Nesterov's approach, we consider the 2D setup depicted in Fig. 4.2 as well as the real world 3D data set presented in the upper panel of Fig. 1.1. For the prototypical 2D example we used a total of 1 335 840 candidate object transformations. Then, application of Thm. 4.3 fixes $\approx 99.7\%$(!) of the variables beforehand. The remaining 3497 variables were determined using SPG, Nesterov's algorithm, and FISTA, respectively.

Concerning FISTA and Nesterov's approach, we used the three methods to numerically determine or estimate the Lipschitz constant $L = \|A^\top A\|_2$ of the gradient of f as presented in Sec. 4.4: the power iteration [56] to infer the exact L, application of Gerschgorin's disk theorem to obtain an upper bound, and the trace operator of $A^\top A$ returning the sum of all eigenvalues as an upper bound.

While in this example the power iteration converged within a few iterations, it required multiple matrix-vector multiplications and therefore took about 0.75 seconds. In contrast Gerschgorin's disk theorem only requires inspection of the data matrix and computed an upper bound in 0.11 seconds. Finally, the trace operator returned an upper bound within 0.02 seconds whose quality highly depends on the number of dominant eigenvalues that increases with the number of objects in the scene. For the prototypical 2D setup of Fig. 4.2 the real Lipschitz constant, determined by power iterations is approximately

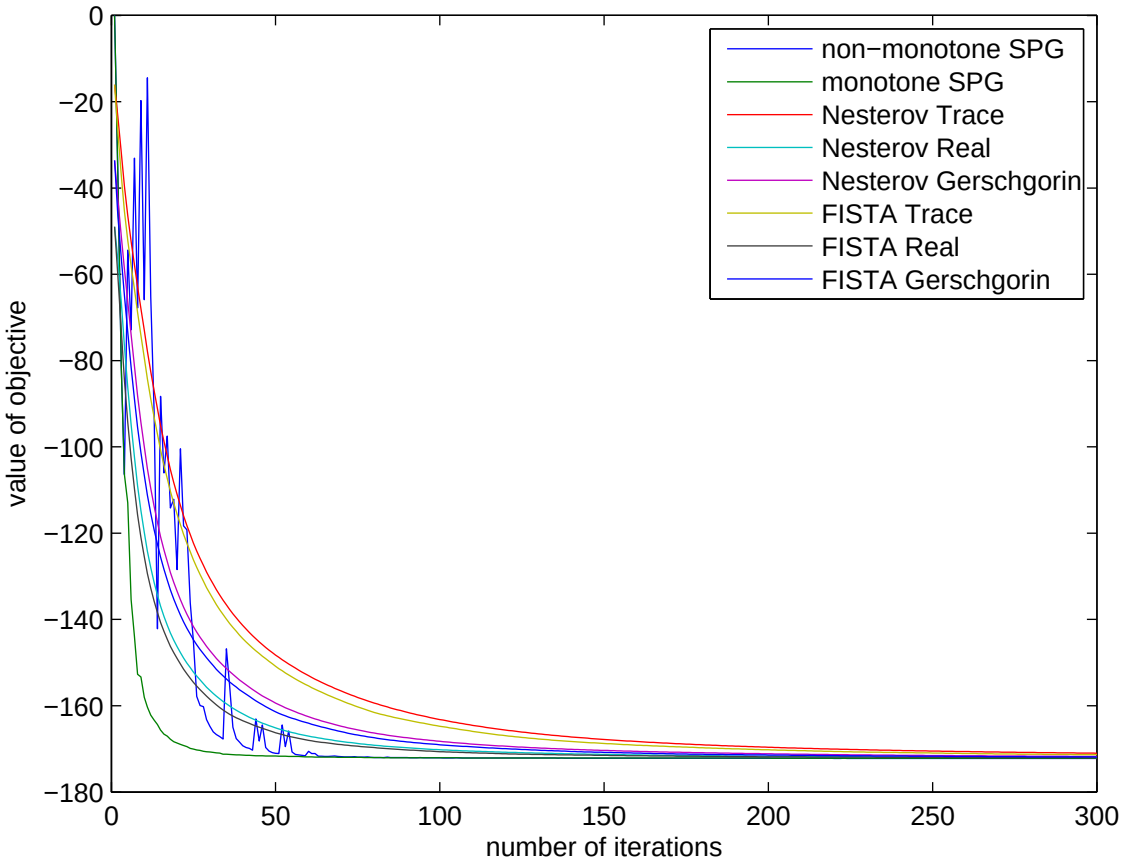


Figure 5.2: Comparison of optimization algorithms using the experimental setup of Fig. 4.2. The SPG algorithm typically requires far less iterations to converge to the global optimum compared to FISTA and Nesterov's approach at the costs of additional function evaluations involved in the line search. The performance of FISTA and Nesterov's method highly depend on the accuracy of the upper bound on L.

$2.4 \cdot 10^4$ whereas the upper bounds obtained by Gerschgorin's disk theorem and the trace operator are $4.8 \cdot 10^4$ and $1.2 \cdot 10^5$, respectively.

Our numerical experiments confirmed that the value chosen for L highly influences the performance of FISTA and Nesterov's algorithm, see Fig. 5.2. Compared to SPG, that does not depend on L, FISTA and Nesterov's approach also need more iterations to converge to an optimizer. The fast convergence of SPG is primarily due to the line search involved that causes the costs of additional function evaluations, however, such that the time per iteration is significantly smaller for FISTA and Nesterov's approach. This results in a typically faster overall convergence of FISTA and Nesterov's approach (Fig. 5.3).

For the real world scenario shown in the upper panel of Fig. 1.1, the dependency of the performance of FISTA and Nesterov's algorithm on L becomes even clearer as depicted in Fig. 5.4. As more object instances are available in this scene than in the 2D setup, the trace operator returns a poor upper bound on L yielding slow convergence of FISTA and Nesterov's approach, respectively.

Although FISTA reveals slightly faster convergence in our experiments, due to the theo-

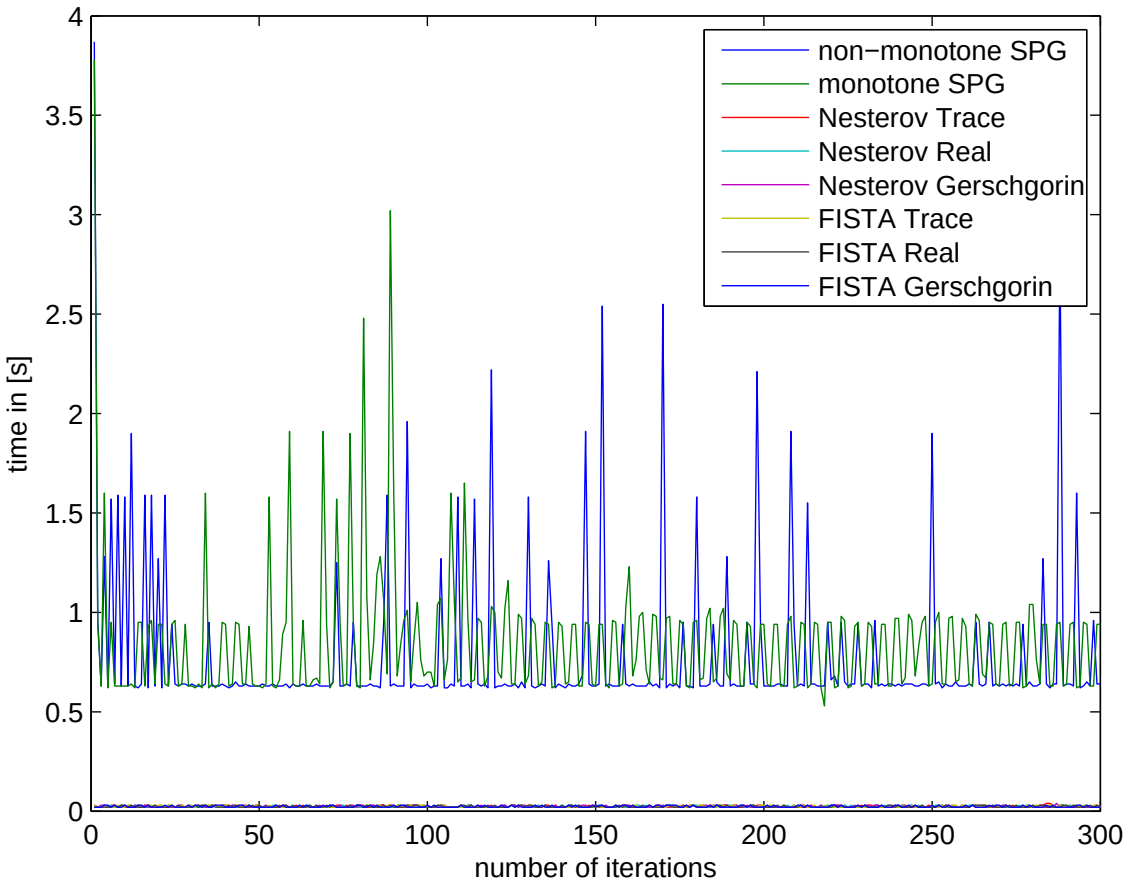


Figure 5.3: Comparison of optimization algorithms using the experimental setup of Fig. 4.2: While the SPG algorithm typically requires less iterations to converge to the global optimum, due to the line search involved, it requires more time per iteration (in seconds) in comparison to other approaches.

retical tighter bounds of Nesterov's algorithm (cf. Sec. 4.4), for our real world experiments summarized in Sec. 5.3, we throughout used Nesterov's algorithm to determine the solution of the convex initialization problem.

5.1.3 Binarization of the Solution

To infer a corresponding high-quality discrete configuration based on the real-valued solutions of the convex initialization problem, in our real world experiments we used the simple k-means clustering approach presented in Sec. 4.5 taking into account the underlying geometry of the manifold of Euclidean transformations.

Due to the compactness of the clusters in the model-pose space that can be well-localized in the image domain (see Fig. 4.6 as a prototypical example), simple clustering turns out to reveal reasonable initializations for subsequent refinement procedures as we will see in Sec. 5.3. Moreover, this approach does not require solving the large-scale optimization problem multiple times as is the case for randomized rounding techniques, and thus provides a compromise between accuracy of initialization and computational speed.

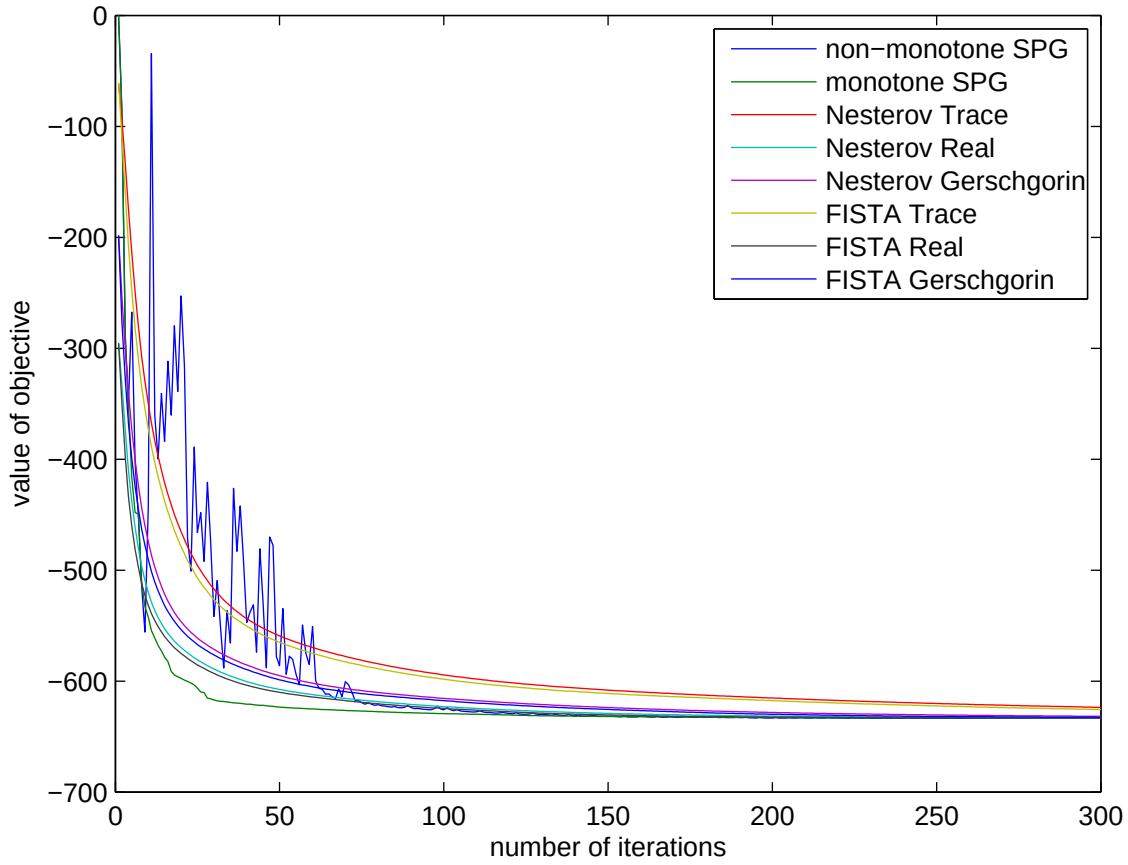


Figure 5.4: Comparison of optimization algorithms using the experimental setup shown on the upper panel of Fig. 1.1: While the SPG algorithm outperforms FISTA and Nesterov's approach, the performance of these algorithms highly depend on the accuracy of the estimate of L. As there are more object instances in the scene, compared to Fig. 4.2, the trace operator yields poor approximations on L causing slow convergence of the algorithms.

5.2 Fine Registration

To evaluate the refinement of rough estimates, we considered geometric optimization procedures as described in Chap. 3 with respect to the formulation of the Euclidean alignment problem of Chap. 2. To this end, we used computer-generated point sets obtained from [123] in a fully controlled environment, and analyzed the performance with respect to run-time and robustness to inaccurate initializations as well as comparing the results to state-of-the-art algorithms.

5.2.1 Accuracy of Registration with Ground Truth

In this subsection, we compare our formulation of the Euclidean point set alignment problem (see Chap. 2) to established state-of-the-art algorithms, including ICP [14], the fix-point iteration as a special case of Soft Assign [96], an expectation-maximization variant of ICP [57], and Kernel Correlation [70].

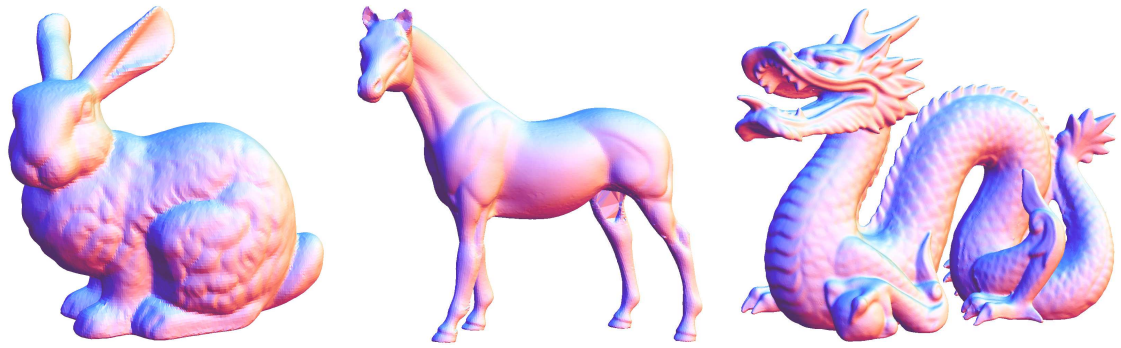


Figure 5.5: Visualization of the artificial models [123] used to evaluate the performance of the proposed alignment procedures with respect to accuracy. To speed up the alignment process of the corresponding algorithms, all models were down-sampled to a set of 200 point measurements.

For the model measurements we used the synthetic, computer generated models of Fig. 5.5. Each model consists of up to 50.000 samples. To speed up the whole process, we randomly sampled the models down to the size of 200 points. Scenes are generated by randomly placing a single model arbitrarily in space. As the primary concern in industrial bin-picking applications are outliers, we randomly added up to 50 % outliers to each object in order to simulate this issue.

As we focus on the accuracy of the problem formulation in this subsection and do not want to cope with false alignments due to algorithmic issues (cf. Sec. 5.2.4), to obtain the optimizer of our novel, smooth formulation presented in Chap. 2, we used a simple extension to gradient descent like algorithms such as Alg. 4 (Sec. 3.3.3).

In order to analyze all algorithms with respect to sensitivity to initialization, we repeated the alignment process 150 times, where for each iteration the scene was transformed to a different location. Random rotations were obtained by sampling the complete space of rotations, while the translation varied randomly between 2 times the diameter of the model.

Two sets were considered to be properly aligned if the length of the curve connecting both transformations was less than 0.2. In Euler angles, this distance corresponds to an error of about 5 degrees in each angle and a total deviation of 5 % of the model's size in translation. This was chosen empirically based on our experience with the industrial application.

Furthermore, the parameters necessary for most algorithms were all tuned by hand to optimize performance and to guarantee a fair comparison. Accordingly, with respect to the annealing schedules required for the Soft Assign [96] and the expectation-maximization variant of ICP [57], we adopted conservative schedules at the cost of slower convergence in order to better escape from local minima.

The best choice of the parameter σ, required for our approach, cannot be specified in general but depends on the given data and its scale. As a rule, too small values will impair robustness of the approach, making it sensitive to local minima like the ICP algorithm (cf. Sec. 2.4). Exceedingly large values, on the other hand, limit the accuracy of the registration. Our experiments show the existence of a reasonably large interval of values resulting in good performance (see Fig. 2.7).

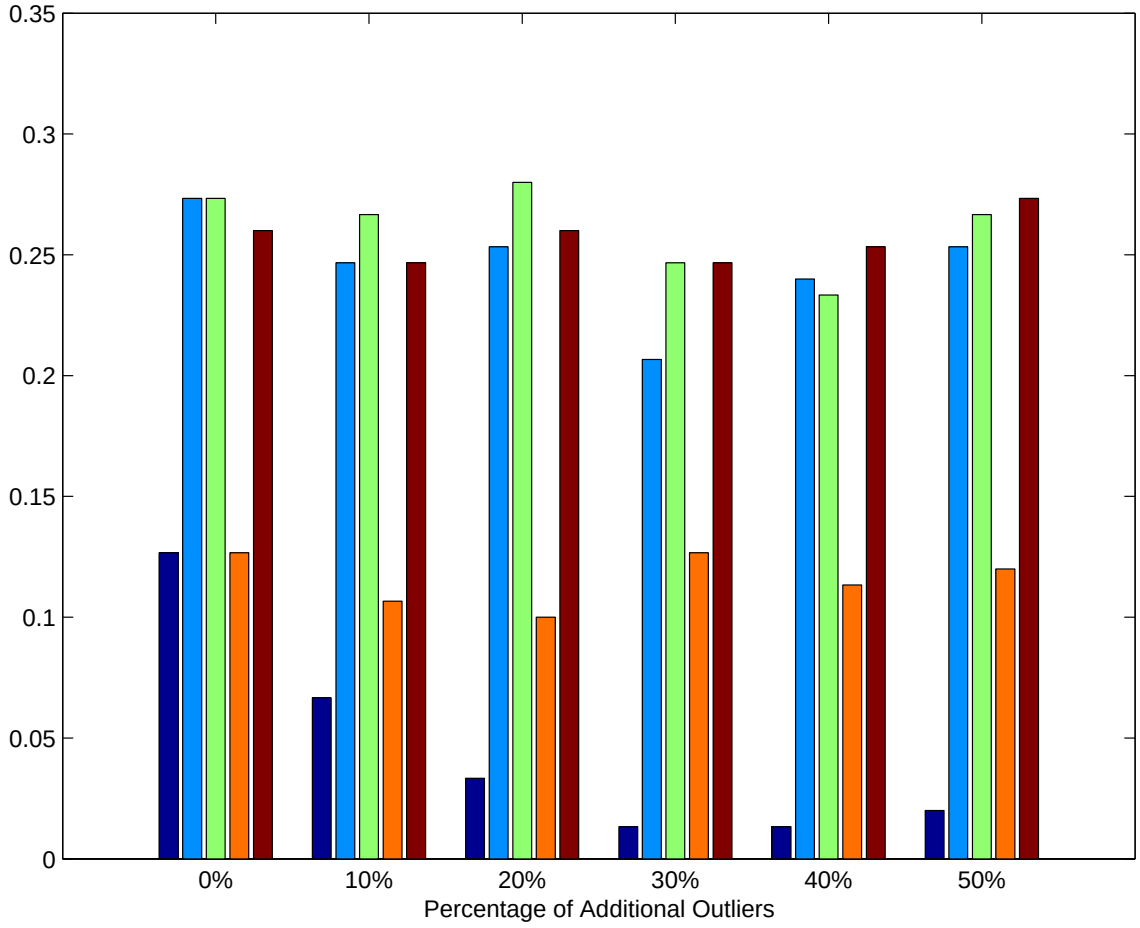


Figure 5.6: Percentage of experiments that converged for the bunny model of Fig. 5.5 to the true solution by applying from left to right ICP (dark blue), EM-ICP (light blue), Soft Assign (green), Kernel Correlation (red), and our approach (brown). All corresponding scenes were supplemented with noisy points in the range up to 50 % of the model size. Our approach outperforms state-of-the-art approaches such as ICP and Kernel Correlation without using an annealing schedule.

The results of our experiments for the different data sets of Fig. 5.5 are presented in Fig. 5.6, Fig. 5.7, and Fig. 5.8, respectively. They show that our novel objective function reveals a significantly increased robustness against inaccurate initializations and uniformly outperforms related state-of-the-art approaches like ICP and Kernel Correlation. Moreover, the proposed reformulation of the Euclidean alignment problem reveals comparable performance to Soft Assign and the expectation-maximization variant of ICP without using an annealing schedule.

To demonstrate the robustness of our novel objective function to structured outliers, we also applied the registration procedure to the horse model with only a small overlap as visualized in Fig. 5.9. An additional background kernel supplemented to the model mixture (cf. Sec. 2.4.1) allows to accurately align both point sets and reconstruct the original model.

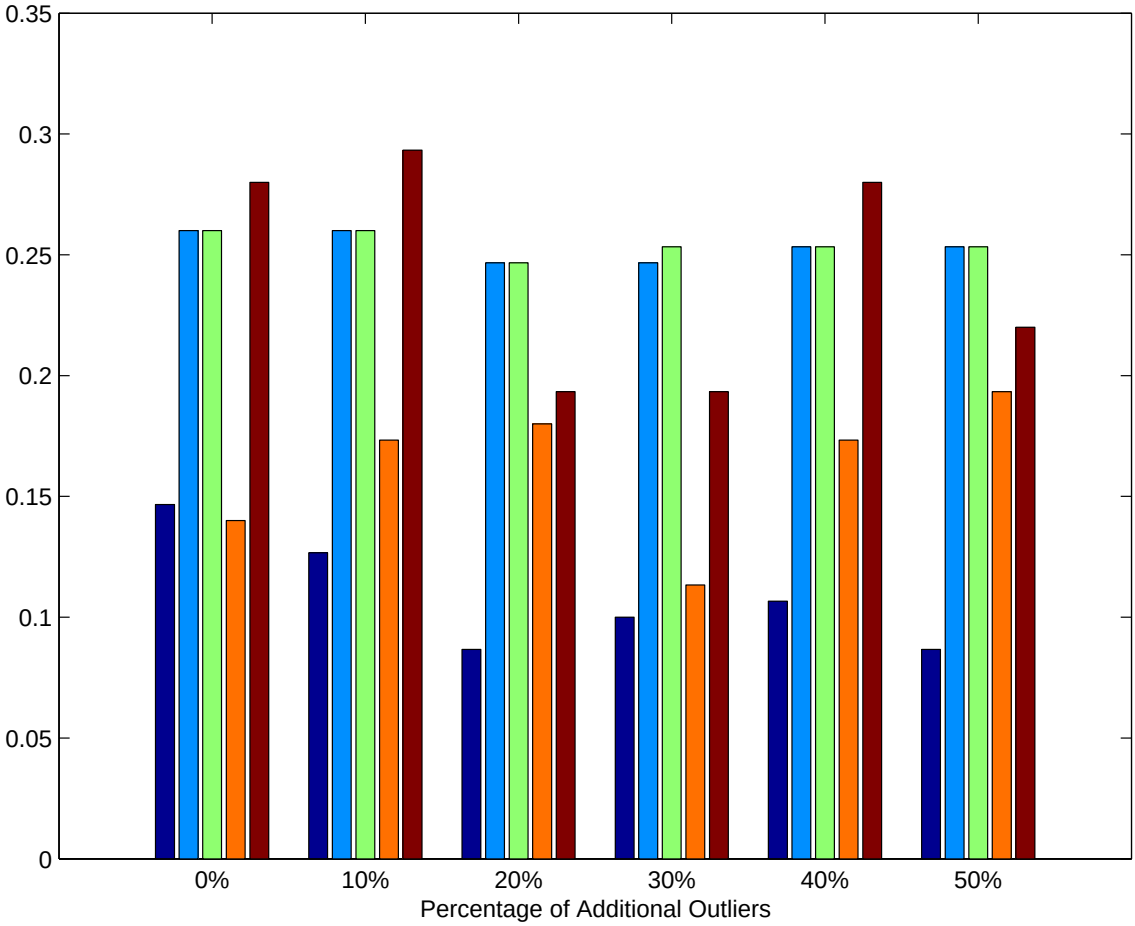


Figure 5.7: Percentage of experiments that converged for the dragon model of Fig. 5.5 to the true solution by applying from left to right ICP (dark blue), EM-ICP (light blue), Soft Assign (green), Kernel Correlation (red), and our approach (brown). All corresponding scenes were supplemented with noisy points in the range up to 50 % of the model size. Our approach outperforms state-of-the-art approaches such as ICP and Kernel Correlation without using an annealing schedule.

Finally, the time required to evaluate our objective function is similar to the computational complexity for computing the value of the objective of the expectation-maximization variant of ICP, Soft Assign, and Kernel Correlation without using speed-up techniques, namely $O(nm)$, where n, m are the number of scene and model samples, respectively.

5.2.2 Accuracy for Real World Applications

In real industrial applications, the rigid alignment of a model point set to scene samples becomes more difficult due to the type of noise. As Fig. 1.1 shows, the scene recorded by a SICK-LMS400 scanning device contains noise as well as multiple structures similar to model parts. Most structures have to be considered as outliers except for those corresponding to the model. Quantitatively, nearly 80% of the samples are structures not belonging to the object of interest.

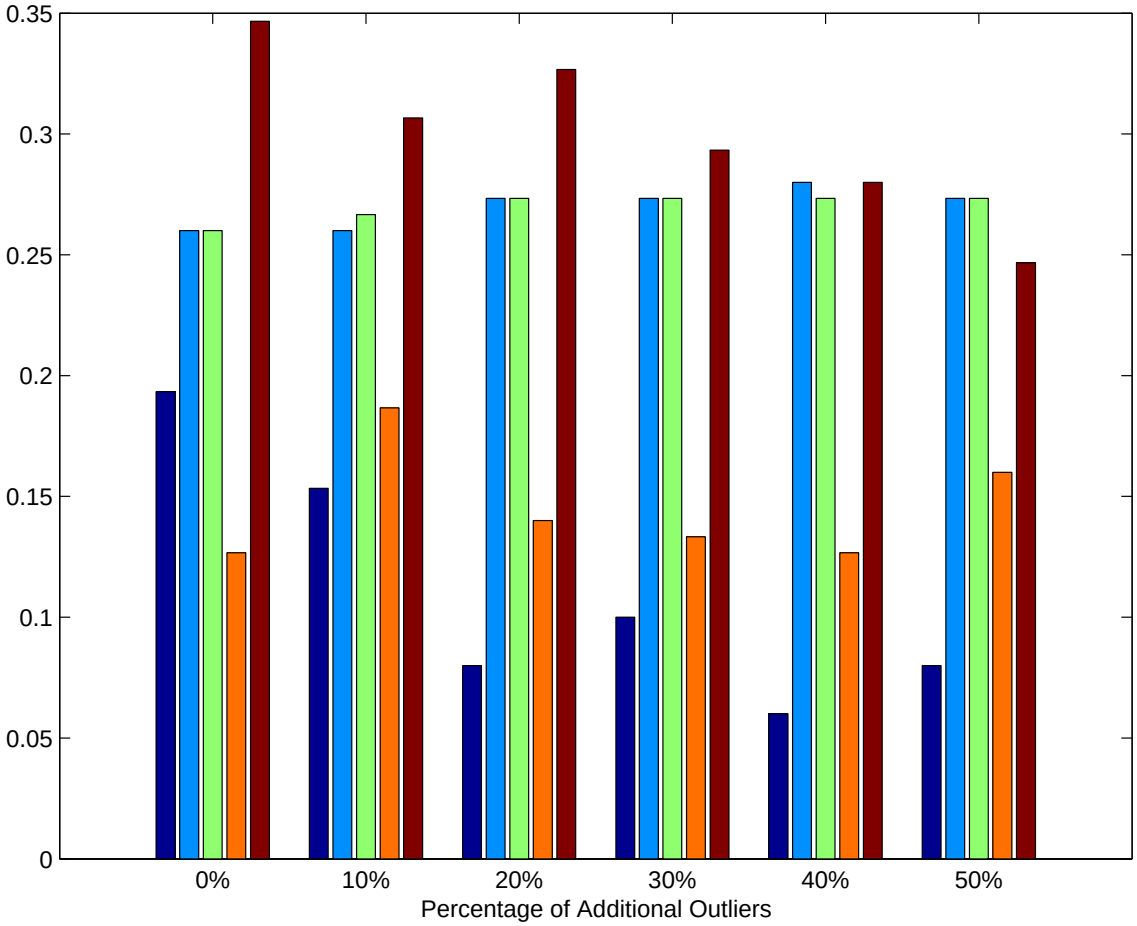


Figure 5.8: Percentage of experiments that converged for the horse model of Fig. 5.5 to the true solution by applying from left to right ICP (dark blue), EM-ICP (light blue), Soft Assign (green), Kernel Correlation (red), and our approach (brown). All corresponding scenes were supplemented with noisy points in the range up to 50 % of the model size. Our approach outperforms state-of-the-art approaches such as ICP and Kernel Correlation without using an annealing schedule.

In order to alleviate these issues, a common procedure is to fit the scene to the model instead of fitting the model to the scene, and to introduce a further background kernel as already mentioned previously. This improves the capability to properly assign even occluded samples to parts.

In the particular scenario depicted on the upper panel of Fig. 1.1, experts from industry expect a maximum variability of rotation of ± 6 degree. Consequently, to be on the safe side, we randomly sampled the rotation space within ± 25 degrees around the model reference position and placed the model at various locations in the scene. As with the synthetic experiments of Sec. 5.2.1, we tuned the parameters by hand.

The results of applying our approach as well as a robust implementation of ICP [104] are visualized in Fig. 5.10. The statistics resulting from these experiments, presented in Tab. 5.1, reveal that according to our objective function 36 of 50 experiments converged to

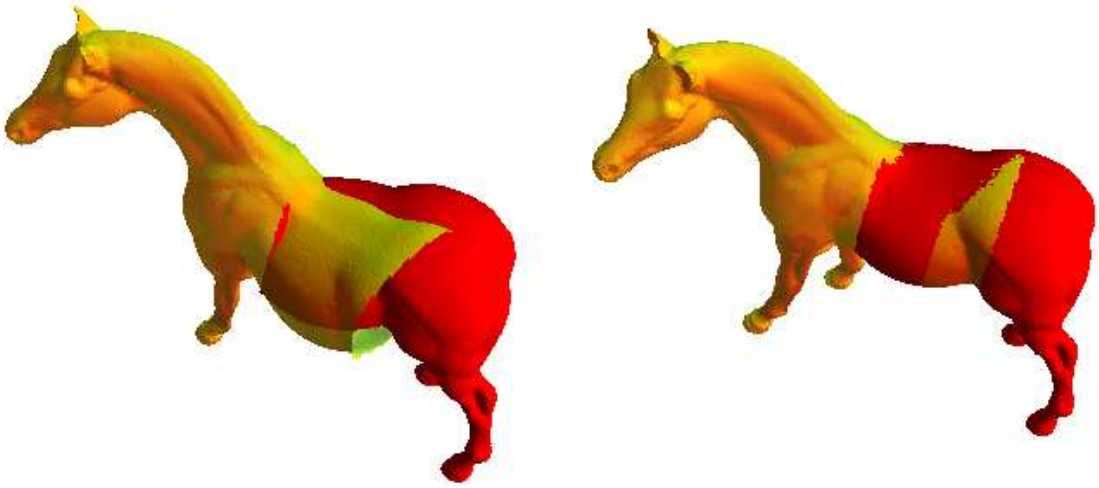


Figure 5.9: To demonstrate the ability of our novel objective function presented in Chap. 2 to handle highly structured outliers we used a discrete set of points sampled from slightly overlapping back and head of horse as scene and model, respectively (left). Instead of fitting the mean values of the point sets, our method accurately merges both point sets and reconstructs the original model (right).

	Our Approach	Robust ICP [104]
# located objects	7	4
# positive detections	36	12
# negative detections	14	38

Table 5.1: Statistical results of aligning a brake disc model to real world data for our approach and a robust implementation of ICP using outlier rejection [104].

a visually correct solution. In contrast, the robust implementation of ICP only converged in 12 of 50 experiments.

This also leads to the fact that our approach locates 7 of 8 model instances in the scene with at least one experiment. The robust implementation of the ICP approach, in contrast, is only able to extract 4 different model instances accurately. Even more complicated models comprising non-symmetries can be handled using our novel objective function, where we demonstrated in Fig. 5.11 the applicability to automatic processes. The model, depicted on the left hand side of Fig. 5.11 that is obtained by a reference scan (cf. Fig. 5.1) is used to specify the transformation to subsequent sample scans. For the initialization we used the recorded position of the model.

5.2.3 Newton-Like Geometric Fine Alignment

Even though these experiments reveal promising results with respect to accuracy of the obtained optimizer, using stabilized versions of gradient descent like algorithms typically yield poor results with respect to speed of convergence and leave room for improvement by considering higher order methods.

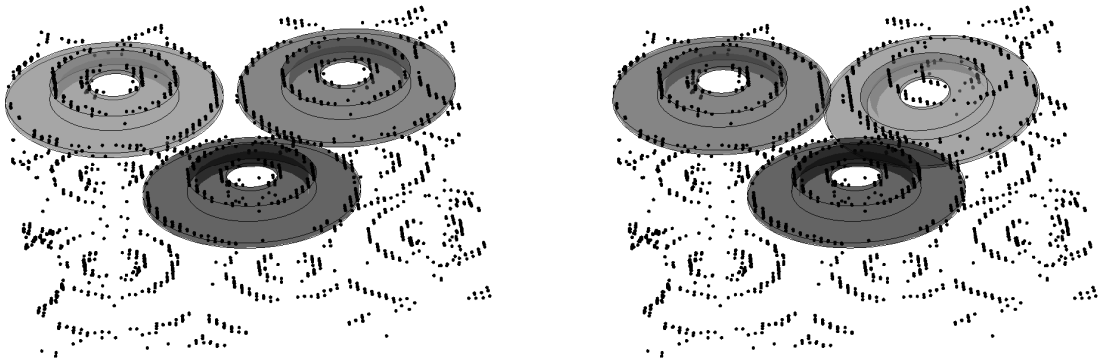


Figure 5.10: Typical results of aligning a brake disc model to sample scans obtained by a SICK-LMS400 scanning device using our novel cost function (left) and the robust ICP algorithm [104] (right). While our approach locates the disk in the upper right corner accurately, robust ICP failed to detect it for the same initialization. This prototypical behavior is primarily due the high non-convexity of the ICP objective and can also be seen for different data sets.

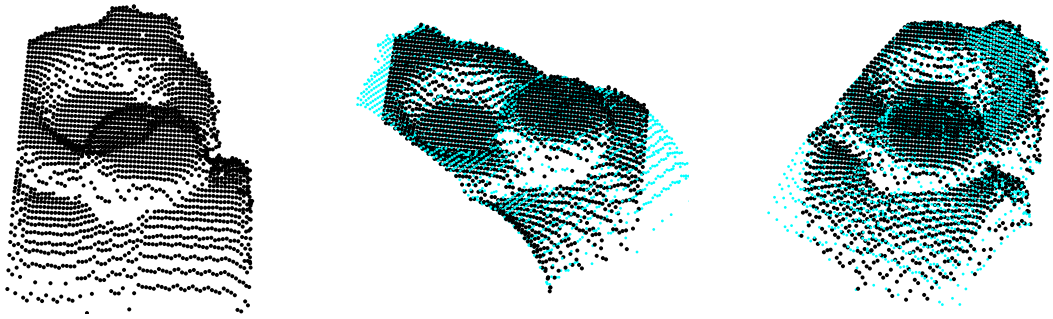


Figure 5.11: Typical results of aligning a model, recorded by a SICK-LMS400 scanning device (left), to subsequent scans of the same object (middle and right) using our novel objective function. The model (light blue) is accurately fitted to the recorded scene (black), even if parts of the scene are missing.

To this end, we evaluate the Newton-like geometric optimization algorithms described in Sec. 3.4 by applying them to computer-generated point sets such as the bunny and the dragon model of Fig. 5.5, and analyze the performance with respect to run-time and robustness to inaccurate initializations.

Speed of Convergence

Algorithms like ICP [14] or Soft Assign [96] return less accurate registrations in cases where the underlying point set has no or only few salient regions. This often occurs in industrial applications where smooth surfaces have to be aligned accurately. To compare the ability of the Newton-like approaches to cope with such scenarios, we generated 2500

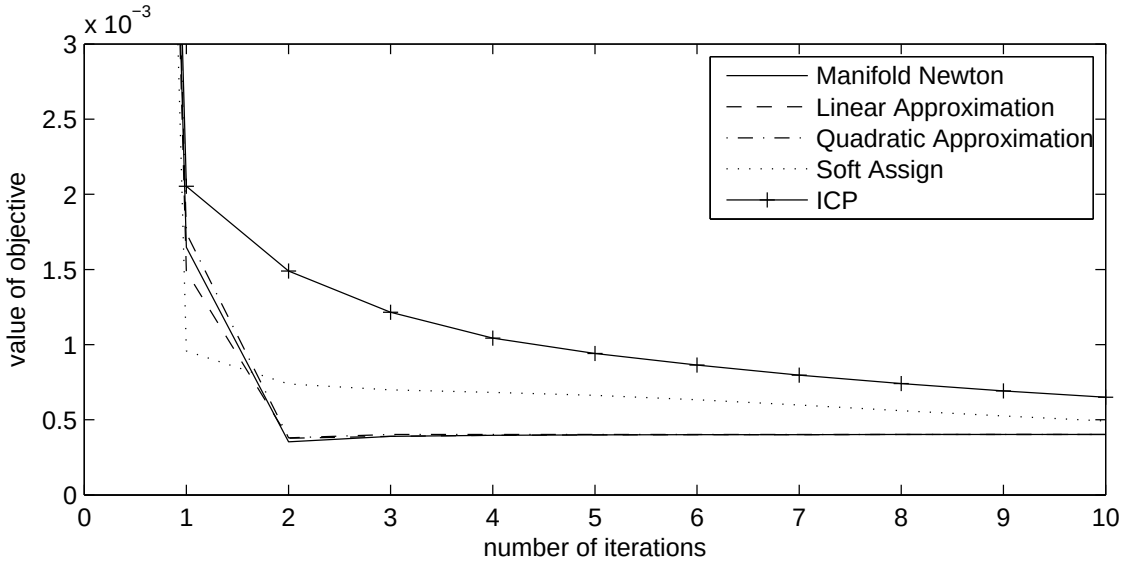


Figure 5.12: Evaluation of the performance of Newton algorithms for aligning two point sets sampled from the smooth function $3(x-1)^2+3\sin(2y)$ on the unit interval $[0,1]^2$. The algorithms are based on linear and quadratic motion approximation [94], the approach proposed in this paper (manifold Newton) as well as ICP [14] and Soft Assign [96], for $\sigma_m = 0.3$, where the plot visualizes the objective function values for the ICP functional for subsequent iterates. While ICP and Soft Assign converge linearly, the remaining approaches converge quadratically to the local optimizer.

data points by randomly sampling from the smooth function $3(x-1)^2 + 3\sin(2y)$ on the unit interval $[0,1]^2$.

We transformed a copy of the model only slightly (about 4 degree in each rotation and by a total of 0.12 in translation), such that all approaches including ICP [14], the fix-point iteration as a special case of Soft Assign [96], the Newton schemes based on local approximation [94] (cf. Sec. 3.4.2), and the approach proposed in Sec. 3.4.3 converged to the true solution. Figure 5.12 and Fig. 5.13 reveal that the convergence rates differ significantly in that with increasing σ_m the local optimizer is reached in few iterates for all algorithms depending on the control parameter.

While for varying σ_m the Newton procedures based on local approximations of the Euclidean group (Sec. 3.4.2) converge slightly faster than the approach presented in Sec. 3.4.3, all of them exhibit quadratic convergence. In contrast, ICP and Soft Assign only converge linearly to the optimal configuration. As a result, they return less accurate registrations under tight run-time constraints such as performing the algorithms for a fixed number of iterations.

The superior performance of the Newton schemes is at the cost of more expensive computations for determining the Hessian in each iteration. A single round of ICP requires about 1 second. In contrast, the computation of the derivatives using only MatLab research code needs between 8 (linear and quadratic approximation [94]) and 12 seconds (our approach). This difference is primarily due to the higher dimension of the ambient space in which the gradient and the Hessian are computed. We expect however that when

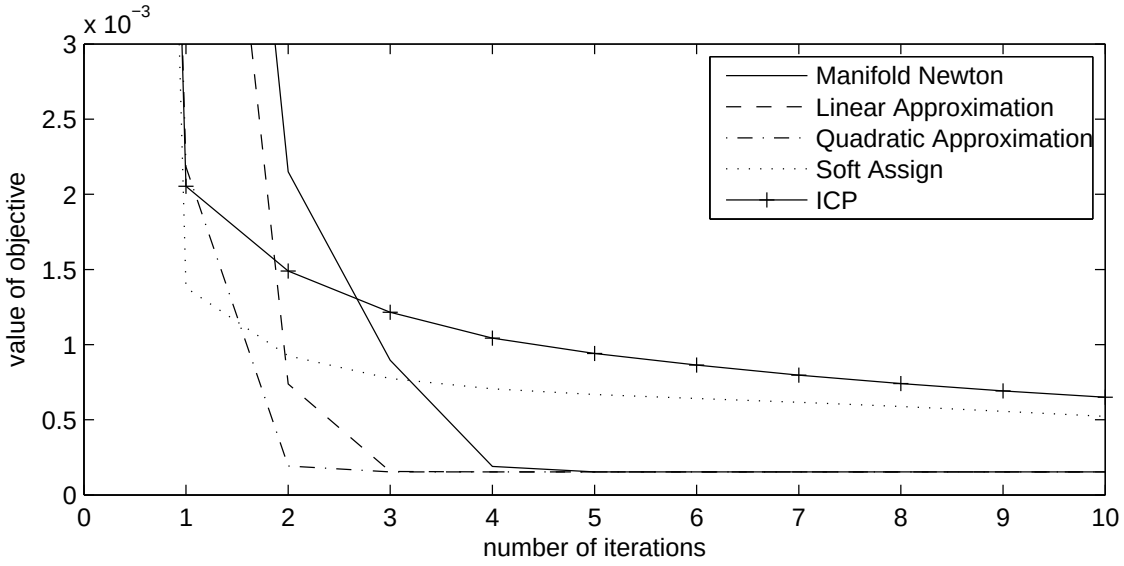


Figure 5.13: Evaluation of the performance of Newton algorithms for aligning two point sets sampled from the smooth function $3(x-1)^2+3\sin(2y)$ on the unit interval $[0,1]^2$. The algorithms are based on linear and quadratic motion approximation [94], the approach proposed in this paper (manifold Newton) as well as ICP [14] and Soft Assign [96], for $\sigma_m = 0.15$, where the plot visualizes the objective function values for subsequent iterates. While ICP and Soft Assign converge linearly, the remaining approaches converge quadratically to the local optimizer. Compared to Fig. 5.12 with increasing σ_m more iterates are required to ensure convergence.

using a C-tuned implementation the Newton approaches will almost catch up with ICP.

Region of Attraction

Fast convergence is immaterial if the algorithm diverges or gets stuck in a wrong local minimum. Robustness to poor initializations is therefore important. The region of attraction for ICP [14] has already been analyzed in [84]. We therefore only consider Newton procedures here.

For comparison, we used the same initial setup as [84], i.e. a model of the Stanford Bunny visualized in Fig. 5.5 is rotated around the z-axis and shifted in the x-y plane by the size of the model, see Fig. 5.14. For the scene we used a copy of the model placed in the origin. Because we are primarily interested in quadratic and fast convergence and the resulting accuracy after a fixed run-time, we terminated all second-order algorithms after 25 iterations.

We observed that especially for transformations with rotational initialization error, the Newton approach proposed in this work has a significantly larger domain of attraction to the correct solution than the procedures based on local approximations of the Euclidean group, as visualized in Fig. 5.15. This finding confirms the discussion in Sec. 3.4.4.

In a related experiment we examined the update direction of a single iterate of each scheme, cf. Fig. 5.16. By only translating the bunny point set in $\mathbb{R}^3$ we found that quadratic

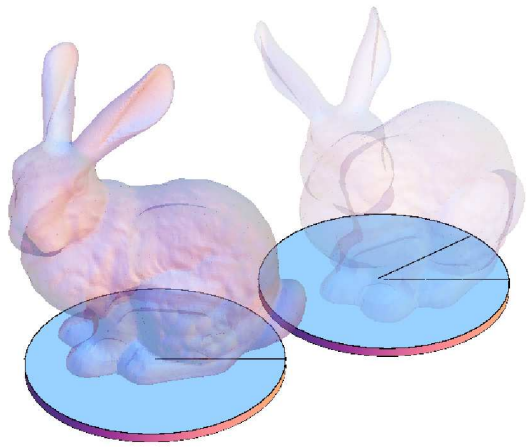


Figure 5.14: Visualization of the experimental setup [84] for evaluating the region of quadratic convergence for Newton algorithms. Each circle center relative to the centering circle, refers to an initial translation offset of the model vs. the scene in the x-y plane.

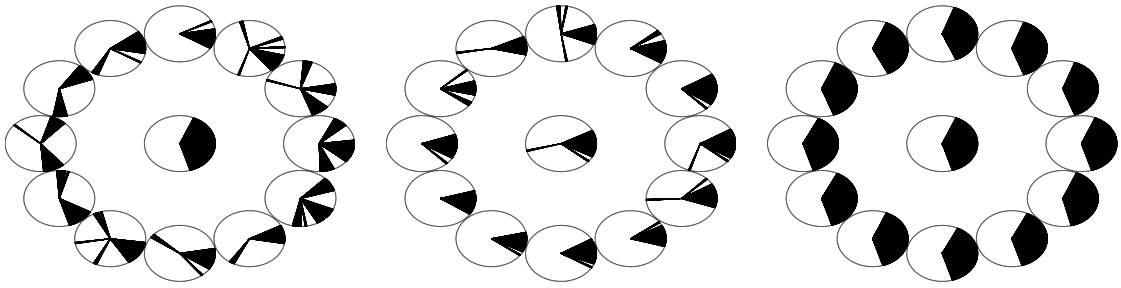


Figure 5.15: Bunny point set: Evaluation of the region of quadratic convergence for Newton algorithms based on linear (left), quadratic (middle) local approximation [94], and on the intrinsic local approximation (Sec. 3.4.3, right), for fixed $\sigma_m = 0.1$. Each circle center together with the circle center in the middle shows the initial translation offset of the model vs. the scene in the x-y plane. Slices in each circle refer to the initial rotation around the z-axis. They are colored black if the model converged to the scene within the first few iterations and otherwise remained white. The results illustrate that the approach proposed in Sec. 3.4.3 is significantly and uniformly more robust against inaccuracies of initialization.

local motion approximation as well as our approach exhibit a lower angular error than the scheme based on linear local approximation. We point out that the angular error of all approaches near the origin is primarily due to the nature of the objective function of Chap. 2, that is a slight detrimental effect of the smoothed objective function discussed in Sec. 2.4. Decreasing the value of σ_m after few iterations would fix this minor issue.

Applying this setup to a different model, such as the dragon point set of Fig. 5.5, reveals similar results (see Fig. 5.17). Even though the region of attraction is less dense than the one of Fig. 5.15, compared to approximating the Euclidean motion group our approach (cf. Sec. 3.4.3) reveals promising performance for rigid point set registration problems.

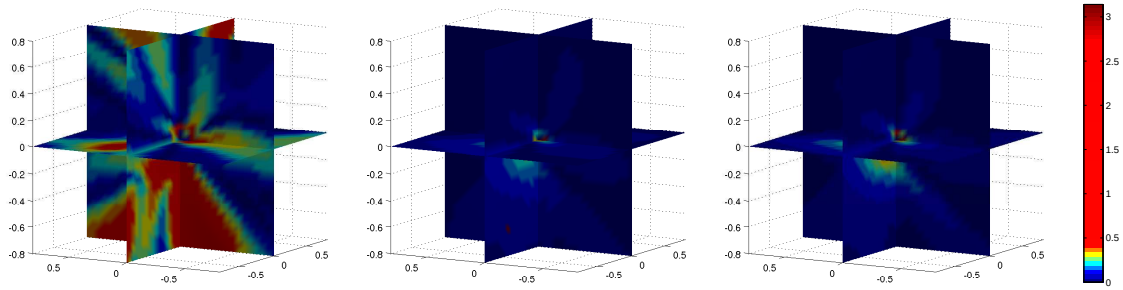


Figure 5.16: Visualization of the angular error of the translational update computed with the linear (left) and quadratic (middle) local approximation approach [94], and with the intrinsic local approximation (right) (Sec. 3.4.3), as a function of the translational offset (ground truth) model ↔ scene in 3D-space. No rotation was applied. Each graphic depicts slices through the three-dimensional "error fields". While the linear local approximation fails again in this simple scenario, both quadratic approximations are more robust against this type of initialization error. Figure 5.15 shows, however, that only the intrinsic approximation (Sec. 3.4.3) remains stable if rotational initialization errors additionally occur.

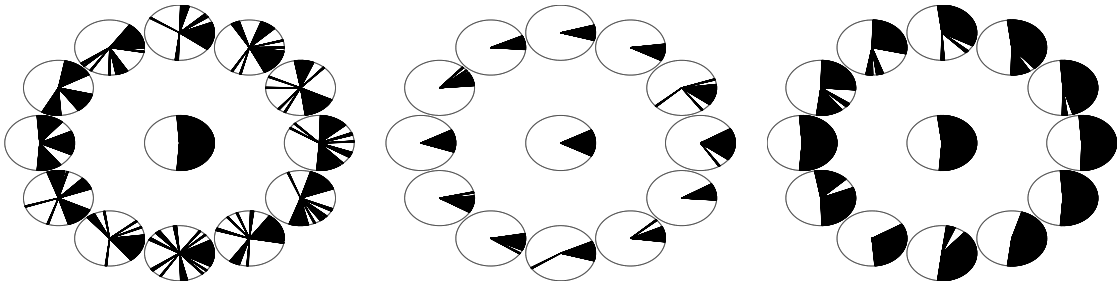


Figure 5.17: Dragon point set: Evaluation of the region of quadratic convergence for Newton algorithms based on linear (left), quadratic (middle) local approximation [94], and on the intrinsic local approximation (Sec. 3.4.3, right), for fixed $\sigma_m = 0.1$ using the initial setup of Fig. 5.15.

5.2.4 Limitations of our Approach

Despite of the increased robustness in comparison to ICP, the expectation-maximization variant of ICP, Soft Assign, and the Kernel Correlation approach, our approach still may also fail for very inaccurate initializations, cf. Fig. 5.6, Fig. 5.7, and Fig. 5.8. Real world results where our approach failed to converge to true solutions are shown in Fig. 5.18.

In these cases, objects are placed such that parts of the model accurately fit various salient structures in the scene that do not belong to a single object instance. Due to the local influence of the kernels involved in our objective function, we cannot currently escape from such pronounced local optima.

Another point is that due to the high non-convexity of the objective function applying Newton-like algorithms might cause poor approximations of the objective function such

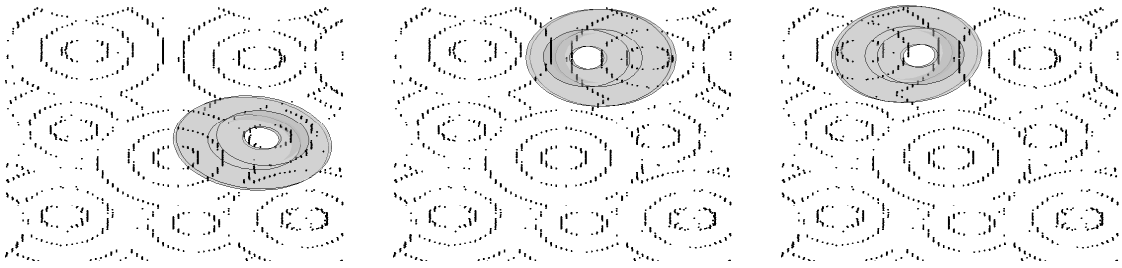


Figure 5.18: Typical negative detections occurring during experiments with too inaccurate initializations. Salient structures of different objects are placed, such that they fit to few parts of the model. Similar to state-of-the-art approaches, this causes our algorithm to get stuck in local optima.

that the corresponding update directions fail to decrease the value of the objective function.

A straightforward extension concerns the interplay between first- and second-order numerical optimization methods on the manifold of Euclidean transforms in order to optimize the speed of convergence while guaranteeing convergence to a local optimum. This goal can be accomplished by adopting numerical trust-region strategies to the manifold setting as previously pointed out in Sec. 3.5.

5.3 Combining Global and Local Optimization

Finally, we demonstrate that the single steps evaluated in Sec. 5.1 and Sec. 5.2 complement each other in that conflicting interpretations are resolved through non-local convex processing, followed by accurate non-convex local optimization based on sufficiently good initializations. Hence, in this section, we apply and evaluate the combination of the initialization and geometric refinement approach to the real-world bin-picking scenario. To this end, we use both computer-generated data allowing for full control of the evaluation by simulating the scanning device and noise, and real industrial data.

5.3.1 Computer-Generated Data

To evaluate the accuracy of our approach in a fully controlled environment, we generated different "realistic" data sets by simulating the real world scanning device of Fig. 1.1 for real objects depicted in Fig. 1.2.

Each object instance was randomly placed in the scene including partially overlapping objects. Additionally, to cover a wide range of applications with different input data, we used both object models exclusively based on edge data as well as models just obtained by reference scans, as depicted in Fig. 5.19, Fig. 5.20, and Fig. 5.21, respectively. We point out again that different input formats are uniformly handled by our approach.

Collections of candidate poses for multiple object detection were compiled by discretizing the space of possible rotations in intervals of 15°. The corresponding translation vectors are chosen such that at least a single point in the scene can be fitted accurately. This resulted in a total of up to 2 413 675 possible candidate instances. Table 5.2 displays all relevant numbers.

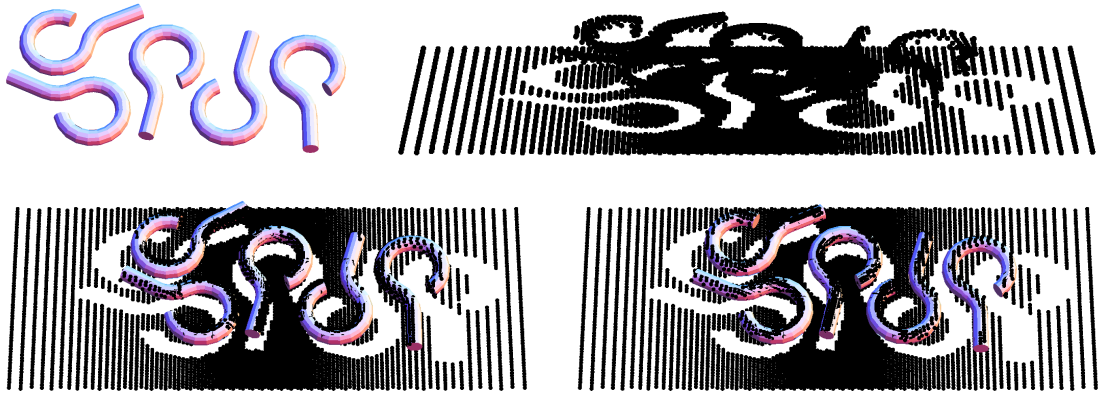


Figure 5.19: Object detection and localization with real world objects (top left) in 3D scanning data obtained by simulating a SICK-LMS400 scanning device (top right). While the convex initialization step simultaneously gives proper estimates of the number of objects as well as the corresponding transformations (bottom left), running subsequently few iterations of the geometric optimization approach yields accurate registration results (bottom right).

Object	Link	Hook	Mech-part
# candidate instances	1 524 600	2 413 675	394 975
# non-zero instances after pruning	228	336 681	4 597
# non-zero instances after optimization	9	47	11
# non-zero instances after clustering	5	5	5

Table 5.2: Quantitative evaluation of the initialization phase and of the first processing stage (non-local multiple object detection through convex optimization) for the data sets shown in Fig. 5.19, Fig. 5.20, and Fig. 5.21.

Our current research code does not yet exploit the features for accelerating the initialization phase, as listed in Sec. 5.1.1. Rather, we computed the full matrix A off-line which took several minutes.

The parameters δ, σ, and μ of our approach (cf. Sec. 4.2) are set by hand for each scenario. These values reflect the characteristics of the scenario, i.e. the noise level and the spacing of the model points. Their choice is therefore straightforward and does not require elaborate tuning. We point out that they only depend on the scenario (noise, object models) and not on the specific given scene (data) of a fixed scenario to be analyzed.

The elimination of variables in the preprocessing step (see Sec. 4.3) fixes between 86% and 99.9% of the variables beforehand such that the final convex optimization procedure detects multiple objects within few seconds only.

An additional pose clustering step provides rough initializations that can be used for the subsequent fine alignment process through geometric optimization. In case of the mechanical part (Fig. 5.20) the deviation of the estimated position from ground truth is at most 5° rotation and $\approx 5.3\%$ translation of the model size. Similar results are obtained for the link (rotation error $\leq 4.5°$, translation error $\leq 1.8\%$) and the hook data set (rotation error $\leq 5°$, translation error $\leq 2.4\%$). The trade-off between the computational costs of the first non-local convex processing stage and the subsequent geometric optimization

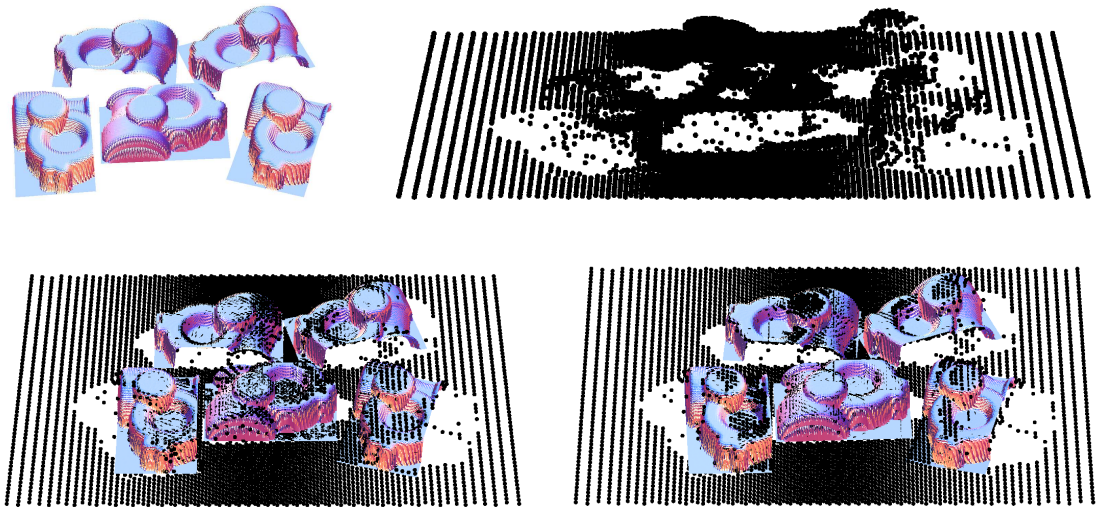


Figure 5.20: Object detection and localization with real world objects in 3D scanning data obtained by simulating a SICK-LMS400 scanning device. While the convex initialization step simultaneously gives proper estimates of the number of objects as well as the corresponding transformations (bottom left), running subsequently few iterations of the geometric optimization approach yields accurate registration results (bottom right).

depends on how finely the pose space is discretized (problems size vs. inaccurate detection) and can certainly be optimized for fixed industrial scenarios.

Running the geometric optimization algorithm evaluated in Sec. 5.2 for each detected object returns a final pose estimate within few iterations. At this second stage of the overall approach, we used an additional background kernel in our objective functional (cf. Sec. 2.4.1) to cope with structured outliers, i.e. nearby objects.

Because geometric optimization converges to a local optimum, occlusion configurations may occasionally lead to erroneous updates of the corresponding Newton algorithm. Figure 5.21 depicts such a scenario where due to locally "looking through holes" no consistent matching of the sparse model points to scene points is possible.

5.3.2 Real-World Industrial Data

We also applied our approach to the real-world industrial scenarios depicted in Fig. 1.1, comprising 3D noisy and unstructured scanning data of brake discs and flanges, respectively.

Assuming that brake disc objects are never located upside down, we sampled the model at 10 different points for each circle and discretized the space of rotations within the interval of $[-15, 15]$ degrees for each free axis (the model is rotation-invariant with respect to the third axis). This resulted in a total of 1420 candidate objects poses that can cause a single scene point and took about 0.2 seconds computation time.

The preprocessing step reduced the problem size by eliminating $\approx 99.5\%$ of the variables immediately. The subsequent global convex optimization determined the remaining 4320

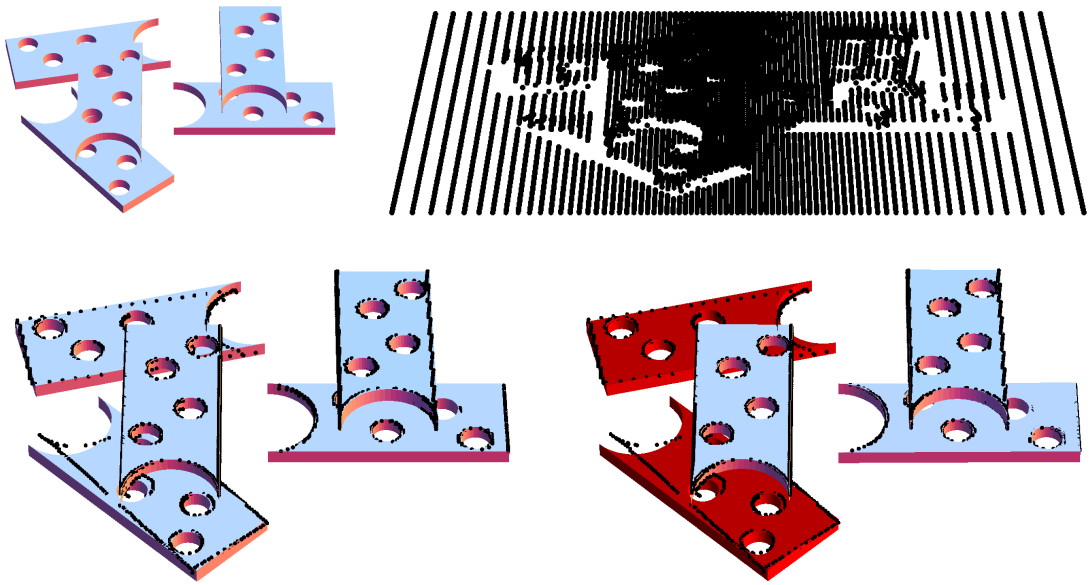


Figure 5.21: Object detection and localization with real world objects in 3D scanning data obtained by simulating a SICK-LMS400 scanning device. Due to the use of edge images, wrong edge detections due to noise ("looking through holes") yields Newton's algorithm to fail to converge. While the convex initialization step simultaneously gives proper estimates of the number of objects as well as the corresponding transformations (bottom left), running subsequently the geometric optimization approach fails to converge for the two objects marked red. The reason is that the ability of "looking through holes" complicates the objective function and narrows down the region of attraction to the correct local minimum.

variables in 14 seconds. Even highly occluded model instances are detected accurately as indicated by the blobs on the right hand side of Fig. 5.22 marking the hypotheses corresponding to the objects' pose.

Subsequent application of the geometric optimization for at most 5 iterations, where each iteration required about 1 second, turned out to be sufficient to accurately locate all object instances placed in the bin.

For the complex objects shown in the lower panel of Fig. 1.1, the approach was able to detect the objects and to determine proper initial pose estimates in the corresponding highly unstructured point set (see Fig. 5.23, left panel). Again, subsequent geometric optimization determined the final object positions within few iterations.

However, geometric optimization might fail if the initial pose estimate does not fall into the region of convergence of the Newton updates on the manifold, as indicated in Fig. 5.15 and Fig. 5.17. This fact is well known from standard Newton-based optimization in Euclidean spaces, too (cf. Sec. 3.5). We cope with this issue by resorting to first-order optimization techniques on the group of Euclidean transformations (cf. Chap. 3) if the initial Newton updates do not sufficiently decrease the objective function value. The object marked with red in Fig. 5.23 constitutes such an example, where the Newton method failed and switching to first-order optimization safely converged, at the cost of a higher number of iterations.

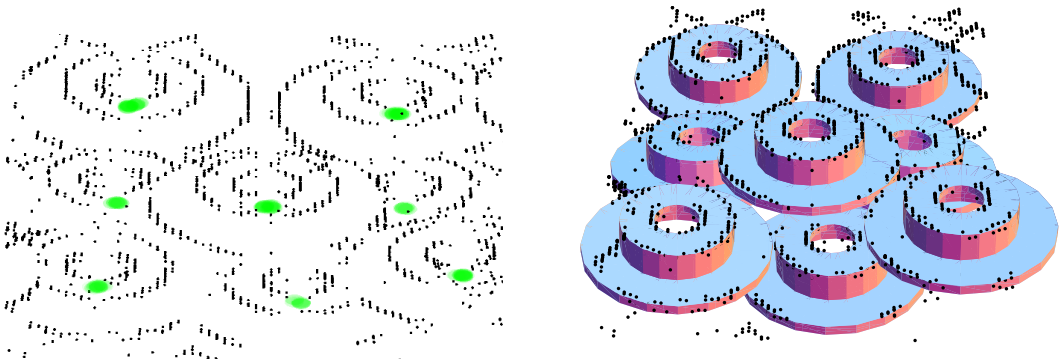


Figure 5.22: Object detection and localization with real world 3D scanning data. The pose clusters of all object instances recovered by convex global optimization are displayed as blobs in the left panel. Selecting a representative of each compact cluster as initialization enables to infer the unique number and localization of objects by subsequent geometric optimization (right panel).

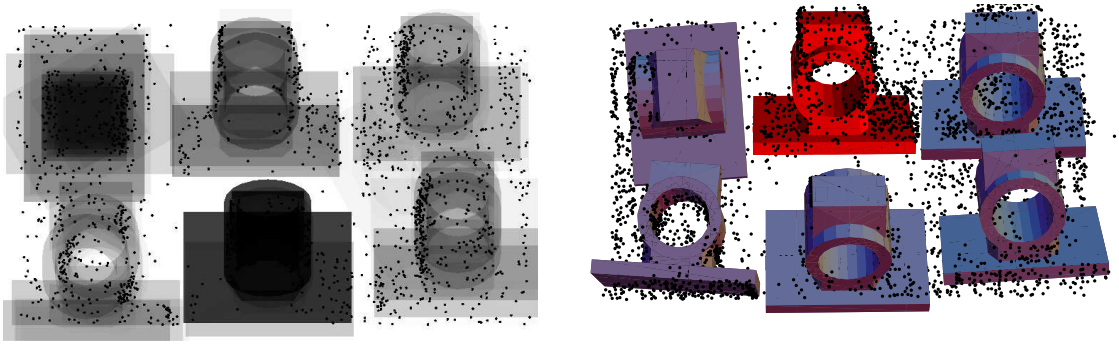


Figure 5.23: Detection and localization of complex objects in unstructured range data. Poses returned by the first convex global optimization stage cluster tightly and are displayed in the left panel, where more likely candidate poses are shown less transparent. Selecting a representative of each cluster as initialization enables to accurately locate the objects through subsequent geometric optimization, cf. right panel and the lower panel of Fig. 1.1.

Finally, we demonstrate the robustness of the non-local detection stage with respect to similar looking but different objects. Figure 5.24 shows a single disc embedded into other discs, whose radius of the inner ring is slightly larger than that of all others. Although this disc is very similar to the other discs, the multiple object detection through convex programming only returns this single object instance.

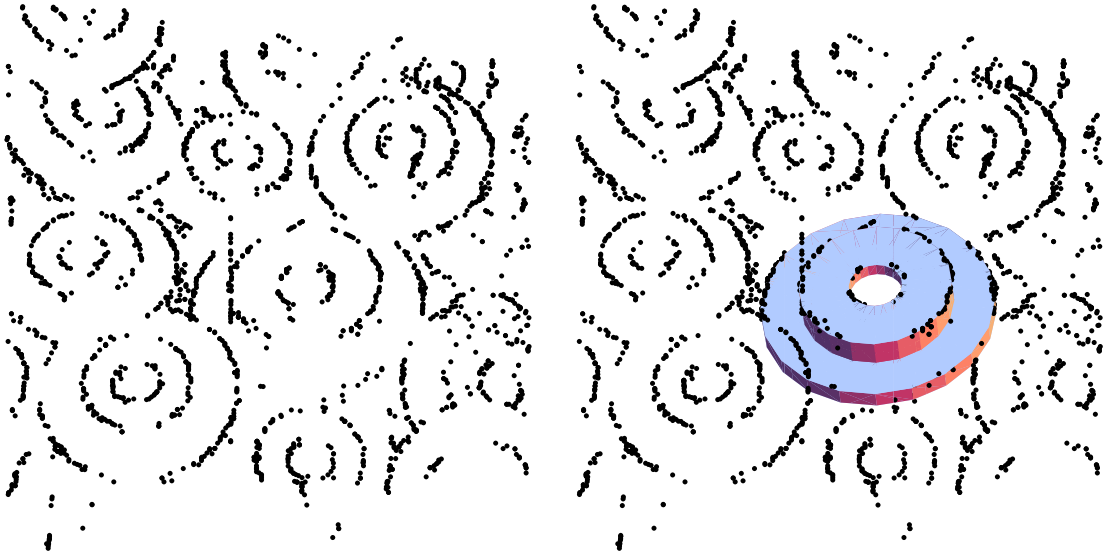


Figure 5.24: Robustness of object detection. A single disc that only slightly differs from all other discs (slightly larger inner ring radius) is reliable returned as single object instance by the first convex programming stage.

Chapter 6
Summary and Further Work

Summary

In this thesis, we presented a novel "initialization and refinement" approach for the problem of model-based detection and determination of the transformations of multiple objects for the real-world industrial bin-picking scenarios where the scene measurements are represented by noisy, unstructured, and sparse points.

The initialization stage of Chap. 4 is designed in terms of a global, large-scale, convex objective functional that accurately reflects the geometric constraints of the pose estimation problem. Additionally, this formulation enables to apply efficient preprocessing techniques derived from sufficient optimality conditions as well as the use of dedicated convex optimization algorithms. In general, it yields that the corresponding optimization problem can be solved efficiently while obtaining promising performance making the approach attractive for real-world applications with tight run-time constraints.

For a subsequent refinement stage (Chap. 2 and Chap. 3), we investigated a novel formulation of the Euclidean alignment problem based on continuous, kernel-based point set representations where obtaining the optimal registration amounts to minimize the Kullback-Leibler divergence between the model and the scene function. This provides convenient approximations of state-of-the-art approaches while resolving the problem of explicit determination of point-to-point correspondences. Moreover, it yields a smooth objective functional that allows to use higher order optimization techniques at the costs of lacking from closed form solutions.

To obtain the local optimizer of our new objective, we considered advanced optimization techniques that fully exploit the intrinsic properties of the underlying group of Euclidean transformations. To this end, we extended a Newton-like optimization approach to the smooth manifold of transformation and experimentally showed that it provides fast convergence to the local optimum while exhibiting sufficiently large regions of attraction.

Finally, we empirically demonstrated in Chap. 5 that the single steps of initialization and refinement complement each other using a collection of computer-generated and real-world measurements. This means that conflicting interpretations are resolved through non-local convex processing, followed by accurate non-convex local optimization based on sufficiently good initializations. Additionally we showed that the entire two-step approach provides the potential to meet the accuracy and run-time constraints of many real-world industrial scenarios.

Further Work

Despite of the generalization of the individual steps, there are major issues that have to be addressed in further work. On the one hand, we want to investigate run-time improvements by means of faster function evaluations for fine alignment. While there are

dozens of extensions to speed up state-of-the-art approaches such as using efficient data structures [11] or applying multi-resolution schemes [72], evaluating the log-exponential used in Chap. 2 and the corresponding derivative numerically is time-consuming in general.

Additionally, as demonstrated by experiments in Chap. 5, the convergence of Newton's algorithm depends on the smoothness of the underlying objective function. Thus, in order to guarantee convergence, in further work we want to estimate the size of the region of attraction [35], by deriving a criterion that ensures convergences to an optimal configuration beforehand. Moreover, it is a common strategy to introduce further step-size regularization techniques [133] to enlarge the region of attraction and to cope with poor approximations of the objective. Another approach is to consider trust region algorithms [2] that slightly complicate the Newton steps while guaranteeing stable convergence. Such approaches have to be addressed in further work as well.

Next, we hope to improve the theoretical basis of the preprocessing procedure by means of studying relaxations of the non-convex, binary sparse reconstruction problem to convex formulations for realistic setups such as the bin-picking scenario. Additionally, we want to improve the bounds for preprocessing, presented in Chap. 4, in order to fix variables of the optimal configuration in early processing steps.

Finally, we want to derive tight bounds for the postprocessing stage that enable us to infer the globally optimal configuration of a binary optimization problem based on a single or only few optimal configurations of certain continuous relaxation. Such bounds could prove to be useful in many other applications, too.

Bibliography

[1] Abramowitz, M., and Stegun, I. A. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover, 1964.

[2] Absil, P.-A., Mahony, R., and Sepulchre, R. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, 2008.

[3] Adler, R. L., Dedieu, J.-P., Margulies, J. Y., Martens, M., and Shub, M. Newton's Method on Riemannian Manifolds and a Geometric Model for the Human Spine. *IMA J. Numer. Anal. 22*, 3 (2002), 359 – 390.

[4] An, L. T. H., and Tao, P. D. The DC (Difference of Convex Functions) Programming and DCA Revisited with DC Models of Real World Nonconvex Optimization Problems. *Ann. Operations Research 133* (2005), 23–46.

[5] Ballard, D. H. Generalizing the Hough Transform to Detect Arbitrary Shapes. *Pattern Recogn. 13* (1981), 111–122.

[6] Basu, A., Harris, I. R., Hjort, N. L., and Jones, M. C. Robust and Efficient Estimation by Minimizing a Density Power Divergence. *Biometrika 85* (1998), 549–559.

[7] Beck, A., and Teboulle, M. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. *SIAM J. Imag. Sciences 2*, 1 (2009), 183–202.

[8] Belongie, S., Malik, J., and Puzicha, J. Shape Matching and Object Recognition Using Shape Contexts. *IEEE Trans. Pattern Anal. 24* (2002), 509–522.

[9] Benhimane, S., and Malis, E. A New Approach to Vision-Based Robot Control with Omni-Directional Cameras. In *IEEE Int. Conf. Robot. Autom.* (2006).

[10] Bennett, K., and Parrado-Hernández, E. The Interplay of Optimization and Machine Learning Research. *J. Mach. Learning Res. 7* (2006), 1265–1281.

[11] Bentley, J. L. Multidimensional Binary Search Trees Used for Associative Searching. *Commun. ACM 18*, 9 (1975), 509–517.

[12] Bertsimas, D., and Weismantel, R. *Optimization Over Integers*. Dynamic Ideas, 2005.

[13] Besl, P. J., and Jain, R. C. Three-Dimensional Object Recognition. *ACM Comput. Surv. 17*, 1 (1985), 75–145.

[14] Besl, P. J., and McKay, N. D. A Method for Registration of 3-D Shapes. *IEEE Trans. Pattern Anal. 14*, 2 (1992), 239–256.

[15] Bezdek, J. C., and Pal, S. K. *Fuzzy Models for Pattern Recognition*. IEEE Press, 1992.

[16] Birgin, E. G., Martínez, J. M., and Raydan, M. Nonmonotone Spectral Projected Gradient Methods on Convex Sets. *SIAM J. Optim. 10* (2000), 1196–1211.

[17] Bolles, R. C., and Horaud, P. 3DPO: A Three-Dimensional Part Orientation System. *Int. J. Robotics Research 5*, 3 (1986), 3–26.

[18] Boyd, S., and Vanderberghe, L. *Convex Optimization*. Cambridge Univ. Press, 2004.

[19] Boykov, Y., and Funka-Lea, G. Graph Cuts and Efficient N-D Image Segmentation. *Int. J. Comput. Vision 70*, 2 (2006), 109–131.

[20] Bradley, P. S., and Mangasarian, O. L. Feature Selection via Concave Minimization and Support Vector Machines. In *Int. Conf. on Mach. Learn.* (1998), pp. 82–90.

[21] Breitenreicher, D., and Schnörr. Model-Based Multiple Object Detection and Registration in Unstructured Range Data. *Int. J. Comput. Vision* (2010). submitted.

[22] Breitenreicher, D., and Schnörr, C. Intrinsic Second Order Geometric Optimization for Robust Point Set Registration without Explicit Correspondences. In *Int. Conf. Engergy Minimization Methods in Comput. Vision* (2009).

[23] Breitenreicher, D., and Schnörr, C. Robust 3D Object Registration Without Explicit Correspondence Using Geometric Integration. *Mach. Vision Appl.* (2009).

[24] Campbell, R. J., and Flynn, P. J. Eigenshapes for 3D Object Recognition. In *IEEE Int. Conf. Comput. Vision and Pattern Recogn.* (1999).

[25] Chan, T., Esedoglu, S., and Nikolova, M. Algorithms for Finding Global Minimizers of Image Segmentation and Denoising Models. *SIAM J. Appl. Math. 66*, 5 (2006), 1632–1648.

[26] Chen, S., Donoho, D., and Saunders, M. Atomic Decomposition by Basis Pursuit. *SIAM Review 43*, 1 (2001), 129–159.

[27] Chen, Y., and Medioni, G. Object Modelling by Registration of Multiple Range Images. *Image Vision Comput. 10*, 3 (1992), 145–155.

[28] Cheng, Y. Mean Shift, Mode Seeking, and Clustering. *IEEE Trans. Pattern Anal. 17* (1995), 790–799.

[29] Chin, R. T., and Dyer, C. R. Model-Based Recognition in Robot Vision. *ACM Comput. Surv. 18*, 1 (1986), 67–108.

[30] Chua, C. S., and Jarvis, R. 3D Free-Form Surface Registration and Object Recognition. *Int. J. Comput. Vision 17* (1996), 77–99.

[31] Chua, C. S., and Jarvis, R. Point Signatures: A New Representation for 3D Object Recognition. *Int. J. Comput. Vision 25*, 1 (1997), 63–85.

[32] Comaniciu, D., and Meer, P. Mean Shift: A Robust Approach Toward Feature Space Analysis. *IEEE Trans. Pattern Anal. 24*, 5 (2002), 603–619.

[33] Cover, T., and Thomas, J. *Elements of Information Theory*. John Wiley & Sons, Inc., 1991.

[34] Crouch, P. E., and Grossman, R. Numerical Integration of Ordinary Differential Equations on Manifolds. *J. of NonLinear Science 3* (1993), 1–33.

[35] Dedieu, J.-P., Priouret, P., and Malajovich, G. Newton's Method on Riemannian Manifolds: Covariant Alpha-Theory. *IMA J. Numer. Anal. 23* (2003), 295–419.

[36] Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum Likelihood from Incomplete Data via the EM Algorithm. *J. Royal Statist. Soc. Series B (Methodological) 1* (1977), 1–38.

[37] do Carmo, M. P. *Riemannian Geometry*. Birkhäuser Boston, 1992.

[38] Donoho, D. L. Compressed Sensing. *IEEE Trans. Inf. Theory 52* (2006), 1289–1306.

[39] Donoho, D. L., Elad, M., and Temlyakov, V. N. Stable Recovery of Sparse Overcomplete Representations in the Presence of Noise. *IEEE Trans. Inf. Theory 52* (2006), 6–18.

[40] Drummond, T., and Cipolla, R. Real-time Tracking of Complex Structures with On-Line Camera Calibration. *Image Vision Comput. 20*, 5-6 (2002), 427–433.

[41] Edelman, A., Arias, T. A., and Smith, S. T. The Geometry of Algorithms with Orthogonality Constraints. *SIAM J. Matrix Anal. Appl. 20* (1999), 303–353.

[42] Eggert, D. W., Lorusso, A., and Fisher, R. B. Estimating 3-D Rigid Body Transformations: A Comparison of Four Major Algorithms. *Mach. Vision Appl. 9*, 5-6 (1997), 272–290.

[43] Enqvist, O., Josephson, K., and Kahl, F. Optimal Correspondences from Pairwise Constraints. In *IEEE Intl. Conf. Comput. Vision* (2009).

[44] Fan, T.-J., Medioni, G., and Nevatia, R. Recognizing 3-D Objects Using Surface Descriptions. *IEEE Trans. Pattern Anal. 11*, 11 (1989), 1140–1157.

[45] Faugeras, O. D., and Hebert, M. The Representation, Recognition and Locating of 3-D Objects. *Int. J. Robot. Research 5* (1986), 27–52.

[46] Fischler, M. A., and Bolles, R. C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM 24*, 6 (1981), 381–395.

[47] Fitzgibbon, A. W. Robust Registration of 2D and 3D Point Sets. *Image Vision Comput. 21*, 13-14 (2003), 1145–1153.

[48] Frome, A., Huber, D., Kolluri, R., and Bülow, T. Recognizing Objects in Range Data using Regional Point Descriptors. In *Proc. Europ. Conf. Comp. Vision* (2004).

[49] Fukunaga, K., and Hostetler, L. The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition. *IEEE Tans. Inf. Theory 21*, 1 (1975), 32–40.

[50] Gabay, D. Minimizing a Differentiable Function over a Differential Manifold. *J. Opt. Theory Appl. 37* (1982), 117–219.

[51] Gal, R., and Cohen-Or, D. Salient Geometric Features for Partial Shape Matching and Similarity. *ACM Trans. Graph. 25* (2006), 130–150.

[52] Gall, J., Rosenhahn, B., and Seidel, H.-P. Clustered Stochastic Optimization for Object Recognition and Pose Estimation. In *Proc. Symp. Pattern Recogn.* (2007).

[53] Gelfand, N., Mitra, N. J., Guibas, L. J., and Pottmann, H. Robust Global Registration. In *Proc. Symp. Geom. Processing* (2005).

[54] Gold, S., Lu, C. P., Rangarajan, A., Pappu, S., and Mjolsness, E. New Algorithms for 2D and 3D Point Matching: Pose Estimation and Correspondence. In *Advances in Neural Inf. Process Systems* (1995).

[55] Goldberger, J., Gordon, S., and Greenspan, H. An Efficient Image Similarity Measure Based on Approximations of KL-Divergence Between Two Gaussian Mixtures. In *IEEE Int. Conf. Comput. Vision* (2003).

[56] Golub, G. H., and Van Loan, C. F. *Matrix Computations*. Johns Hopkins Univ. Press, 1996.

[57] Granger, S., and Pennec, X. Multi-Scale EM-ICP: A Fast and Robust Approach for Surface Registration. In *Proc. Europ. Conf. Comput. Vision* (2002).

[58] Greenspan, M. Geometric Probing of Dense Range Data. *IEEE Trans. Pattern Anal. 24* (2002), 495–508.

[59] Greenspan, M., and Boulanger, P. Efficient and Reliable Template Set Matching for 3D Object Recognition. In *Proc. Int. Conf. 3-D Digital Imaging and Modeling* (1999).

[60] Grippo, L., Lampariello, F., and Lucidi, S. A Nonmonotone Line Search Technique for Newton's Method. *SIAM J. Num. Anal. 23*, 4 (1986), 707–716.

[61] Gumhold, S., Wang, X., and MacLeod, R. Feature Extraction from Point Clouds. In *Proc. Int. Meshing Roundtable* (2001).

[62] Gwak, S., Kim, J., and Park, F. C. Numerical Optimization on the Euclidean Group with Applications to Camera Calibration. *IEEE Trans. Robot. Automation 19* (2003), 65–74.

[63] Hairer, E., Nørsett, S. P., and Wanner, G. *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer Verlag, 2008.

[64] Hartley, R. I., and Kahl, F. Global Optimization through Rotation Space Search. *Int. J. Comput. Vision 82*, 1 (2009), 64–79.

[65] Horn, B. K. P. Extended Gaussian Images. *Proc. IEEE 72* (1984), 1671–1686.

[66] Horn, B. K. P., Hilden, H. M., and Negahdaripour, S. Closed-Form Solution of Absolute Orientation using Orthonormal Matrices. *J. Optic. Soc. 5*, 7 (1988), 1127–1135.

[67] Huang, J., Ng, M., Rong, H., and Li, Z. Automated Variable Weighting in k-Means Type Clustering. *IEEE Trans. Pattern Anal. 27*, 5 (2005), 657–668.

[68] Hüper, K., and Trumpf, J. Newton-like Methods for Numerical Optimization on Manifolds. In *Conf. on Signals, Systems and Computers* (2004).

[69] Iserles, A., Munthe-Kaas, H. Z., Nørsett, S. P., and Zanna, A. Lie-Group Methods. *Acta Numerica 9* (2000), 215–365.

[70] Jian, B., and Vemuri, B. C. A Robust Algorithm for Point Set Registration Using Mixture of Gaussians. In *Proc. Int. Conf. Comput. Vision* (2005).

[71] Johnson, A. E., and Hebert, M. Using Spin Images for Efficient Object Recognition in Cluttered 3D Scenes. *IEEE Trans. Pattern Anal. 21*, 5 (1999), 433–449.

[72] Jost, T., and Hügli, H. A Multi-Resolution Scheme ICP Algorithm for Fast Shape Registration. In *Proc. Int. Symp. 3D Data Processing, Visualization and Transmission* (2002).

[73] Karp, R. M. Reducibility Among Combinatorial Problems. In *Complexity of Computer Computations*. Plenum Press, 1972.

[74] Katsoulas, D., Bastidas, C. C., and Kosmopoulos, D. Superquadric Segmentation in Range Images via Fusion of Region and Boundary Information. *IEEE Trans. Pattern Anal. 30* (2008), 781–795.

[75] Kazhdan, M., Funkhouser, T., and Rusinkiewicz, S. Rotation Invariant Spherical Harmonic Representation of 3D Shape Descriptors. In *Proc. Symp. Geometry Processing* (2003).

[76] Krebs, B., Sieverding, P., and Korn, B. A Fuzzy ICP Algorithm for 3D Free-Form Object Recognition. In *Proc.Int. Conf. Pattern Recogn.* (1996).

[77] Krishnan, S., Lee, P. Y., Moore, J. B., and Venkatasubramanian, S. Optimisation-on-a-Manifold for Global Registration of Multiple 3D Point Sets. *Int. J. Intell. Syst. Technol. Appl. 3* (2007), 319 – 340.

[78] Kühnel, W. *Differentialgeometrie*. Vieweg & Teubner Verlag, 2008.

[79] Lee, S., Choi, M., Kim, H., and Park, F. Geometric Direct Search Algorithms for Image Registration. *IEEE Trans. Image Processing 16* (2007), 2215–2224.

[80] Lee, Y. P., and Moore, J. B. Pose Estimation via Gauss-Newton-on-Manifold. In *Proc. Int. Symp. Mathematical Theory of Networks and Systems* (2004).

[81] Li, H., and Hartley, R. The 3D-3D Registration Problem Revisited. In *Proc. Int. Conf. Comput. Vision* (2007).

[82] Matsushima, Y. *Differentiable Manifolds*. Marcel Dekker, Inc. New York, 1972.

[83] Miller, A. J. *Subset Selection in Regression*. Chapman and Hall, 2002.

[84] Mitra, N. J., Gelfand, N., Pottmann, H., and Guibas, L. Registration of Point Cloud Data from a Geometric Optimization Perspective. In *Proc. Symp. Geometric Processing* (2004).

[85] Munthe-Kaas, H. Runge-Kutta Methods on Lie Groups. *BIT 38* (1998), 92–111.

[86] Murray, R. M., Sastry, S. S., and Zexiang, L. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Inc., 1994.

[87] Natarajan, B. K. Sparse Approximate Solutions to Linear Systems. *SIAM J. Comput. 24* (1995), 227–234.

[88] Nesterov, Y. Smooth Minimization of Non-Smooth Functions. *Math. Program. 103*, 1 (2005), 127–152.

[89] Nesterov, Y. Gradient Methods for Minimizing Composite Objective Function. Tech. Rep. 2007076, Center for Operations Research and Econometrics, Univ. Catholique de Louvain, 2007.

[90] Olsson, C., Kahl, F., and Oskarsson, M. Branch-and-Bound Methods for Euclidean Registration Problems. *IEEE Trans. Pattern Anal. 31*, 5 (2009), 783–794.

[91] Owren, B., and Welfert, B. The Newton Iteration on Lie Groups. *BIT 40* (2000), 121–145.

[92] Park, J., and Chung, W.-K. Geometric Integration on Euclidean Group with Application to Articulated Multibody Systems. *IEEE Trans.Robot. 21* (2005), 250–263.

[93] Petra, S., and Schnörr, C. TomoPIV meets Compressed Sensing. *Pure Math. Appl. 20* (2010). in press.

[94] Pottmann, H., Huang, Q.-X., Yang, Y.-L., and Hu, S.-M. Geometry and Convergence Analysis of Algorithms for Registration of 3D Shapes. *Int. J. Computer Vision 67*, 3 (2006), 277–296.

[95] Quek, F., Jain, R., and Weymouth, T. E. An Abstraction-Based Approach to 3-D Pose Determination from Range Images. *IEEE Trans. Pattern Anal. 15* (1993), 722–736.

[96] Rangarajan, A., Chui, H., and Bookstein, F. L. The Softassign Procrustes Matching Algorithm. In *Proc. Int. Conf. Inf. Process. Med. Imaging* (1997).

[97] Raydan, M. The Barzilai and Borwein Gradient Method for the Large Scale Unconstrained Minimization Problem. *SIAM J. Optim. 7*, 1 (1997), 26–33.

[98] Robert, C. P., and Casella, G. *Monte Carlo Statistical Methods*. Springer Texts in Statistics, 1999.

[99] Rockafellar, R., and Wets, R.-B. *Variational Analysis*. Springer, 1998.

[100] Rockafellar, R. T. *Convex Analysis*. Princeton Univ. Press, 1970.

[101] Rodgers, J., Anguelov, D., Pang, H.-C., and Koller, D. Object Pose Detection in Range Scan Data. In *Proc. Conf. Comput. Vision Pattern Recogn.* (2006).

[102] Rose, K. *Deterministic Annealing for Clustering, Compression, Classification, Regression and Speech Recognition*. IEEE Press, 2001, ch. 5.

[103] Rusinkiewicz, S., Hall-Holt, O., and Levoy, M. Real-Time 3D Model Acquisition. *ACM Trans. Graph. 21*, 3 (2002), 438–446.

[104] Rusinkiewicz, S., and Levoy, M. Efficient Variants of the ICP Algorithm. In *Proc. Int. Conf. 3-D Digital Imaging and Modeling* (2001).

[105] Salvi, J., Matabosch, C., Fofi, D., and Forest, J. A Review of Recent Range Image Registration Methods with Accuracy Evaluation. *Image Vision Comput. 25* (2007), 578–596.

[106] Sandhu, R., Dambreville, S., and Tannenbaum, A. Particle Filtering for Registration of 2D and 3D Point Sets with Stochastic Dynamics. In *Int. Conf. Comput. Vision Pattern Recogn.* (2008).

[107] Schellewald, C., Roth, S., and Schnörr, C. Evaluation of a Convex Relaxation to a Quadratic Assignment Matching Approach for Relational Object Views. *Image Vision Comput. 25* (2007), 1301–1314.

[108] Schwarz, H. R., and Köckler, N. *Numerische Mathematik*, 6 ed. B. G. Teubner Verlag, 2006.

[109] Shang, L., and Greenspan, M. Pose Determination By Potential Well Space Embedding. In *Proc. Int. Conf. 3-D Digital Imaging and Modeling* (2007).

[110] Shang, L., Jasiobedzki, P., and Greenspan, M. Model-Based Tracking by Classification in a Tiny Discrete Pose Space. *IEEE Trans. Pattern Anal. 29* (2007), 976–989.

[111] Shi, Q., Xi, N., Chen, Y., and Sheng, W. Registration of Point Clouds for 3D Shape Inspection. In *Int. Conf. Intelligent Robots and Systems* (2006).

[112] Smith, S. T. Optimization Techniques on Riemannian Manifolds. *Fields Inst. Commun. Hamiltonian and Gradient Flows, Alg. Control 3* (1994), 113–136.

[113] Spivak, M. *A Comprehensive Introduction to Differential Geometry*, vol. 2. Publish or Perish, 1990.

[114] Stein, F., and Medioni, G. Structural Indexing: Efficient 3-D Object Recognition. *IEEE Trans. Pattern Anal. 14*, 2 (1992), 125–145.

[115] Subbarao, R., and Meer, P. Nonlinear Mean Shift over Riemannian Manifolds. *Int. J. Comput. Vision 84* (2009), 1–20.

[116] Taylor, C. J., and Kriegman, D. J. Minimization on the Lie Group SO(3) and Related Manifolds. Tech. Rep. 9405, Center for Systems Science, Dept. of Elect. Eng., Yale Univ., 1994.

[117] Teboulle, M. A Unified Continuous Optimization Framework for Center-Based Clustering Methods. *J. Mach. Learn. Res. 8* (2007), 65–102.

[118] Torr, P. H. S., and Davidson, C. IMPSAC: Synthesis of Importance Sampling and Random Sample Consensus. In *Proc. Europ. Conf. Comput. Vision* (2000).

[119] Tropp, J. A. Greed is Good: Algorithmic Results for Sparse Approximation. *IEEE Trans. Inf. Theory 50* (2004), 2231–2242.

[120] Tropp, J. A. Just Relax: Convex Programming Methods for Identifying Sparse Signals in Noise. *IEEE Trans. Inf. Theory 52* (2006), 1030–1051.

[121] Tsin, Y., and Kanade, T. A Correlation-Based Approach to Robust Point Set Registration. In *Proc. Europ. Conf. Comput. Vision* (2004).

[122] Tucker, T. M., and Kurfess, T. R. Newton Methods for Parametric Surface Registration. Part I. Theory. *Computer-Aided Design 35* (2003), 107–114.

[123] Turk, G., and Mullins, B. Large Geometric Models Archive. http://www-static.cc.gatech.edu/projects/large_models/.

[124] Varadarajan, V. S. *Lie Groups, Lie Algebras, and their Representations.* Prentice-Hall, Inc., 1974.

[125] Žefran, M., Kumar, V., and Croke, C. Metrics and Connections for Rigid-Body Kinematics. *Int. J. Robot. Res. 18* (1999), 1–16.

[126] Wainwright, M., and Jordan, M. Graphical Models, Exponential Families, and Variational Inference. *Found. Trends Mach. Learning 1* (2008), 1–305.

[127] Wang, C., Liu, Q., and Yang, X. Convergence Properties of Nonmonotone Spectral Projected Gradient Methods. *J. Comput. Appl. Math. 182*, 1 (2005), 51–66.

[128] Wang, F., Vemuri, B. C., Rangarajan, A., Schmalfuss, I. M., and Eisenschenk, S. J. Simultaneous Nonrigid Registration of Multiple Point Sets and Atlas Construction. In *Proc. Europ. Conf. Comput. Vision* (2006).

[129] Wiedemann, C., Ulrich, M., and Steger, C. Recognition and Tracking of 3D Objects. In *Proc. Symp. Pattern Recogn.* (2008).

[130] Wolfson, H. J., and Rigoutsos, I. Geometric Hashing: An Overview. *IEEE Comput. Sci. Eng. 4* (1997), 10–21.

[131] Wolsey, L. A., and Nemhauser, G. L. *Integer and Combinatorial Optimization.* Wiley-Interscience Series in Discrete Mathematics and Optimization, 1999.

[132] Wright, J., Yang, A. Y., Ganesh, A., Sastry, S. S., and Ma, Y. Robust Face Recognition via Sparse Representation. *IEEE Trans. Pattern Anal. 31*, 2 (2009), 210–227.

[133] Yang, Y. Globally Convergent Optimization Algorithms on Riemannian Manifolds: Uniform Framework for Unconstrained and Constrained Optimization. *J. Optim. Theory Appl. 132* (2007), 245–265.

[134] Zhang, Z. Iterative Point Matching for Registration of Free-Form Curves and Surfaces. *Int. J. Comput. Vision 13* (1994), 119–152.

[135] Zhu, L., Barhak, J., Shrivatsan, V., and Katz, R. Efficient Registration for Precision Inspection of Free-Form Surfaces. *Int. J. Adv. Manuf. Technol. 32* (2007), 505–515.