Fitting model parameters in conformal geometric algebra to Euclidean observation data
Cibura, C.

Link to publication

Citation for published version (APA):
Cibura, C. (2012). Fitting model parameters in conformal geometric algebra to Euclidean observation data
FITTING MODEL PARAMETERS IN CONFORMAL GEOMETRIC ALGEBRA TO EUCLIDEAN OBSERVATION DATA

Academisch Proefschrift

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. D.C. van den Boom
ten overstaan van een door het college voor promoties
ingestelde commissie,
in het openbaar te verdedigen in de Agnietenkapel
op woensdag 1 februari 2012, te 14:00 uur

door

Carsten Cibura

geboren te Berlijn, Duitsland
Promotiecommissie:
Promotor: Prof. dr. ir. F.C.A. Groen
Copromotor: dr. ir. L. Dorst
Overige Leden: Prof. dr. J.A. Bergstra
Prof. dr. M. de Berg
Prof. dr.-ing. B. Rosenhahn
dr. H.G.J. Pijls
dr. J. Lasenby

This research was supported by the Netherlands Organization for Scientific Research (NWO) under project number 612.065.517

This work was carried out in the ASCI graduate school.
ASCI dissertation series number 249.

Copyright © 2011 Carsten Cibura
ISBN: 978-94-6182-072-3

Typeset by the author with the \LaTeX documentation system

Intelligent Systems Lab Amsterdam
Faculteit der Natuurwetenschappen Wiskunde en Informatica
Universiteit van Amsterdam
Science Park 904, 1098 XH Amsterdam
Abstract

Fitting geometric models to observed data is a fundamental problem that occurs frequently in a variety of applications such as 3D reconstruction, registration, motion capturing, object recognition, augmented reality and many more. The models that are fit and the available observation data vary greatly. Common models include geometric primitives such as planes, spheres or ellipsoids, geometric transformations such as rigid body motions or Euclidean similarities and kinematic models of articulated structures like the human body (or approximations thereof).

Classically, geometric objects are represented by vectors, a collection of numbers accompanied by an explicit statement of what those numbers represent: a point in space, a direction, a plane etc. In geometric algebra geometric entities can be represented directly and become automatically identifiable by their algebraic properties. Of particular interest in this thesis is the conformal model of geometric algebra (CGA) whose fundamental building blocks represent Euclidean hyperspheres and conformal transformations of Euclidean space. The basic mathematical details are recounted in chapter 2 of this thesis.

The high dimensionality of a geometric algebra, the non-commutativity of the geometric product and the complexity of the interaction between CGA elements sometimes make it difficult to solve equations — exactly or approximately — for certain terms. In the course of solving the presented problems, we develop techniques for that and establish conditions under which those techniques apply.

In chapter 3 we use CGA to infer a conformal transformation in Euclidean space from exact correspondence data. To achieve this goal we first reduce the problem to finding a Euclidean transformation between correspondence data. Classically, conformal transformations are represented by complex valued matrices and restricted to the complex 2D plane. The representation of these transformations has been extended to geometric algebra valued Vahlen matrices. This extends their applicability to higher-dimensional spaces and also enables us to perform the proposed problem reduction.

In chapter 4 we fit a number of component 3D screw motions to an observed composite 3D screw motion. Such composite motions are, for example, exhibited by limbs of a kinematic chain. We employ CGA to formalize the problem statement and uncover the underlying structure that enables us to find a unique decomposition of the composite motion. In order
to find the parameters of the CGA objects that represent screw motions we employ matrix algebra. This is useful in two ways. Firstly, it enables us to formulate a number of linear equations which can be solved for the components of CGA objects. Secondly, where an exact solution of those equations is not possible, established matrix algebra methods like singular value decomposition can yield approximate results.

Finally, in chapter 5 we consider the estimation of Euclidean hyperspheres from the viewpoint of Bayesian reasoning. An advantage of the Bayesian framework — which has proven useful for parameter estimation in many different settings — is that it forces the creator of an estimator to make explicit all parameters, prior knowledge (or absence of it) about them and all assumptions that are relevant to the problem at hand. In our case we derive a closed form formula for the likelihood of a hypersphere in arbitrary dimensions given point observations degraded by isotropic Gaussian noise. We analyze the properties of a maximum likelihood estimator based on that formula and compare it to a least squares estimator for the same problem.

For the problems tackled in chapter 4 and chapter 5 we have run experiments with simulated data. The numerical results of those can be found in appendix A and appendix B, respectively.

By solving the above problems we gain new insights into the practical handling of geometric algebra. We prove its usefulness for processing geometric data and deepen our understanding of its relation to classical branches of mathematics. The latter is especially useful in order to exploit an existing body of research in these related fields, while adding to it the conciseness, intuitiveness and calculational power of geometric algebra. We reflect on some of these aspects in chapter 6.
Samenvatting

Het aanpassen van geometrische modellen aan geobserveerde data is een fundamenteel probleem dat vaak optreedt in uiteenlopende applicaties zoals 3D reconstructie, registratie, motion capturing, object herkenning, augmented reality en veel meer. De modellen die worden gebruikt en de beschikbare observaties variëren sterk. Gebruikelijke modellen omvatten geometrische primitieven zoals vlakken, kogels of ellipsoïden, geometrische transformaties zoals bewegingen van starre lichamen, Euclidische gelijkvormige transformaties en kinematische modellen van lichamen met gewrichten zoals het menselijke lichaam (of benaderingen daarvan).

De klassieke manier om geometrische objecten te representeeren is door middel van vectoren, een verzameling van getallen met een een expliciete beschrijving van wat deze getallen repre- senteeren: een punt in de ruimte, een richting, een vlak etc. In geometrische algebra kunnen geometrische objecten echter direct gerepresenteerd worden en ze worden automatisch identificeerbaar door hun algebraïsche eigenschappen. Van specifiek belang in dit proefschrift is het conforme model van geometrische algebra (CGA). De fundamentele bouwstenen van CGA representeren Euclidische hyperkogels en conforme transformaties van de Euclidische ruimte. De wezenlijke wiskundige details worden samegevat in hoofdstuk 2 van dit proefschrift.

De hoge dimensionaliteit van een geometrische algebra, de niet-commutativiteit van het geometrische product en de complexiteit van de interactie tussen CGA-elementen maken het soms moeilijk om vergelijkingen op te lossen — precies of bij benadering — voor bepaalde termen. Gedurende het oplossen van de gestelde problemen ontwikkelen we daar technieken voor en stellen we vast onder welke voorwaarden deze technieken toepasbaar zijn.

In hoofdstuk 3 gebruiken we CGA om een conforme transformatie in de Euclidische ruimte van exacte correspondenties af te leiden. Om dit te bereiken reduceren we het probleem eerst tot het vinden van een Euclidische transformatie tussen de correspondenties. Gewoonlijk zijn conforme transformaties geregistribueerd door complexe matrices en beperkt tot het complexe 2D-vlak. De representatie van deze transformaties wordt uitgebreid tot Vahlen matrices met elementen van geometrische algebra. Dit vergroot hun toepasbaarheid op hoger-dimensionale ruimten en stelt ons in staat om de voorgestelde reductie van het probleem uit te voeren.

In hoofdstuk 4 wordt een geobserveerde 3D schroefbeweging ontleed in een aantal elementaire 3D schroefbewegingen. Dergelijke samengestelde bewegingen worden bijvoorbeeld vertoond door ledematen die een kinematische keten vormen. Wij gebruiken CGA om de probleem-
stelling te formaliseren en ontdekken de onderliggende structuur die het mogelijk maakt de samengestelde beweging uniek op te delen. Om de parameters te vinden van de CGA objecten die schroefbewegingen representeren gebruiken we matrix-algebra. Dit is nuttig om twee redenen. Ten eerste stelt het ons in staat om een aantal lineaire vergelijkingen te formuleren die opgelost kunnen worden voor de componenten van deze CGA objecten. Ten tweede, waar een exacte oplossing van deze vergelijkingen niet mogelijk is, kunnen matrix-algebra methoden zoals singular value decomposition een benadering opleveren.

Tenslotte beschouwen we in hoofdstuk 5 het schatten van Euclidische hyperkogels vanuit het standpunt van Bayesiaans redeneren. Een voordeel van de Bayesiaanse benadering — die nuttig is gebleken voor het schatten van parameters in veel verschillende scenario’s — is dat het de ontwerper van een schatter dwingt om alle parameters, voorkennis (of het ontbreken ervan) over hen en alle veronderstellingen die relevant zijn voor een probleem expliciet te maken. In ons geval leiden we een formule in gesloten vorm af voor de waarschijnlijkheid van een hyperkogel van willekeurige dimensionaliteit, gegeven geobserveerde punten op de hyperkogel die door isotrope Gaussische ruis verstoord zijn. We analyseren de eigenschappen van een maximum likelihood schatter op basis van deze formule en vergelijken het met een kleinste kwadraten schatter voor hetzelfde probleem.

Voor de problemen die in hoofdstuk 4 en hoofdstuk 5 worden aangepakt hebben we experimenten met gesimuleerde data uitgevoerd. De numerieke resultaten van deze experimenten bevinden zich respectievelijk in appendix A en appendix B.

Door het oplossen van de bovengenoemde problemen hebben we nieuwe inzichten gekregen in de praktische toepassing van geometrische algebra. We bewijzen het nut ervan voor de verwerking van geometrische gegevens en verdiepen ons begrip van de relatie tot klassieke takken van de wiskunde. Dit laatste is vooral van belang om van het bestaande corpus van onderzoek naar deze aanverwante gebieden te profiteren, terwijl we hier de beknoptheid, intuïtie en rekenkracht van geometrische algebra aan toevoegen. We reflecteren op enige van deze aspecten in hoofdstuk 6.
Acknowledgements

I wish to thank my supervisor, Leo Dorst, who has played many roles: mentor, sparring partner, companion. Without his guidance I could not have done this. And his gentle grasp of the reins has kept me wanting to do it.

I thank my family and my friends who have proven constants in troubled times.

And I extend my special gratitude to Thanassis and Isaac with whom I had the pleasure of seeing many a free variable in the world outside of the ivory tower.
Contents

List of Figures xiii

List of Tables xv

1. Introduction 1
   1.1. Geometric Models and Linear Algebra 2
   1.2. Geometric Algebra 3
   1.3. Challenges 5
   1.4. Focus 5
   1.5. Overview 6

2. Introduction to Geometric Algebra 9
   2.1. Inner, Outer and Geometric Product 10
   2.2. Grades, Bases and Frames 13
   2.3. Orthogonal Transformations in Geometric Algebra 15
   2.4. The Conformal Model of Geometric Algebra (CGA) 19
   2.5. Relation to Linear Algebra 24
   2.6. Relation to Differential Geometry 25
   2.7. Summary 27

3. Determining Conformal Transformations from Minimal Correspondence Data 29
   3.1. Introduction 30
   3.2. Representation by Extended Vahlen Matrices 31
   3.3. Exact Conformal Transformations and Minimal Data 36
      3.3.1. Related Methods 37
      3.3.2. Input Data Requirements 38
      3.3.3. Solution 40
3.4. Generalization and Limitations ........................................... 44
  3.4.1. Even vs. Odd Versors ................................................. 44
  3.4.2. Minimal Data from Exact Point Correspondences ................. 44
  3.4.3. Special Case: Rigid Body Motions .................................. 47
  3.4.4. Non-Exact Correspondence Data ..................................... 48
3.5. Conclusion ................................................................. 49

4. Finding Articulated Structures from Motion of the End-Frame .......... 51
  4.1. Introduction ..................................................................... 52
    4.1.1. Related Work .......................................................... 52
    4.1.2. Our Approach ......................................................... 54
  4.2. Preliminaries ............................................................... 56
    4.2.1. Notation and Glossary ............................................... 56
    4.2.2. Transformations of Rigid Body Motion Versors .................. 61
  4.3. Solution to Component Versors ......................................... 66
    4.3.1. Zero Elemental Motions .............................................. 67
    4.3.2. One Elemental Motion ............................................... 67
    4.3.3. Two Elemental Motions .............................................. 67
    4.3.4. Three Elemental Motions ............................................ 69
    4.3.5. More Than Three Elemental Motions ................................. 75
  4.4. Examining Assumptions and Degeneracies ............................... 76
    4.4.1. Theoretical Remarks .................................................. 76
    4.4.2. Empirical Evaluation .................................................. 78
  4.5. Conclusion ................................................................. 79

5. A Bayesian Approach to Hypersphere Estimation ......................... 81
  5.1. Motivation ..................................................................... 82
  5.2. Background ............................................................... 82
    5.2.1. Structural Model vs. Functional Model ............................. 83
    5.2.2. Cartesian Model vs. Radial Model .................................. 84
    5.2.3. Maximum Likelihood vs. Least Squares ............................. 85
    5.2.4. Consistency of MLE in General and in Particular ................. 86
    5.2.5. Efficiency of the MLE .................................................. 86
  5.3. A Bayesian Approach ...................................................... 87
  5.4. Evaluation of the Closed Form Likelihood ............................. 90
  5.5. Results ................................................................... 92
  5.6. Conclusion ................................................................. 98

6. Discussion and Conclusion ................................................... 101
  6.1. Conformal Transformations and Extended Vahlen Matrices .......... 102
  6.2. Decomposing Composite Motions ........................................ 104
  6.3. Bayesian Hypersphere Estimation ........................................ 105
6.4. Future Research ........................................ 106

A. Simulation Experiments
   on Motion Decomposition .................................. 111
   A.1. Issues with Uncertainty ................................ 111
   A.2. Experimental Setup ..................................... 113
   A.3. Experimental Results .................................... 115
      A.3.1. Random Configuration, 3 Links ...................... 117
      A.3.2. Cube Configuration, 3 Links ......................... 121
      A.3.3. Random Configuration, 2 Links ...................... 125
      A.3.4. Cube Configuration, 2 Links ........................ 129
      A.3.5. Cube Configuration, 3 Links, Accelerated Third Link 133
      A.3.6. Cube Configuration, 2 Links, Accelerated Second Link 136

B. Quantitative Results
   of Bayesian Hypersphere Estimation ....................... 139

C. Background
   on the Hypergeometric Function .......................... 147

Bibliography .................................................. 149
List of Figures

2.1. A 2-blade $a \wedge b$ and distributivity over addition .................. 11
2.2. Reflection at a vector .................................................. 15
2.3. Purely Euclidean rotation about the origin as a versor ................. 18
2.4. A conformal circle by the outer product of three conformal points .......... 23
2.5. Inversion in a hypersphere ............................................. 23

3.1. Orthogonal transformation as a sequence of reflections in mid-planes .... 37
3.2. Set of minimal data: a localized frame and an additional point .............. 39
3.3. Determining minimal data from point correspondences .................... 45
3.4. Scaling with respect to Cartesian transfer frame .......................... 46

4.1. Articulated structure with three links in 3D .............................. 55
4.2. True motion vs. instantaneous motion between two constellations in 2D .... 56
4.3. Progression of meta-motions at different meta-levels ....................... 59
4.4. Parameters of a screw motion in 3D .................................... 62
4.5. Chords between Euclidean vectors on a screw axis and rotated versions .... 64
4.6. 1-parameter family of rotations between two Euclidean 3D bivectors ........ 68
4.7. Mid-plane between two 3D Euclidean bivectors .......................... 71
4.8. A geometric degeneracy where a translation is partially subsumed by a preceding screw motion .................................................. 78

5.1. Bayesian network showing dependencies in parameter estimation .......... 83
5.2. Gaussian Cartesian functional model for circle estimation .................. 84
5.3. Geometric rationale for closed form likelihood of circle parameters ......... 90
5.4. Sphere imaged under (near) parallel projection ............................ 94
5.5. Noisy observations of points on a sphere with different spread parameters .. 95
5.6. MLE and non-linear least squares estimates of spheres given noisy points ... 96
5.7. Estimators’ bias as a function of the spread parameter ..................... 97
5.8. Determinant of estimators’ covariance matrices ............................. 98
5.9. Ratio of determinants of estimators’ covariance matrices ................... 99
List of Tables

B.1. sample points with spread $q = 0.1$ ......................................................... 140
B.2. sample points with spread $q = 0.2$ ......................................................... 141
B.3. sample points with spread $q = 0.5$ ......................................................... 142
B.4. sample points with spread $q = 1.0$ ......................................................... 143
B.5. sample points uniformly distributed ......................................................... 144
B.6. sample points cosine distributed over the half-sphere .............................. 145
Introduction

While the human mind has evolved to be very efficient at understanding its environment, this task poses a great challenge in computer vision. Data acquired by sensors may be noisy or ambiguous, or stem from sources inaccessible to human senses. The models chosen to represent the data may conform more or less to reality and strike a more or less favorable trade-off between computational effort and conformity with geometric intuition.

The field of geometric data processing is concerned with automatic computation with data representing geometric circumstances and often describing physical reality. Increasingly affordable and ubiquitous sensors — especially video cameras — are recording big parts of our world. Whether in robotics, motion capturing, 3D reconstruction or augmented reality, fitting the parameters of geometric models to observations is a crucial step in turning the raw data into understandable objects of computation. For example, if an articulated structure in motion is imaged by a video camera, the resulting image sequence or the constituent pixels may be regarded as raw geometric data. Often the interest lies in recovering the location of the joints or in describing the motion that each of the links of the structure performs over time.\(^1\)

Linear algebra presents a useful way of representing and calculating with geometric data. It offers many benefits in terms of computability. A vast body of research has developed which supports the user of classical linear algebra with an arsenal of matrix decomposition algorithms and techniques for manipulating and solving systems of linear equations, exactly

\(^1\)The NWO project “Detection of Articulated Structures in Image Sequences” (DASIS) is concerned with just this type of problem and provided funding for this thesis.
or approximately. However, as we will point out below, using classical linear algebra for representing geometric entities has its shortcomings. A number of these shortcomings is tackled by using geometric algebra, an intuitive mathematical framework, which is moreover efficiently implementable in many common programming languages \cite{Fon07}. Geometric algebra is, however, somewhat more difficult to handle algebraically and therefore, when it comes to fitting geometric models to data, it offers some challenges which this thesis addresses.

1.1. Geometric Models and Linear Algebra

Classical linear algebra is an algebra of linear transformations represented by matrices acting on other matrices and vectors. Vectors, as the target of transformations, are therefore used to represent geometric objects, such as points, planes, lines, directions etc.

This approach brings about some problematic consequences. Firstly, it immediately introduces a coordinate system. By using a set of coefficients to describe the location of a point or a direction in space, one implicitly assumes a vector basis on which the coefficients are to be interpreted. Matrices acting on this coefficient representation also implicitly assume a basis and therefore work only on objects which are represented on that specific basis. Changing that basis usually involves coordinate transformations, which again only translate an equation from one particular basis to another. A general, coordinate free formulation of concepts is not possible in classical linear algebra.

Secondly, if one only has a set of numbers representing geometric objects, it is necessary to explicitly keep track of the type of object thus represented. In classical linear algebra, for example, points, planes and directions are indiscriminately represented by vectors, which are mathematically indistinguishable from each other. What makes matters worse is that different geometric objects may transform differently. For example, in homogeneous coordinate representation, vectors representing directions may be added, while adding vectors that represent points may not yield another valid point representation.

Finally, the parametrization of linear transformations by matrices is usually redundant. For instance, a 3D rotation about the origin can be represented by a $3 \times 3$-matrix with nine entries, when the transformation only has three degrees of freedom. This over-parametrization is compensated for by complicated algebraic constraints, e.g. requiring the matrix determinant to be equal to 1. In calculations which have the transformation as a result, the matrix entries are often regarded as independent parameters and the internal constraints have to be enforced explicitly. This may not be the most efficient approach, and may lead to imprecisions in the obtained transformation.

Some of these problems are alleviated by employing specialized representations. For instance, lines in 3D can be represented by six-dimensional vectors of Plücker coordinates, which sets them apart from points and directions, which can be represented by three- or — using homogeneous coordinates — four-dimensional vectors. A quadric can be represented by the coefficients of a quadratic equation in the coordinates describing it. The coefficients can be arranged into a suitable matrix that captures their transformation under geometric transforma-
tions of the quadric. Other problems are remedied by using different techniques from different mathematical branches — such as using unit quaternions for the representation of rotations. But this practice quickly leads to a large number of case distinctions and specialized tools. Understanding the finer points of those often requires the study of their respective mathematical field.

On the other hand, classical linear algebra offers a number of practical advantages. Linear models are easy to compute with and serve as a first order approximation to more complicated problems. The matrix representation yields to a number of well-known techniques for analyzing, constraining or solving for the represented transformations or the argument upon which the transformation acts. For example, approaches based on the singular value decomposition (SVD) yield exact solutions to systems of linear equations — or a least squares approximation where such exact solutions do not exist. Invariant spaces of linear transformations can be found by available algorithms which recover the eigenvectors of a matrix. The wide research field of (applied) linear algebra is under active development and new applications are found constantly. Efficient implementations in various programming languages are publicly available and updated frequently while specialized hardware facilitates matrix computations.

So there appears to be a gap between the suitability for representing geometric data on the one hand and the convenience of implementation and computation with this data on the other hand. In this thesis we will use geometric algebra to fill this gap and examine to which extent it holds up to the requirements on both ends.

1.2. Geometric Algebra

Hermann Günther Grassmann was the first to develop geometric algebra in a systematic way in 1844. He introduced methods to encode geometrical information of a given space such as length, area and volume. In 1878 William Kingdon Clifford examined Grassmann’s algebraic system with special attention to the quaternions of William Rowan Hamilton. He defined the geometric product adding to Grassmann’s ideas the description of certain transformations, such as rotations, and revealing quaternions to be a part of Grassmann’s framework. Rudolf Lipschitz generalized Clifford’s interpretation of quaternions to describe the geometry of rotations in \( n \) dimensions. [Cib07] The emerging associative algebra over a metric vector space has become known as Clifford algebra. Often, in applied literature the terms Clifford algebra and geometric algebra are used interchangeably. The latter emphasizes the use of the algebraic entities for representing geometric objects and transformations.

In recent years the development of geometric algebra has been spearheaded by David Hestenes (who holds a patent [HRL05] on the conformal model of geometric algebra) and Garret Sobczyk [HS84] as well as research groups in Cambridge (e.g. [LD03]) and Amsterdam (e.g. [DFM07]). As more research groups pick up the topic, geometric algebra has entered a stage of free experimentation, with different models of geometric algebra being tailored to specific problems.

One aspect that makes geometric algebra uniquely suited for geometric data processing
is the fact that it unifies different branches of mathematics in a single framework. It brings together such mathematical fields as linear algebra, quaternions, differential geometry, calculus and several representational models known from classical linear algebra. Its elements are multivectors, high-dimensional multi-linear combinations of vectors. In this thesis we focus on the conformal model of geometric algebra (CGA). It is a particular Clifford algebra, constructed over an \( n + 2 \)-dimensional vector space equipped with a particular metric (more precisely, a Minkowski metric) and used to represent geometric objects in \( n \)-dimensional Euclidean space. CGA vectors represent \textit{Euclidean hyperspheres}, with planes and points forming special cases of \textit{spheres with infinite and zero radius}, respectively.

In CGA, transformations are represented by \textit{versors}. These parametrize translations, rotations, uniform scaling and general conformal transformations in a Euclidean space \textit{minimally}. That is, the number of parameters agrees with the number of degrees of freedom for these transformations. Moreover, versors act on geometric objects in a structure preserving manner. Regardless of which object is transformed, the algebraic operation is always the same but the geometrical result is precisely adapted to the type of the transformed object. No longer is there a need for specialized transformations that have to be manually adapted to the object they act upon.

Furthermore, all geometric objects are \textit{immediately identifiable by their algebraic properties}. Explicit typification is not required. An additional advantage of geometric algebra is that it allows a coordinate free formulation of many concepts. An explicit basis is introduced only when numerical computation is required. Often, however, calculations carry quite far without introducing coordinates. We will revisit these notions in chapter 2, where we give a more thorough introduction to geometric algebra based on [HS84, LD03, DFM07].

CGA goes beyond purely Euclidean geometrical data processing by adding “book-keeping” dimensions that enhance calculations in two ways. On the one hand they help linearize non-linear problems much in the way that the well-known homogeneous coordinates do. On the other hand they place a truly higher class of transformations, namely conformal transformations at our disposal. Conformal transformations are transformations that preserve angles locally. They are closely linked to (complex) differentiable mappings and useful in a number of applications in physics and engineering. For example, functions that are governed by a potential can be transformed by a conformal mapping and are then still governed by a potential. Classically, only conformal mappings of the complex plane have been studied. CGA is a framework that allows the study of conformal mappings of spaces of arbitrary dimensions. Because Euclidean transformation (or rigid body motions) are a notable special case of conformal transformations, CGA has recently found applications in robotics, computer graphics, computer vision and other fields.

Different research groups may use \textit{variants} of geometric algebra that deviate from the one presented here to varying degrees. Some [LD03] use CGA with a slightly different notation or motivation. Others [Per08] employ CGA to describe geometric relationships, but perform calculations almost exclusively in classical linear algebra. In that case the Minkowski metric is implemented explicitly using a “metric matrix”. Still others (e.g. [Li08]) construct a geometric
algebra over different base spaces or implement different metrics that are tailored to their specific application.

1.3 Challenges

Until recently geometric algebra has been in the hands of mathematicians who valued its formal elegance and mathematical beauty. Those are virtues that can pay off in practical implementations, for example, allowing insights in previously hard problems, saving time or storage space in computations and making derivations easy to perform and understand. But these benefits come at a price.

In the past, geometric algebra has mostly been used as a generative tool, for applying known transformations to data, for deriving theoretical results or for the representation of geometric data at the end of some data processing chain. The power of geometric algebra as a tool for data analysis and data processing is largely unexplored. One reason for that is the relative difficulty in manipulating geometric algebra equations. It arises, for example, from the non-commutativity of the geometric product, the complicated interaction between multivector components or the unusual metric employed in the conformal model of geometric algebra.

Another reason is that geometric algebra is still lacking a convincing consistent model for noise or uncertainty. When trying to integrate redundant, ambiguous and noisy data such a model would be of great benefit.

1.4 Focus

In this thesis we focus on the use of CGA for fitting model parameters to geometric observation data in Euclidean spaces. Instead of completely replacing classical approaches, geometric algebra will act as a mediator between the geometric aspects of models and data on one side and the practical solution for parameters on the other. More concisely, geometric algebra will play the role of an interface between geometric intuition and classical linear algebra techniques.

The research presented here was funded by the NWO-project “Detection of Articulated Structures in Image Sequences” (DASIS), which had two main objectives: the application of CGA for the detection of articulated structures in image sequences and the development of CGA into a state where it can be used for geometric data processing in general. In pursuit of these goals we pick three representative practical — and sufficiently hard — problems in Euclidean space, namely determining a conformal transformation from correspondence data, inferring the structure of a kinematic chain from motion of its last limb and fitting a hyper-sphere to noisy point observations.

Using CGA for the description of the (rigid body) motions of articulated structures also allows the representation of conformal transformations in Euclidean space. In order to understand how CGA operates on rigid body motions, it is useful to investigate how it operates on full conformal transformations which are a natural part of the framework. Once we understand how to use observation data to determine a transformation in CGA, we can use it to decompose
that transformation. In the context of our research goals this is important, for example, to infer the links of an articulated structure even in cases where a direct observation of these links is not possible. In practice such situations may arise from occlusion or otherwise missing data. Finally, since Euclidean hyperspheres are naturally represented by vectors in CGA, studying their behavior in the presence of uncertainty we hope to gain insights into the inner workings of the CGA framework and how to deal with uncertainty in it.

In the course of solving these problems we investigate to what extent it is beneficial to use geometric algebra to represent the observation data, formulate the model equations and solve the resulting equations for the model parameters. We determine what kind of data is necessary or sufficient to determine the models, explore the theoretical limitations of the resulting fits and develop the methodology of geometric algebra further.

As we find that the formulation in geometric algebra does not take us all the way, we determine ways to interface it with classical linear algebra. This makes our geometric algebra equations amenable to the large body of established techniques and ongoing research regarding the use of classical linear algebra to fit models to data. By exploring to which extent classical approaches and techniques can be exploited to obtain geometric algebra based solutions we deepen our understanding of the geometric and algebraic concepts underlying the specific problems.

1.5. Overview

Chapter 2 of this thesis contains an introduction to geometric algebra based on [HS84, LD03, DFM07]. We familiarize the reader with basic concepts like the geometric product, the inner and outer product and multivectors in section 2.1. In section 2.3 we show how versors can be taken to represent orthogonal transformations which preserve the inner product and act as a morphism on the outer product of vectors.

Then, in section 2.4, we focus on the conformal model of geometric algebra, showing by which algebraic properties CGA vectors represent Euclidean hyperspheres. We analyze special cases leading to the representation of points and planes, before we continue to introduce more general multivectors that represent more complicated geometric objects. The versors of this high-dimensional model represent conformal transformations of Euclidean space that can be applied to any representable geometric object. Finally, in sections 2.5 and 2.6, we take a look at geometric algebra from the viewpoint of linear algebra as well as differential geometry. None of this is original work. Only the order in which the material is presented reflects the author’s preference. Chapter 2 is meant to facilitate understanding of the rest of this thesis and may serve as a reference for concepts used therein.

In chapter 3 we show how to determine a conformal transformation of $n$-dimensional Euclidean space from minimal exact correspondence data. Classically, conformal transformations are studied in — and at the same time restricted to — the (complex) plane, but CGA generalizes the concept to arbitrary dimensions.

In order to derive the transformation between corresponding data sets we use extended
1.5 Overview

Vahlen matrices, which we introduce in section 3.2. These are $2 \times 2$-matrices with Euclidean geometric algebra valued entries, whose structure helps to reduce the problem to finding a Euclidean rotation. To the best of our knowledge, both the reduction to a Euclidean problem as well as the final closed form solution are novel results. In section 3.3 we show that minimal correspondence data required to determine a conformal transformation consists of a localized frame of vectors and an additional point.

The correspondence data we use has to be exact in the sense that we assume the existence of a Euclidean rotation between the corresponding frames of vectors. If such a rotation does not exist — e.g. because the mutual angles between corresponding pairs of vectors are not the same — geometric algebra and classical approaches offer ways to approximate a solution which we present in section 3.3, too.

In chapter 4 we observe a complicated, nested 3D screw motion over time and decompose it into its constituent elemental 3D screw motions. Such nested motions occur when articulated structures move. These structures can be robotic arms or living creatures with an articulated skeleton. The individual joints are assumed to perform elemental 3D screw motions like axis rotations and translations.

When the component motions can be observed directly, they can be factored out of the composite motion or, conversely, the composite motion can be reconstructed from them. However, we assume that only the composite motion, e.g. the motion of the last limb in a kinematic chain, can be observed and show an efficient way to infer the parameters of the component motions in section 4.3.

We use the Lie group structure of 3D screw motions to describe the instantaneous motion (i.e. the apparent motion that an object performs from one instant to the next) of the observed limb and how this motion changes over time. In this endeavor geometric algebra offers a rather intuitive way to expose the relationship between geometrically meaningful parameters of 3D screw motions and the algebraic objects representing these motions. When it comes to solving correspondence equations for those parameters, however, we find it useful to employ matrix representations.

Our approach yields an explicit solution for composite motions consisting of up to three component motions. Beyond that solutions are possible in principle, but they become computationally very expensive to obtain.

We evaluate our approach by running experiments with simulated data. The details of the experimental setup as well as the quantitative results of those experiments are presented in appendix A.

In chapter 5 we present a Bayesian approach to Euclidean hypersphere estimation from noisy point data. One advantage of the Bayesian approach is that it forces the designer of an estimator to make explicit all the parameters he wants to estimate as well as all assumptions, beliefs and prior knowledge about them.

While neither the Bayesian approach to parameter estimation nor the estimation of circles or spheres from point data is novel, we have not found literature that applies one to the other. In sections 5.3 and 5.4 we find an explicit formula for the likelihood of hypersphere parameters
“radius” and “location of center” and generalize it to hyperspheres in arbitrary dimensions. Furthermore, we use the likelihood formula we derived in order to create a maximum likelihood estimator, which we compare to a least squares estimator in detail using simulated data in section 5.5. Extensive numerical results of this comparison can be found in appendix B.

In chapter 5 we do not use geometric algebra to derive our results, nor are we estimating parameters of geometric algebra quantities. However, the ability to represent Euclidean hyperspheres in CGA gave rise to our research in this area. We were unable to derive a full-fledged Bayesian noise model for CGA, but we find the theoretical results obtained here useful. They allow structural insights into the problem of parameter estimation from noisy data and any future approach to parameter estimation in CGA will have to conform to them.

In chapter 6 we summarize our findings in this thesis. With a deeper understanding of geometric algebra and more detailed knowledge of our results, we revisit some of the notions introduced in the present chapter and discuss some conclusions that we draw from our work. Finally, we present some questions that this thesis necessarily leaves open and give a brief outlook on possible future research.
Introduction to Geometric Algebra

Geometric algebra (also called Clifford algebra) is an extension to classical linear algebra, constructed over some vector space. Through the definition of an inner product it is equipped with a metric. The term geometric algebra emphasizes the idea that its elements are used to represent and transform geometric objects, which can be done in a coordinate-free manner. It has an invertible associative geometric product that can be decomposed into an inner and outer product.

In this chapter we provide an introduction to the algebraic structure of geometric algebra, first establishing the fundamental geometric product of the algebra in section 2.1 by describing its action on the elements of a vector space. In section 2.2 we introduce the concept of multivectors, algebraic entities that arise as the result of the geometric product between vectors and that are needed to close the algebra under the geometric product. We explain in section 2.3 how multivectors enable a compact representation of orthogonal transformations.

The algebraic entities of geometric algebra and their interaction readily allow geometric interpretation. The material presented in sections 2.1 through 2.3 is valid in all variants of geometric algebra. But in section 2.4 we introduce the conformal model of geometric algebra (CGA), a particular geometric algebra which can be used to represent a large number of geometric objects in Euclidean spaces and which unifies rigid body motions, Euclidean similarity transformations and conformal transformations as orthogonal linear transformations of a higher-dimensional space. Using techniques from section 2.3 these are then efficiently representable.
The work presented in sections 2.1 through 2.4 is entirely based on [HS84], [LD03] and [DFM07]. Sections 2.5 and 2.6 reflect the author’s perspective but do not contain original work. The concepts that are used can be found in the respective publications referenced there.

As opposed to the relatively young mathematical field of geometric algebra, classical linear algebra offers a host of tools — some of them analytically precise, others well established coping mechanisms — for solving equations exactly or approximately, analyzing transformations, dealing with inconsistencies etc. Being a multilinear algebra, geometric algebra yields to the application of some of these tools, while others have to be modified to acquire geometric algebra analogues. Section 2.5 provides a linear algebra perspective on geometric algebra.

On the other hand geometric algebra can be studied under the aspect of representing transformation groups. Some transformation groups of wide interest — such as the group of orthogonal linear transformations — are differentiable manifolds, which makes them amenable to differential geometry, a mathematical branch which comes with its own set of devices for analysis, calculation and description. In section 2.6 we briefly provide a view on geometric algebra from a differential geometry standpoint. This section also motivates the representation of versors by bivector exponentials.

Readers familiar with geometric algebra may skip sections 2.1 through 2.3. Section 2.4 introduces notation specific to the use of CGA in this thesis. In section 2.5 we consider the interface between geometric algebra and linear algebra in more detail. Therefore this section contributes to the central topic of this thesis. It may be especially interesting to readers with a strong background in linear algebra. Material presented in sections 2.5 and 2.6 prepares the reader for the analysis in chapter 4.

2.1. Inner, Outer and Geometric Product

Geometric algebra operates on a linear vector space just like “classical” linear algebra does. In addition to the known vector algebra, two concepts are introduced, namely that of the geometric product and that of multivectors. To understand these concepts it is useful to consider how geometric algebra works on vectors, e.g. in $n$-dimensional Euclidean space $\mathbb{R}^n$.

There, the scalar- or dot-product between vectors $\mathbf{a}$ and $\mathbf{b}$, denoted $\mathbf{a} \cdot \mathbf{b}$, is defined and specifies how to measure lengths and angles. This product is retained in geometric algebra, where it is called inner product. Applied to vectors, it behaves just like the well-known scalar product does in linear algebra.

In addition to the inner product, geometric algebra introduces an outer product or wedge product, which is identical to the outer product originally introduced by Grassmann. The outer product of two vectors $\mathbf{a}$ and $\mathbf{b}$ results in a 2-blade $\mathbf{a} \wedge \mathbf{b}$, which is an algebraic entity distinct from vectors and scalars. In its geometric interpretation the 2-blade $\mathbf{a} \wedge \mathbf{b}$ gives the oriented area, which arises when vector $\mathbf{a}$ is swept along vector $\mathbf{b}$. Its magnitude is exactly that of the scalar content of the resulting area and it is oriented in the sense that the 2-blade changes sign, if the order of the vectors is inverted, i.e. $\mathbf{a} \wedge \mathbf{b} = -\mathbf{b} \wedge \mathbf{a}$. Moreover, the 2-blade carries a sense of attitude, that is, if $\mathbf{a}$ and $\mathbf{b}$ are embedded in some higher-dimensional space,
2.1 Inner, Outer and Geometric Product

\( a \wedge b \) captures the spatial pose of \( a \) and \( b \). Figure 2.1 shows a 2-blade and distributivity of the outer product over addition in three dimensions.

\[ (a+b) \wedge c = a \wedge c + b \wedge c. \]

Figure 2.1: A 2-blade \( a \wedge b \) has the magnitude of the area of the parallelogram swept out by vector \( a \) along vector \( b \) and a direction. The outer product is distributive over addition.

In the same manner the outer product is defined between three vectors \( a, b \) and \( c \), resulting in a 3-blade \( a \wedge b \wedge c \), which represents the oriented volume spanned by these. Even more general, the outer product between \( r \) vectors results in a general \( r \)-blade.

The outer product has the following properties. As said before, it is anti-commutative between vectors. It is also associative and distributive over addition:

\[ a \wedge b = -b \wedge a, \]
\[ (a \wedge b) \wedge c = a \wedge (b \wedge c), \]
\[ a \wedge (b + c) = a \wedge b + a \wedge c, \]
\[ (a + b) \wedge c = a \wedge c + b \wedge c. \]

With respect to scalar multiplication, the following equation holds.

\[ a \wedge (\lambda b) = (\lambda a) \wedge b = \lambda (a \wedge b), \quad \lambda \in \mathbb{R} \]

As a consequence of property (2.1), the outer product vanishes if two vectors are linearly dependent, especially

\[ \lambda(a \wedge a) = 0, \quad \text{for all vectors } a \text{ and all } \lambda \in \mathbb{R}. \]

While the inner and outer product, respectively, may be familiar from different mathematical branches, the achievement of geometric algebra is to unite these two into one single operation, the geometric product introduced by Clifford. The geometric product between vectors \( a \) and \( b \) is denoted \( ab \). It can also be introduced independently of the inner and outer product.

We choose this approach here and show a little later how the geometric product can be decomposed into inner and outer product. The geometric product obeys the following axioms.

The geometric product is associative

\[ (ab)c = a(bc). \]
The geometric product is *distributive over addition*

\[ a(b + c) = ab + ac, \quad (2.8) \]
\[ (a + b)c = ac + bc. \quad (2.9) \]

The square of any vector \( a \), i.e. the geometric product of \( a \) with itself, denoted \( a^2 \), is a scalar

\[ a^2 = aa \in \mathbb{R}. \quad (2.10) \]

Note that axiom \((2.10)\) leads to the very useful fact that the geometric product between vectors is *invertible*. Just define the inverse of a vector \( a \) as

\[ a^{-1} = \frac{a}{a^2}, \quad \text{for any vector } a \text{ with } a^2 \neq 0, \quad (2.11) \]

and we find that

\[ aa^{-1} = a \frac{a}{a^2} = \frac{a^2}{a^2} = 1. \quad (2.12) \]

To derive one particular property of the geometric product, take the square of the sum of two vectors,

\[ (a + b)^2 = (a + b)(a + b) = a^2 + ab + ba + b^2. \quad (2.13) \]

By axiom \((2.10)\), \( a^2 \) and \( b^2 \) will be scalars and so must \( ab + ba \). We observe that this scalar term is bilinear and symmetric in \( a \) and \( b \), which allows us to identify it with the scalar product between vectors. A scalar factor of \( \frac{1}{2} \) is involved for reasons of consistency, to make sure that \( a^2 = a \cdot a \).

\[ a \cdot b = \frac{1}{2} (ab + ba). \quad (2.14) \]

The full geometric product contains this symmetric part, but also a remaining antisymmetric part, which we can identify with the outer product, because it fulfills all the required properties \((2.1) - (2.4)\);

\[ a \wedge b = \frac{1}{2} (ab - ba). \quad (2.15) \]

Combining equations \((2.14)\) and \((2.15)\) one sees that the geometric product between vectors can be written as a sum of the inner and outer product,

\[ ab = a \cdot b + a \wedge b. \quad (2.16) \]

Note that \( a \wedge b \) vanishes, if \( a \) and \( b \) are parallel and, if \( a \) and \( b \) are perpendicular, \( a \cdot b \) vanishes. In general, however, the geometric product between two vectors returns the sum of a scalar and a 2-blade. Note that the geometric product is associative, but not commutative.
2.2. Grades, Bases and Frames

The above considerations lead to the concept of a general multivector. By definition, the outer product of \( r \) vectors results in an \( r \)-blade

\[
B = b_1 \wedge \cdots \wedge b_r.
\]  

(2.17)

Thus, by the outer product it is possible to create algebraic entities of different grade. Scalars are assigned grade zero and vectors grade 1; 2-blades are said to have grade 2, 3-blades have grade 3 and so on.

Now, a geometric algebra — as a mathematical structure — can be considered. It is a multilinear algebra, denoted \( \mathcal{G}(V) \), over a vector space \( V \) with the geometric product defined by the properties introduced above. In this thesis we will take the position that the vector space \( V \) does not inherently have to have a metric. However, the geometric algebra constructed over \( V \) defines an inner product, which induces a metric\(^2\). If the vector space \( V \) is assumed to have a metric, the geometric algebra constructed over it, inherits that metric. For example, in section 2.4 we will write \( \mathcal{G}(\mathbb{R}^{p,q}) \) to denote a geometric algebra constructed over \( \mathbb{R}^{p+q} \) with a \( (p,q) \) signature.

The maximum grade that an \( r \)-blade in a geometric algebra can have depends on the dimension of the vector space the geometric algebra is based on (but not on its metric). In the geometric algebra over an \( n \)-dimensional vector space, the outer product of \( n + 1 \) vectors is always zero, because of property (2.6) of the outer product. So the maximum grade that a blade can assume is \( n \). The grade-\( n \) object created by taking the outer product of \( n \) vectors is unique up to a scalar factor for the geometric algebra. If it is normalized, it is denoted \( I \) and called the (unit) pseudoscalar. Thus, any grade-\( r \) object, \( A \), created this way generates its own geometric (sub-)algebra (borrowing the metric from the original space) sometimes denoted \( \mathcal{G}(A) = \mathcal{G}(\lambda I) = \mathcal{G}(I) \) and every geometric algebra determines a pseudoscalar \( I \).

As opposed to the outer product of vectors, the full geometric product returns a sum of objects of different grade, called a general multivector. Such a sum is assigned the grade of the object of the highest grade comprising it, e.g. the geometric product of two vectors returns a general multivector of grade 2. The blades of grade \( r \) introduced above are multivectors which can be written as the outer product of \( r \) vectors. Note that every multivector can be written as a sum of blades. We would like to stress that in the following when we invoke the term multivector without further qualification it expressly includes blades, vectors and scalars as special cases.

Many calculational and rewriting rules extend to blades of higher grades and — by linearity of the different products — to multivectors. We introduce a subset of these extensions insofar as we find them useful within the scope of this thesis. For a more comprehensive introduction we refer the reader to [HS84, LD03, DFM07].

For example, we note that (2.16) can be extended to the case where \( a \) is a vector and \( B \) is

\(^2\)To explicitly call attention to that fact one might denote the geometric algebra as \( \mathcal{G}(V, \cdot) \), but we drop the second argument as it is understood that a metric is defined on \( \mathcal{G}(V) \).
a multivector by
\[ aB = a \cdot B + a \wedge B. \] (2.18)
By convention the inner product of a vector with a scalar (i.e. a multivector of grade zero) is zero, while the outer product with a scalar returns a scalar multiple of the other factor.

If \( B \) is an \( r \)-blade, then the inner product of a vector \( a \) with it is given by
\[ a \cdot B = \sum_{k=1}^{r} (-1)^{k-1} (a \cdot b_k) b_1 \wedge \cdots \wedge \check{b}_k \wedge \cdots \wedge b_r, \] (2.19)
where the check over \( b_k \) indicates that this vector is missing in the outer product. The result is a blade of grade \( r - 1 \).

It is useful to introduce a grade selection operator, \( \langle \rangle_r \), which selects the components of grade \( r \) of a multivector. Multivectors with the property
\[ \langle A \rangle_r = A, \text{ for any } r \in \mathbb{Z} \] (2.20)
are called pure \( r \)-vectors. Not every pure \( r \)-vector is also an \( r \)-blade. For example, in four dimensions one can construct a bivector
\[ A = e_1 \wedge e_2 + e_3 \wedge e_4, \] (2.21)
which cannot be written as the outer product of any number of vectors, although \( \langle A \rangle_2 = A \). The scalar part \( \langle A \rangle_0 \) is sometimes abbreviated to \( \langle A \rangle \). By convention the grade selection operator results in zero when its index is negative.

To integrate entities of different grade into some closed form standard formulas (for example, in section 2.3) we need operators that change the sign of an entity depending on its grade. In particular, for blades of grade \( r = 0, 1, 2, \ldots \) we need the progressions of signs \( \{+1, -1, +1, -1, +1, -1, \ldots \} \) and \( \{+1, -1, -1, +1, +1, -1, -1, \ldots \} \). We introduce the involutions grade involution \( \hat{\circ} \) and Clifford conjugation \( \bar{\circ} \) on elements of \( \mathcal{G}(I) \) by
\[ \hat{V} = \sum_r (-1)^r \langle V \rangle_r, \quad V \in \mathcal{G}(I), \] (2.22)
\[ \bar{V} = \sum_r (-1)^{\frac{r(r+1)}{2}} \langle V \rangle_r, \quad V \in \mathcal{G}(I). \] (2.23)
The grade involution is an automorphism of \( \mathcal{G}(I) \), while the Clifford conjugation is an anti-automorphism of \( \mathcal{G}(I) \), i.e. \( \hat{V}W = \hat{W} \hat{V} \) and \( \bar{V} \bar{W} = \bar{W} \bar{V} \) for \( V, W \in \mathcal{G}(I) \). As a consequence of (2.22) and (2.23) we also get the progression of signs \( \{+1, -1, -1, +1, +1, -1, -1, \ldots \} \) by
\[ \hat{V} = \sum_r (-1)^{\frac{r(r+3)}{2}} \langle V \rangle_r, \quad V \in \mathcal{G}(I). \] (2.24)

Basis vectors of an \( n \)-dimensional Euclidean vector space are denoted \( \{e_k\} \), shorthand for
$\bigcup_{k=1}^{n}\{e_k\}$. Such a set makes up a frame and makes it possible to write any vector in that space in terms of coordinates, denoted $\alpha^k$, i.e.

$$a = \alpha^1e_1 + \cdots + \alpha^ne_n. \quad (2.25)$$

An important notion is that of duality. For any $r$-vector $B$ the $|n-r|$-vector $(B \cdot I)$ is called the dual of $B$ by the $n$-blade $I$. We get the valuable relations

$$
(a \cdot B)I = a \wedge (BI) \quad \text{if} \quad a \wedge I = 0, \quad (2.26)\\
(a \wedge B)I = a \cdot (BI) \quad \text{if} \quad a \wedge I = 0. \quad (2.27)
$$

Of course, our considerations in this section are not restricted to a basis of Euclidean $\mathbb{R}^n$. Any space can be described, and the definitions of inner, outer and geometric product hold.

### 2.3. Orthogonal Transformations in Geometric Algebra

Any invertible vector of a geometric algebra can be used to represent an orthogonal transformation using a sandwiching geometric product. Consider a vector space $V$, the vectors $v, w \in V$ and the invertible vector $u \in V$ and define the linear transformation $f_u : V \to V$ acting on vectors by

$$f_u(v) = -uvu^{-1}. \quad (2.28)$$

This transformation can be interpreted geometrically as performing a reflection of the vector argument $v$ in a plane perpendicular to the vector parameter $u$. As a shorthand we shall call this a reflection at the vector $u$. See figure 2.2 for a visualization of this fact. To demonstrate

![Figure 2.2](image.png)

**Figure 2.2:** The transformation $f_u(v) = -uvu^{-1}$ is linear in $v$. It can be interpreted as a reflection of the vector $v$ at the vector $u$.

it algebraically we use (2.18) and employ (2.19) to “distribute” the inner product over the outer product with appropriate sign changes,

$$u \cdot (v \wedge w) = (u \cdot v)w - (u \cdot w)v. \quad (2.29)$$
We get
\[
\begin{align*}
  f_u(v) &= -uvu/u^2 \\
  &= (-u \cdot (vu) - u \wedge (vu))/u^2 \\
  &= (-u \cdot (v \cdot u + v \wedge u) - u \wedge (v \cdot u))/u^2 \\
  &= (-u \cdot (v \wedge u) - u \wedge (v \cdot u))/u^2 \\
  &= -(u \cdot v)u + (u \cdot u)v - u(v \cdot u))/u^2 \\
  &= (u^2v - 2(u \cdot v)u)/u^2 \\
  &= v - 2(u \cdot v)u/u^2. \\
\end{align*}
\]
(2.30)

By decomposing \(v\) into a component \(v_\perp\) perpendicular to \(u\) and a component \(v_\parallel\) parallel to \(u\) we have
\[
\begin{align*}
  f_u(v) &= f_u(v_\perp + v_\parallel) \\
  &= f_u(v_\perp) + f_u(v_\parallel) \\
  &= v_\perp - 2(u \cdot v_\perp)u/u^2 + v_\parallel - 2(u \cdot v_\parallel)u/u^2 \\
  &= v_\perp + v_\parallel - 2v_\parallel \\
  &= v_\perp - v_\parallel. \\
\end{align*}
\]
(2.31)

Therefore, in a Euclidean geometric algebra, \(f_u(v)\) is a reflection of \(v\) in the hyperplane with normal vector \(u\).

Moreover, this transformation preserves the inner product. Algebraically this can be shown as follows.
\[
\begin{align*}
  f_u(v) \cdot f_u(w) &= (v_\perp - v_\parallel) \cdot (w_\perp - w_\parallel) \\
  &= v_\perp \cdot w_\perp - v_\parallel \cdot w_\perp - v_\perp \cdot w_\parallel + v_\parallel \cdot w_\parallel \\
  &= v_\perp \cdot w_\perp + v_\parallel \cdot w_\perp + v_\perp \cdot w_\parallel + v_\parallel \cdot w_\parallel \\
  &= (v_\perp + v_\parallel) \cdot (w_\perp + w_\parallel) \\
  &= v \cdot w, \\
\end{align*}
\]
where \(v_\parallel \cdot w_\perp = v_\perp \cdot w_\parallel = 0\). Thus, every linear transformation of the form (2.28) is an orthogonal transformation. Especially, every concatenation of linear transformations of the form (2.28) is an orthogonal transformation. Denoting the concatenation of functions by \(\circ\), we have
\[
\begin{align*}
  f_{u_1} \circ f_{u_2} \circ \cdots \circ f_{u_k}(v) &= (-1)^k u_1u_2 \ldots u_kv_k^{-1} \ldots u_2^{-1}u_1^{-1}, \quad k \in \mathbb{N}, \\
\end{align*}
\]
(2.33)
in which we can substitute
\[
  u_1u_2 \ldots u_k = S \tag{2.34}
\]
in order to obtain
\[
  f_S(v) = SvS^{-1}, \tag{2.35}
\]
where the grade involution \((2.22)\) enforces the proper sign changes \((-1)^k\) depending on the grade \(k\) of \(S\). The Cartan-Dieudonné theorem (e.g. [Mar87], chapter 13) states that every orthogonal transformation of an \(n\)-dimensional non-degenerate symmetric bilinear space over a field with characteristic not equal to 2 is a composition of (at most \(n\)) reflections. Therefore, every orthogonal transformation on such fields can be written in the form \((2.34)\).

The geometric product \(S\) of a number \(k \in \mathbb{N}\) of invertible vectors is called a versor. If \(k\) is even (respectively odd) \(S\) is called an even (respectively odd) versor. Due to the properties of the grade involution \((2.22)\) and the Clifford conjugation \((2.23)\) a well as axiom \((2.10)\), for any versor \(S\),

\[
S \hat{S} \in \mathbb{R} \setminus \{0\}. \tag{2.36}
\]

We will call \(S\) a unit versor if \(S \hat{S} = 1\). For the sake of brevity we will call the sandwiching geometric product of the form \((2.35)\) versor product. The versor product preserves the structure of the outer product, and a mapping \(f_S : \mathcal{G}(V) \to \mathcal{G}(V)\) is induced. For arbitrary blades \(X\) of grade \(m\) we define

\[
f_S(X) = (-1)^m SX \hat{S}^{-1} = (-1)^m SX S^{-1}. \tag{2.37}
\]

For example, for the 2-blade \(v \wedge w\) we then get,

\[
f_S(v \wedge w) = f_S(v) \wedge f_S(w), \tag{2.38}
\]

as we will show now using the properties of the geometric product introduced so far. We have

\[
f_S(v \wedge w) = (-1)^{2k} u_k \ldots u_1 (v \wedge w) u_1^{-1} \ldots u_k^{-1}
= u_k \ldots u_1 (v \wedge w) u_1^{-1} \ldots u_k^{-1}
= u_k \ldots u_1 (vw - v \cdot w) u_1^{-1} \ldots u_k^{-1}
= u_k \ldots u_1 (vw) u_1^{-1} \ldots u_k^{-1} - u_k \ldots u_1 (v \cdot w) u_1^{-1} \ldots u_k^{-1}
= u_k \ldots u_1 (vw) u_1^{-1} \ldots u_k^{-1} - (v \cdot w) u_k \ldots u_1 u_1^{-1} \ldots u_k^{-1}
= u_k \ldots u_1 (vw) u_1^{-1} \ldots u_k^{-1} - (v \cdot w)
= u_k \ldots u_1 v(u_1^{-1} \ldots u_k^{-1} u_k \ldots u_1 w) u_1^{-1} \ldots u_k^{-1} - (v \cdot w)
= (-1)^k f_S(v) (-1)^k f_S(w) - f_S(v) \cdot f_S(w)
= (-1)^{2k} f_S(v) f_S(w) - f_S(v) \cdot f_S(w)
= f_S(v) \wedge f_S(w). \tag{2.39}
\]

The versor product is therefore called an outermorphism in [HS84]. By \((2.38)\) the versor product extends to blade arguments and by linearity to general multivector arguments, because any multivector can be written as a sum of blades.

Instead of constructing even versors by the geometric product of an even number of invertible vectors, they can be written as the exponential of a bivector. Every even unit versor can be
Successive reflection in two hyperplanes \( n_1 \) and \( n_2 \) through the origin amounts to a rotation about the origin with the plane of rotation perpendicular to both hyperplanes. The versor \( S = n_2 n_1 \), which represents this rotation — and therefore is sometimes called a rotator — is a purely Euclidean multivector.

written as the exponential of a bivector and every bivector exponential results in an even unit versor. We will consider this notion in more detail in section 2.6. Here, we only point out that in this way it is possible to characterize a special orthogonal transformation by the minimal number of parameters. The number of degrees of freedom in a special orthogonal transformation is \( \binom{n}{2} \), the same as the number of degrees of freedom in a general bivector from the geometric algebra over an \( n \)-dimensional vector space.

Since versors represent orthogonal linear transformations, they can be interpreted geometrically. In Euclidean space, for example, an even versor represents a rotation about the coordinate origin, i.e. a transformation that preserves lengths of vectors and angles between vectors (which are defined via the inner product); see figure 2.3 for an example. According to the geometrical transformation that they represent versors are often given specific names, such as “rotators” (for rotation versors), “translators” (for translation versors), “motors” (for rigid body motion versors) etc. For simplicity we will just call them versors, irrespective of the geometrical transformation they represent, emphasizing their algebraic properties rather than their geometric interpretation.

Of special interest as a versor is the mid-plane between two linearly independent vectors of equal length. Let \( \mathbf{v} \) and \( \mathbf{w} \) be such vectors, i.e. \( \mathbf{v}^2 = \mathbf{w}^2 \). Then the mid-plane between the two is the vector \( \mathbf{u} = \mathbf{w} - \mathbf{v} \). What makes the mid-plane useful is the fact that reflection in it
aligns two vectors of equal length, i.e. \( \mathbf{w} = \mathbf{u}\mathbf{v}^{-1} \), as we can see algebraically by

\[
\mathbf{u}\mathbf{v}^{-1} = -\mathbf{u}\mathbf{v}/\mathbf{u}^2 \\
= -(\mathbf{w} - \mathbf{v})(\mathbf{w} - \mathbf{v})/(\mathbf{w} - \mathbf{v})^2 \\
= (-\mathbf{w}\mathbf{v}\mathbf{w} + \mathbf{v}\mathbf{w} + \mathbf{w}\mathbf{v} - \mathbf{w}\mathbf{v}\mathbf{w})/(\mathbf{w} - \mathbf{v})^2 \\
= (\mathbf{v}^2\mathbf{w} + \mathbf{v}^2\mathbf{w} - \mathbf{v}^2\mathbf{w} - \mathbf{w}\mathbf{v}\mathbf{w})/(\mathbf{w} - \mathbf{v})^2 \\
= \mathbf{w} (\mathbf{w}^2 + \mathbf{v}^2 - \mathbf{w}\mathbf{v} - \mathbf{v}\mathbf{w})/(\mathbf{w} - \mathbf{v})^2 \\
= \mathbf{w}. 
\] 

(2.40)

2.4. The Conformal Model of Geometric Algebra (CGA)

The conformal model of geometric algebra (CGA) is the smallest algebra we know that represents rigid body motions minimally as the exponential of bivectors. It is the geometric algebra \( \mathcal{G}(\mathbb{R}^{n+1,1}) \) constructed over the Minkowski space \( \mathbb{R}^{n+1,1} \), that is, a real vector space with \( n + 1 \) basis vectors that square to a positive real number and one basis vector that squares to a negative real number. We will use this model to represent entities — such as points, circles and directions — residing in Euclidean vector space \( \mathbb{R}^n \), which is a subspace of the Minkowski vector space. For brevity’s sake we will simply call vectors from this Euclidean space-to-be-represented Euclidean vectors and denote them by lower case bold italic font. A general vector from CGA we will call a conformal vector and denote it by lower case italic font. Note that conformal vectors can, and in general do, contain Euclidean parts. Multivectors will be denoted in uppercase italic font.

In order to work with CGA one can introduce a vector basis \{\( e_+, e_1, \ldots, e_n, e_- \)\} with \( e_i^2 = 1 \), for \( i \in \{1, \ldots, n\} \cup \{+\} \) and \( e_-^2 = -1 \), as well as \( e_i \cdot e_j = 0 \), for \( i \neq j \). This basis is called the orthonormal basis, but it is not the only basis used in practice. The mixed signature of the Minkowski space allows for the construction of null or isotropic vectors, which square to zero. Equation (2.11) implies that null vectors are not invertible.

For CGA it is common to introduce the null basis \{\( n_0, e_1, \ldots, e_n, n_\infty \)\} with orthonormal Euclidean vectors \( e_i, i = 1, \ldots, n \) and \( e_i \cdot e_j = 0 \), for \( i = 1, \ldots, n, j = 0, \infty \), while \( n_0 \) and \( n_\infty \) are defined in terms of the orthonormal basis as

\[
n_0 = \frac{1}{2} (e_- + e_+), \\
n_\infty = e_- - e_+, 
\]

(2.41, 2.42)

which implies that

\[
n_0^2 = n_\infty^2 = 0 \\
n_0 \cdot n_\infty = -1.
\]

(2.43, 2.44)
The conformal model of geometric algebra can be used to represent a variety of geometric objects in Euclidean space \cite{DFM07, HJ03}. The lowest-dimensional geometric object in Euclidean space, a Euclidean point, is represented in CGA by a (non-invertible) null vector of the form

\[ p = \alpha \left( n_0 + \mathbf{p} + \frac{1}{2} \mathbf{p}^2 n_\infty \right), \quad (2.45) \]

where \( \mathbf{p} \) denotes the Euclidean position of the point being represented and \( \alpha \in \mathbb{R} \) is called the point’s *weight*, which can be viewed as a homogeneous degree of freedom. As a result, \( n_0 \) represents the point at the origin while \( n_\infty \) represents the point at infinity. When we talk about a conformal vector representing a point, we will call it a *conformal point* \( \mathbf{p} \) in order to set it apart from its position in Euclidean space \( \mathbf{p} \). Note that that position can be retrieved from a conformal point by

\[ \mathbf{p} = \mathbf{p} - n_\infty \cdot \mathbf{p}, \quad (2.46) \]

where \( -n_\infty \cdot \mathbf{p} \) retrieves the point’s weight \( \alpha \).

The inner product between two conformal points, \( \mathbf{p} \) and \( \mathbf{q} \) with unit weights results in

\[ \mathbf{p} \cdot \mathbf{q} = \left( -\frac{1}{2} \mathbf{q}^2 + \mathbf{p} \cdot \mathbf{q} - \frac{1}{2} \mathbf{p}^2 \right) \]

\[ = -\frac{1}{2} (\mathbf{p} - \mathbf{q})^2, \quad (2.47) \]

which is proportional to the squared Euclidean distance between the points. This useful fact is one of the main reasons for assuming a Minkowski metric and for representing points as (2.45) in CGA. It follows that for conformal points \( \mathbf{p} \) and \( \mathbf{q} \), \( \mathbf{p} \cdot \mathbf{q} = 0 \) if and only if \( \mathbf{p} = \mathbf{q} \) up to a scalar factor.

A general conformal vector does not necessarily obey (2.45), in particular the relationship between the Euclidean part and the \( n_\infty \)-coordinate. But we can bring an arbitrary conformal vector into the form

\[ \mathbf{s} = \alpha \left( n_0 + \mathbf{s} + \frac{1}{2} (\mathbf{s}^2 - \rho^2) n_\infty \right). \quad (2.48) \]

This represents a hypersphere \cite{DFM07, HJ03} with its center at Euclidean location \( \mathbf{s} \) and squared radius \( \rho^2 \) in the following sense. For any conformal point \( \mathbf{p} \),

\[ \mathbf{p} \cdot \mathbf{s} = \alpha_p \left( n_0 + \mathbf{p} + \frac{1}{2} \mathbf{p}^2 n_\infty \right) \cdot \alpha_s \left( n_0 + \mathbf{s} + \frac{1}{2} (\mathbf{s}^2 - \rho^2) n_\infty \right) \]

\[ = -\alpha_p \alpha_s \frac{1}{2} (\mathbf{s}^2 - \rho^2) - \alpha_p \alpha_s \frac{1}{2} \mathbf{p}^2 + \alpha_p \alpha_s \mathbf{p} \cdot \mathbf{s} \]

\[ = -\alpha_p \alpha_s \frac{1}{2} ((\mathbf{s} - \mathbf{p})^2 - \rho^2), \quad (2.49) \]

which for nonzero weights is zero if and only if \( (\mathbf{s} - \mathbf{p})^2 = \rho^2 \), i.e. the conformal point \( \mathbf{p} \) lies on the hypersphere \( \mathbf{s} \). Note that (2.48) allows the interpretation of conformal points as hyperspheres with zero radius. That is, substituting \( \rho = 0 \) into (2.48) results in (2.45). It is possible to construct a conformal vector of the form (2.48) with an \( n_\infty \)-component greater
than \( \frac{1}{2}s^2 \), which could be interpreted as a hypersphere with imaginary radius, i.e. \( \rho^2 < 0 \). Even though of seemingly little real geometric value, imaginary spheres have to be included in the conformal model of geometric algebra for closure, as they appear as the result of certain calculations.

By the same token as in (2.49), a hyperplane \( \pi \) is represented by its unit normal vector \( n \) and its scalar offset \( \delta \in \mathbb{R} \) from the origin as

\[
\pi = \alpha (n + \delta n_{\infty}).
\]  

(2.50)

We evaluate the inner product with an arbitrary conformal point \( p \) as

\[
p \cdot \pi = \alpha_p \left( \alpha_\pi \left( n_0 + p + \frac{1}{2}p^2 n_{\infty} \right) \cdot \alpha_\pi (n + \delta n_{\infty}) \right)
= -\alpha_p \alpha_\pi \delta + \alpha_p \alpha_\pi p \cdot n
= \alpha_p \alpha_\pi (p \cdot n - \delta),
\]

(2.51)

which for nonzero weights is zero if and only if \( p \cdot n = \delta \), i.e. \( p \) lies on the hyperplane \( \pi \). It can be shown that by a limiting process (2.48) allows the interpretation of hyperplanes as hyperspheres centered at infinity with infinite radius.

For obvious reasons, this way of representing a geometric object as the locus of all conformal points which have a zero inner product with the representing vector is called the inner product null space (IPNS) representation by Perwass [Per08]. The inner product can also be used to define local angles of intersection between two represented objects [HJ03]. For example, two hyperspheres of unit weight, \( s_1 \) and \( s_2 \), intersect locally under an angle of

\[
\theta = \arccos(s_1 \cdot s_2).
\]

(2.52)

So far we have focused on simple conformal vectors, but a number of geometric objects can be represented employing the outer product between conformal vectors. We will introduce some of them in as far as they are useful in the context of this thesis. For a more comprehensive account see, for example, [DFM07].

A (weighted) direction can be represented by

\[
D = \alpha (d \land n_{\infty}).
\]

(2.53)

This null 2-blade has the property that only the point at infinity is incident with it (i.e. \( n_{\infty} \cdot D = 0 \), as well as the property that \( D^2 = D \cdot D = 0 \), but for no other point \( p \) is \( p \cdot D = 0 \)). A tangent vector can be identified with a Euclidean direction vector anchored at a given Euclidean point. It is represented by a conformal tangent, \( t \), a null 2-blade that retains information about a Euclidean direction, \( t \), as well as a location, \( p \), represented by a conformal point \( p \). Formally,

\[
t = -p \cdot (p \land t \land n_{\infty}).
\]

(2.54)
In chapter 3 we will make use of a localized frame, i.e. a set of linearly independent conformal tangents anchored at a common point.

Another type of object that can be represented directly in CGA are circles, which are curves uniquely determined by three conformal points $p_1, p_2, p_3$. The equation

$$C = p_1 \wedge p_2 \wedge p_3 \quad (2.55)$$

represents a circle in the sense that any conformal point $p$ lying on it yields a zero outer product, i.e.

$$p \wedge C = 0, \quad (2.56)$$

if and only if $p$ lies on $C$. Note that this representation uses the outer product rather than the inner product to determine incidence and therefore is called the outer product null space (OPNS) representation of a circle. By using the duality relationship (2.27) we could equivalently use the IPNS representation of the circle, but we will adhere to the definition (2.55) instead, because it is geometrically more intuitive to construct a circle representation as the outer product of three points lying on it. Moreover, a conformal tangent $t$ to a conformal circle $C$ in a conformal point $p$ on $C$ can be easily determined by

$$t = p \cdot C. \quad (2.57)$$

Note that a conformal line is a special case of a conformal circle containing the point at infinity. We can write

$$L = p_1 \wedge p_2 \wedge n_\infty = p_1 \wedge t \wedge n_\infty, \quad (2.58)$$

where $t = p_2 - p_1$.

We would like to emphasize that these constructions hold in any dimension, but unlike hyperspheres (who are named for their co-dimension) circles are always one-dimensional entities in a two-dimensional "carrier" space. There are no "hypercircles." See figure 2.4 for a visualization in 3D. In chapter 3 we will use intersecting circles to determine a localized frame.

Since the versor product introduced in section 2.3 represents the reflection at a vector and invertible vectors in CGA represent hyperspheres, a versor in CGA represents a series of inversions — i.e. generalized reflections — in hyperspheres. See figure 2.5 for an example of an inversion.

The concept of the mid-plane, introduced at the end of section 2.3, holds in CGA. Let $p$ and $p'$ be two distinct conformal points. Since conformal points are represented by null vectors, both of them square to the same number, namely zero. Then $s = p' - p$ represents a Euclidean hypersphere, inversion in which makes the points coincide. If moreover $p$ and $p'$ have equal weight, $s = p' - p$ represents the Euclidean mid-hyperplane between them. More generally, $p$ and $p'$ can be taken to represent Euclidean hyperspheres. If and only if they square to the same number, is $s = p' - p$ the versor that aligns the two. Note that for simplicity we will
Figure 2.4.: A conformal circle $C$ represents a circle in Euclidean space directly by the outer product of three conformal points $p_1$, $p_2$ and $p_3$. The conformal tangent $t$ to the oriented circle in a conformal point $p$ on $C$ is given by $t = p \cdot C$. Here the circle is shown with the 2-dimensional subspace of $\mathbb{R}^3$ in which it lies. The conformal point at the origin, $n_0$, is shown for reference.

Figure 2.5.: An inversion in a hypersphere is a generalized reflection. The point at infinity is mapped to the hypersphere’s center. Here we see a point $p$ and a line $L$ mapped to $p'$ and $L'$, respectively, by an inversion in a circle $s$, i.e. a 2-dimensional hypersphere.

continue calling this versor the mid-plane, even though in general it does not represent a plane, nor does it necessarily lie in the (Euclidean) middle of $p$ and $p'$.

By Liouville’s theorem every conformal transformation of Euclidean space is made up of inversions — i.e. reflections in suitably chosen spheres or planes. In conjunction with the Cartan-Dieudonné theorem this means that the orthogonal transformations of $\mathbb{R}^{n+1,1}$ are the conformal transformations of the Euclidean space $\mathbb{R}^n$ which it represents. A first in-depth algebraical treatment of the versor representation of the conformal group using Clifford algebra can be found in [Ang80]. We will return to conformal transformations in more detail in chapter 3.
2.5. Relation to Linear Algebra

Due to the linearity of the geometric product — and its more specialized companions, the inner and outer product — a geometric algebra over an $n$-dimensional vector space can be expressed as a linear algebra over a $2^n$-dimensional vector space with the basis consisting of a number of basis blades of all possible grades. For example the geometric algebra over the three-dimensional Euclidean space, $\mathcal{G}(\mathbb{R}^3)$, can be expressed as a vector space with the basis $\{1, e_1, e_2, e_3, e_2 \wedge e_3, e_3 \wedge e_1, e_1 \wedge e_2, e_1 \wedge e_2 \wedge e_3\}$. Any multivector from $\mathcal{G}(\mathbb{R}^3)$ can be written as a linear combination of these basis elements. Linear algebra would then study the linear mappings — e.g. represented by $8 \times 8$-matrices acting on the basis components — between those vector elements. The inner, outer and geometric product are particular instances of such mappings.

An obvious advantage of this approach is that linear algebra is a well established framework. Techniques for solving equations in it are readily available, accepted criteria for assessing the quality of a solution exist and have been well studied, methods for analysis such as the singular value decomposition and a number of other matrix decompositions are commonplace. When it comes to implementation on a computer, one finds highly optimized hardware, specialized sub-processors and a vast number of software libraries and tools for facilitating calculations in linear algebra. For an efficient implementation of geometric algebra based on linear algebra, see for example [Fon07]. Software for automatic code generation using this geometric algebra implementation (Gaigen 2.5 [Fon]), for optimizing geometric algebra based algorithms (Gaalop [PH]) and for geometric algebra based scientific calculation and visualization (CLUCalc [Per]) is available online and uses linear algebra implementations.

A reason why geometric algebra interfaces with classical linear algebra is that they are both based on similar principles. Just like the geometric product, matrix multiplication is associative but not commutative. For two multivectors $A$ and $B$ their geometric product $AB$ is a transformation linear in $A$ as well as in $B$. Since in classical linear algebra linear transformations are represented by matrices, the geometric product $AB$ is either the matrix that represents multiplication on the left with $A$ acting on $B$ or the matrix that represents multiplication on the right with $B$ acting on $A$, depending on which viewpoint we take.

In order to faithfully represent the geometric product between general multivectors we have to resort to the language of tensors. For any given geometric algebra Perwass [Per08] introduces the geometric product tensor $\Gamma^k_{ij}$. Given a $2^n$-dimensional blade basis, multivectors can be represented by a collection of their coefficients, similar to the representation of vectors in classical linear algebra. With the scalar components $\alpha^i$ of $A$ and $\beta^j$ of $B$ the components of the multivector $AB$ that is the geometric product between $A$ and $B$ are given by $\gamma^k = \Gamma^k_{ij} \alpha^i \beta^j$, where summation over repeated indices is implied. The derived inner and outer product and the grade selection operator can be represented similarly by specific tensors.

While this enables us to apply linear algebra techniques, casting geometric algebra problems into linear algebra introduces a number of disadvantages. Besides the high dimensionality ($2^n$) of the formulation, the involved matrices have to be designed specifically for the problem at
hand using the geometric product tensor. Moreover, when using classical linear algebra on the
representation of geometric algebra concepts, one enforces a (often tacitly assumed Euclidean)
distance metric on the vector representation of multivectors. As a result entities from the
geometric algebra are blended in ways that may not be admissible in the problem’s context
or not geometrically meaningful — e.g. what is the “proper” distance metric between a vector
and, say, a 2-blade, or between a conformal point and a versor?

Additionally, it is not easy to switch back and forth between the different formulations.
Consider a versor $S$ acting on a vector $v$ by the versor product $Sv\hat{S}^{-1}$. The linear algebra
formulation of this transformation would be the orthogonal matrix $M$ with
$$M^k_l = \Gamma^k_{ij} \Gamma^j_{lm} \sigma^i \sigma'_{m},$$
where $\sigma$ and $\sigma'$ denote the components of $S$ and $\hat{S}^{-1}$, respectively. In geometric algebra
the vector argument $v$ is positioned awkwardly in the transformation equation and it is not easy
to isolate it or solve the equation for the versor components. In classical linear algebra
the term $Mv$ is easier to handle, different techniques to invert matrix $M$ can be attempted. On
the other hand, recovering the versor’s components from such a matrix is a very difficult task
— as is tracking the changes of the matrix entries with changes of the versor or assigning a
meaningful distance measure to both representations.

Finally, some established linear algebra techniques allow us to find a solution to some system
of equations, which is even optimal by some criterion, say, in a least squares sense. But the
solution may not be admissible according to the problem statement, for example a recovered
matrix may not be orthogonal or some internal constraints on the matrix’ entries may be
violated. Such constraints have to be enforced explicitly, while often the original geometric
algebra formulation implicitly precluded such solutions.

2.6. Relation to Differential Geometry

Focusing on the even unit versors of a given geometric algebra $G(V)$, we can treat them as
a representation of the special orthogonal group of the underlying vector space $V$, that is,
the transformation group which preserves the inner product. This transformation group is a
Lie group [DHSvA93] which, in turn, can be viewed as a differentiable manifold, forming a
bridge to the field of differential geometry. We would like to spend a few words elucidating
this relationship.

There, differential 1-forms are defined, linear functions mapping vectors from $V$ to real
numbers. It is well known that differential 1-forms form a vector space, the so called dual
space $V^*$ to the vector space $V$ upon which they act.

If $\varphi \in V^*$ is a differential 1-form and $v \in V$ a vector, one finds many different notations
for the action of $\varphi$ on $v$. Since $\varphi$ is a linear function, a common notation is $\varphi(v)$. Physics
texts often use the bracket notation $\langle \varphi, v \rangle$ or $\langle \varphi | v \rangle$, sometimes denoting $\varphi$ by $\langle \varphi \rangle$ and $v$ by $|v\rangle$
in order to keep track of the space of which they are elements. Confusingly, $\langle .. \rangle$ is also the
notation used for the inner product of a space with a bilinear form in some texts.

Differential geometry also defines the exterior product (or “wedge product”) between differ-
etial forms. The wedge product between two 1-forms, $\varphi$ and $\psi$, results in a differential 2-form
\( \varphi \wedge \psi \), which linearly maps two vectors to a scalar. That is, if \( \varphi, \psi \in V^* \) and \( v, w \in V \), then 
\[
(\varphi \wedge \psi)(v, w) = \varphi(v)\psi(w) - \varphi(w)\psi(v) \in \mathbb{R}.
\]
This can be extended to arbitrary differential \( r \)-forms.

The properties of the wedge product between differential forms match the behavior of the outer product introduced between vectors in section 2.1. Together with the properties of the inner product, this leads to the fact that, in purely Euclidean as well as in conformal geometric algebras, the vector basis of \( V \) can be identified with the dual basis of \( V^* \), the action of the differential 1-forms being given by the inner product between vectors [HS84].

When [HS84] introduces the notion of reciprocal vectors, this identification of \( V \) and \( V^* \) happens implicitly. The reciprocal vector \( e^j \in V \) to the vector \( e_j \in V \) is the unique vector for which \( e_j \cdot e_i = \delta^j_i \), the Kronecker delta, for all vectors \( e_i \) in the basis of \( V \). Thus, the vector \( e^j \) takes the place of the basic differential 1-form \( \varphi^j \in V^* \) for which \( \langle \varphi^j, e_i \rangle = \delta^j_i \). In that respect the outer product used in geometric algebra is not (just) an abuse of notation, the similarity with the outer product between differential forms not coincidental. It is the (implicit) identification of the basis of a vector space with the basis of its dual that reconciles the two frameworks.

That being said, we turn our attention to tangent spaces to the manifold \( M \) which is the orthogonal group of \( V \). It should be noted here that the orthogonal group is usually not connected, i.e. consist of more than one connected component. Considering the component \( M^0 \) which contains the identity, the tangent space at the identity \( T_1M^0 \) is also called the Lie algebra of \( M^0 \) and can be represented as a pure bivector algebra [DHSvA93], i.e. the sub-algebra of elements of the geometric algebra which can be written as a sum of 2-blades. Every element of the Lie algebra of \( M^0 \) can be mapped to an even unit versor (and vice versa) by the exponential map (respectively, the logarithmic map). In terms of geometric algebra, the exponential map of a pure bivector \( B \) — as well as of any multivector — can be calculated by the power series [HS84]
\[
\exp(B) = \sum_{k=0}^{\infty} \frac{B^k}{k!},
\]
where \( B^k \) means the \( k \)-fold geometric product of \( B \) with itself.

The union over all tangent spaces, \( TM = \bigcup_S T_SM \), is called the tangent bundle of the manifold \( M \) and can be identified with the full geometric algebra \( \mathcal{G}(V) \) over \( V \).

In differential geometry, a metric is the assignment of an inner product to the tangent space \( T_pM \) at every point \( p \in M \) of the manifold \( M \) — as opposed to a distance metric, which is a measure between two points and fulfills the metric axioms, such as commutativity and the triangle inequality. A metric is often used to define a notion of lengths and angles of intersection.

The properties of geometric algebra and, in particular, CGA that are beneficial for the representation of geometric objects and transformations appear from a differential geometry standpoint as follows. For a geometric algebra the inner product introduced in section 2.1 works on arbitrary elements and thus is a global metric on the whole tangent bundle. In section 2.4 we explained how it can be used to define incidence relationships between different geometric
objects. The fact that the action of the orthogonal group preserves the inner and outer product means that the versor product acts on geometric objects in a structure preserving manner. More specifically, hyperspheres are mapped to hyperspheres and incidence relationships as well as local angles of intersection are preserved.

2.7. Summary

We have explained that by suitably choosing an embedding function the conformal model of geometric algebra is capable of representing points in Euclidean space by null vectors of a higher-dimensional Minkowski space. The properties of the inner product with which the underlying Minkowski space is endowed ensure that the inner product between unit weight conformal points is proportional to the squared Euclidean distance between the represented Euclidean points. Moreover, the inner product can be used to define the representation of a number of geometric objects, such as hyperspheres and hyperplanes in Euclidean space. The inner product is also used to define a notion of incidence and angles of intersection between these objects.

We have demonstrated how versors are used to represent orthogonal transformations, which preserve the inner product. Moreover, they also preserve the structure of the outer product by acting as an outermorphism. The outer product is used to define more complicated geometric objects such as tangents and circles, which are therefore also preserved as respective classes by the versor product.

The Lie group structure of the group of even geometric algebra versors and the fact that its Lie algebra is representable as an algebra of bivectors allows a minimal parametrization of the special orthogonal transformations as bivector exponentials. All of these facts make geometric algebra suitable for representing geometric objects and transformations in Euclidean spaces. Geometric algebra extends classical linear algebra in a natural and powerful way. This makes it a desirable interface between geometric intuition used to describe and reason about geometric entities at one end and the mathematical methods used to utilize the algebraic description of those entities on the other end.
3

Determining Conformal Transformations from Minimal Correspondence Data

In this chapter we will consider the two main aspects that extend CGA beyond purely Euclidean spaces, namely the additional “book-keeping” dimensions and the power to represent conformal transformations. We derive a method to determine a conformal transformation in \( n \)-dimensional Euclidean space from geometrical data alone, treating the additional representational dimensions as unknowable. The solution is derived from exact correspondences between data and is presented in closed form. We show that a minimal dataset needed for correspondence is a localized vector frame and an additional point.

Determining a conformal transformation from correspondence data is difficult in CGA, because of the non-commutativity of the geometric product in conjunction with the double-sidedness of the versor product which performs the transformation on geometric objects. In order to obtain a solution we use the representation of CGA by extended Vahlen matrices. These are \( 2 \times 2 \)-matrices with entries from purely Euclidean geometric algebra (the Clifford algebra of \( \mathbb{R}^n \)). This ultimately reduces the problem to the determination of a Euclidean orthogonal transformation from given vector correspondences, for which solutions are known. We give a closed form solution for the general case of conformal (respectively anti-conformal) transformations, which preserve (respectively reverse) angles locally. We also treat the impor-
tant special case when it is known that the conformal transformation is a rigid body motion — also known as a Euclidean transformation — which additionally preserves Euclidean distances.

### 3.1. Introduction

General conformal transformations have applications in physics and engineering. For example, functions that are governed by a potential are still governed by a potential after a conformal transformation. Therefore, problems in fluid dynamics, electromagnetic or gravitational theory with complicated geometric conditions can be transformed to well-known geometric configurations which are more manageable. Moreover, conformal transformations preserve hyperspheres (i.e. hyperspheres are mapped to hyperspheres), as well as intersection and incidence relationships. We will use this fact later to determine sufficient correspondence data from Euclidean geometric configurations.

Vahlen matrices are known matrix representations of conformal transformations acting by fractional linear transformations. Historically, Vahlen matrices were introduced as complex valued $2 \times 2$ matrices acting on a vector representation of complex numbers. After the introduction of the invertible geometric product, they could be generalized to $2 \times 2$-matrices with entries from an $n$-dimensional Euclidean geometric algebra with some additional algebraic constraints. They could act on other Vahlen matrices by matrix multiplication with the multiplication between individual entries being realized by the geometric product. Vahlen matrices and CGA versors are different representations of the same group. In fact, there exists an isomorphism from one representation to the other. Just like versors, Vahlen matrices act on the representation of geometric objects by a double-sided product. The geometric objects, in that case are represented by extended Vahlen matrices [Dor09]. These are still $2 \times 2$-matrices with entries from an $n$-dimensional Euclidean geometric algebra, but with the algebraic constraints pertaining to Vahlen matrices relaxed. There exists an isomorphism from CGA to extended Vahlen matrices.

The representation of CGA by extended Vahlen matrices has some advantages over the representation on a (multi-)vector basis as introduced in chapter 2. In the extended Vahlen matrix representation the closed form formulas (such as (2.35)) are decomposed into a number of individual, algebraically simpler, calculations. Moreover, these individual calculations are restricted to purely Euclidean geometric algebra, which means that they do not require the Minkowski metric of the CGA base space and therefore are easier to implement. This simplification is essential to our approach for solving correspondence equations for the parameters of the versor representing the common conformal transformation between correspondence data. Furthermore, the representation by extended Vahlen matrices conveniently allows us to disregard representational aspects, such as the weight of conformal points, and focus on geometrical data, such as the location of conformal points in Euclidean space. In practice, geometrical information is available as the result of some measurement or design process, while representational data is often unknown or considered irrelevant.

In this chapter we use the structuring power of Vahlen matrices to derive a closed form solu-
tion for an even conformal versor (the geometric algebra representation of a general conformal transformation of Euclidean space) given a minimal set of exact correspondence data. The method introduced here finds this transformation in closed form and does so with little computational effort. In order to derive this solution we will draw upon a method given in [FD11], which recovers an orthogonal transformation between two sets of fully known corresponding vectors.

Since the given correspondences have to be exact, the solution is suitable for those applications in physics, robotics, computer graphics, differential geometry etc., where a pair of start and target configurations is known precisely and the conformal transformation between the two has to be found.

If the correspondence data is not exact or more than minimal data is used, our method can still find a solution. In those cases some steps have to be approximated using various workarounds [RFC10, Per08, FD11] or heuristics. The impact of these approximations on the overall solution is difficult to quantify or interpret geometrically. Moreover, none of these method alone can recover a conformal transformation from geometric data. Either they need to know the representational dimensions (i.e. the weights of corresponding points) or they are restricted to orthogonal transformations, e.g. rotations about the origin in the vector space under consideration. A more detailed discussion follows in section 3.3.1.

The structure of this chapter is as follows. Section 3.2 shows how to represent the conformal model of geometric algebra by extended Vahlen matrices. In section 3.3 we consider different kinds of input data and explain why a localized vector frame and an additional point in Euclidean space are minimal data for recovering a conformal transformation. Here, we also present in detail our closed form solution for this conformal transformation. We generalize our method to anti-conformal transformations in section 3.4, where we also show a way to obtain a minimal set of correspondence data from Euclidean point correspondences and present the special case of rigid body motions. We briefly consider the case of non-exact correspondence data. Concluding remarks are found in section 3.5.

3.2. Representation of Geometric Objects and Transformations by Extended Vahlen Matrices

In chapter 2 we introduced geometric algebra as a mathematical structure defined by properties of its elements and operations. We mentioned that geometric algebra allows for coordinate free representation of geometric objects and transformations and then proceeded to introduce the conformal model of geometric algebra using a particular basis for the vector space \( \mathbb{R}^{n+1,1} \) over which CGA is constructed.

Now we introduce a different way of representing this vector basis — and therefore all elements of CGA. There is an isomorphism between the (multi-)vector basis representation of \( \mathcal{G}(\mathbb{R}^{n+1,1}) \) introduced in section 2.4 and a representation by \( 2 \times 2 \)-matrices with entries from \( \mathcal{G}(\mathbb{R}^n) \), the geometric algebra over the \( n \)-dimensional Euclidean space. We introduce this representation and show how concepts from chapter 2 extend to it, only in so far as they
contribute to the main concern of finding a conformal transformation from correspondence data. All proofs pertaining to Vahlen matrices are omitted here and can be found, for example, in [Lou01, HJ03]. For a more extensive introduction to extended Vahlen matrices, we refer the reader to [Dor09].

We use the symbol $\equiv$ to denote the equivalence between a (multi-)vector and its matrix representation. The null basis introduced in section 2.4 can be represented by

$$e_i \equiv \begin{pmatrix} e_i & 0 \\ 0 & -e_i \end{pmatrix},$$

(3.1)

$$n_0 \equiv \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix},$$

(3.2)

$$n_\infty \equiv \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}.$$  

(3.3)

The geometric product is then implemented by matrix multiplication between the representations, where the non-commutative geometric product of $\mathcal{G}(\mathbb{R}^n)$ is employed between matrix entries. The inner and outer product arise from (2.14) and (2.15). Note that the neutral elements of addition and multiplication are no longer the scalar quantities 0 and 1, but have matrix representations, too, namely the zero matrix and the identity matrix, respectively.

It is easy to verify basic properties, such as

$$e_i^2 \equiv \begin{pmatrix} e_i & 0 \\ 0 & -e_i \end{pmatrix} \begin{pmatrix} e_i & 0 \\ 0 & -e_i \end{pmatrix}$$

$$= \begin{pmatrix} e_i^2 & 0 \\ 0 & e_i^2 \end{pmatrix},$$

$$= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

$$\equiv 1,$$  

(3.4)

$$e_i \cdot n_0 = \frac{1}{2} (e_in_0 + n_0e_i) $$

$$= \frac{1}{2} \left( \begin{pmatrix} e_i & 0 \\ 0 & -e_i \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} e_i & 0 \\ 0 & -e_i \end{pmatrix} \right)$$

$$= \frac{1}{2} \left( \begin{pmatrix} 0 & 0 \\ -e_i & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ e_i & 0 \end{pmatrix} \right)$$

$$= \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix},$$

$$\equiv 0,$$  

(3.5)
\[ n_0 \cdot n_\infty = \frac{1}{2} (n_0 n_\infty + n_\infty n_0) \]
\[ \equiv \frac{1}{2} \left( \begin{array}{cc} 0 & 0 \\ 1 & 0 \end{array} \right) \left( \begin{array}{cc} 0 & -2 \\ 0 & 0 \end{array} \right) + \left( \begin{array}{cc} 0 & -2 \\ 0 & 0 \end{array} \right) \left( \begin{array}{cc} 0 & 0 \\ 1 & 0 \end{array} \right) \]
\[ = \frac{1}{2} \left( \begin{array}{cc} 0 & 0 \\ 0 & -2 \end{array} \right) + \left( \begin{array}{cc} -2 & 0 \\ 0 & 0 \end{array} \right) \]
\[ = \left( \begin{array}{cc} -1 & 0 \\ 0 & -1 \end{array} \right) \]
\[ \equiv -1. \quad (3.6) \]

The advantage in calculations with this representation lies in the fact that the isotropic elements of the null basis, \( n_0 \) and \( n_\infty \), never appear explicitly. The structure of the matrix representation realizes the Minkowski metric and calculations between elements are carried out in the geometric algebra over an \( n \)-dimensional Euclidean vector space only. So, even though a multivector may have a non-invertible matrix representation, explicit calculations on an element level never involve null vectors.

The matrix representation of a finite conformal point \( p \) is given by (2.45)
\[ p = \alpha \left( n_0 + p + \frac{1}{2} p^2 n_\infty \right) \]
\[ \equiv P = \alpha \left( \begin{array}{cc} p & -p^2 \\ 1 & -p \end{array} \right), \quad (3.7) \]

that of a conformal tangent \( t \) by (2.54)
\[ t = -p \cdot (p \wedge t \wedge n_\infty) \]
\[ \equiv T = \left( \begin{array}{cc} pt & -ptp \\ t & -tp \end{array} \right). \quad (3.8) \]

As a subset of the set of general conformal multivectors, versors play a special role. They represent both conformal transformations of the Euclidean space \( \mathbb{R}^n \) as well as orthogonal transformations of the Minkowski space \( \mathbb{R}^{n+1,1} \). As we saw in section 2.3, versors are the product of a number of invertible conformal vectors and have a special matrix representation called Vahlen matrices. In general, we will call a matrix \( S = \left( \begin{array}{cc} a & b \\ c & d \end{array} \right) \) with entries from Euclidean geometric algebra, \( a, b, c, d \in G(\mathbb{R}^n) \), a Vahlen matrix, if it fulfills the conditions [Lou01, HJ03]
\[ a, b, c, d \in \Gamma(\mathbb{R}^n) \cup \{0\}, \quad (3.9) \]
\[ \widehat{ab}, \widehat{bd}, \widehat{dc}, \widehat{ca} \in \mathbb{R}^n, \quad (3.10) \]
\[ \Delta(S) = \widehat{ad} - \widehat{bc} \in \mathbb{R} \setminus \{0\}, \quad (3.11) \]
Determining Conformal Transformations from Minimal Correspondence Data

where $\Gamma(\mathbb{R}^n)$ denotes the group of Euclidean versors in $\mathcal{G}(\mathbb{R}^n)$, also called the Clifford group of the vector space $\mathbb{R}^n$, i.e. [Por95]

$$\Gamma(V) = \left\{ S \in \mathcal{G}(V) | S^{-1} \text{ exists and } SV\hat{S}^{-1} \in V \right\}.$$ (3.12)

A derived property is that in a Vahlen matrix representing an even (respectively odd) versor, the diagonal entries are even (respectively odd) Euclidean versors, while the off-diagonal entries are odd (respectively even) Euclidean versors. These properties can all be proven [Por95] from the definition of a versor as the product of invertible CGA vectors.

The concepts of grade involution (2.22) and Clifford conjugation (2.23) extend to Vahlen matrices as follows [Dor09].

$$\hat{S} = \begin{pmatrix} \hat{a} & -\hat{b} \\ -\hat{c} & \hat{d} \end{pmatrix}$$ (3.13)

$$S = \begin{pmatrix} \hat{d} & -\hat{b} \\ -\hat{c} & \hat{a} \end{pmatrix}$$ (3.14)

A Vahlen matrix is always invertible with

$$S^{-1} = \frac{1}{\Delta(S)} S.$$ (3.15)

Note that (3.7) and (3.8) are not Vahlen matrices, since they fail condition (3.11). As matrices representing null blades they are not invertible.

Now let us assume an element $X \in \mathcal{G}(\mathbb{R}^{n+1,1})$ in its matrix representation. Vahlen matrix $S$ acts on it via the twisted adjoint action

$$X' = SX\hat{S}^{-1} = \frac{1}{\Delta(S)} SX\hat{S}.$$ (3.16)

Spelling out the action of a Vahlen matrix on a conformal point (3.7) and a conformal tangent (3.8), respectively, we find that it preserves the type of object represented and we get

$$SP\hat{S}^{-1} = \frac{(cp + d)(cp + d)}{\Delta(S)} \begin{pmatrix} p' & -p'^2 \\ 1 & -p' \end{pmatrix},$$ (3.17)

$$ST\hat{S}^{-1} = \frac{1}{\Delta(S)} \begin{pmatrix} p't' & -p't'p' \\ t' & -t'p' \end{pmatrix},$$ (3.18)

where

$$p' = (ap + b)(cp + d)^{-1},$$ (3.19)

$$t' = (cp + d)t(cp + d),$$ (3.20)
if \((cp + d) \neq 0\), that is if \((cp + d)^{-1}\) exists. Otherwise we find that

\[
SP\hat{S}^{-1} = \begin{pmatrix} 0 & -2 \\ 0 & 0 \end{pmatrix}, \tag{3.21}
\]

\[
ST\hat{S}^{-1} = \frac{1}{\Delta(S)} \begin{pmatrix} 0 & -2t' \\ 0 & 0 \end{pmatrix}, \tag{3.22}
\]

where now

\[
(cp + d) = 0. \tag{3.23}
\]

\[
t' = (ap + b)t(ap + b), \tag{3.24}
\]

The matrices (3.22) and (3.21) represent a conformal tangent anchored at \(n_\infty\) (also called a
direction vector in section 2.4 or a free vector in [DFM07]) and the conformal point at infinity,
\(n_\infty\), respectively.

Representing a versor by a Vahlen matrix facilitates both design and interpretation of the transformation performed by it. Any conformal transformation can be written as the composition of a translation, a rotation, uniform scaling and a special conformal transformation (also called a transversion [DFM07]). The elemental conformal transformations in \(n\)-dimensional Euclidean space are represented as follows [Lou01].

- Translation along a Euclidean vector \(t\)

\[
S = \begin{pmatrix} 1 & t \\ 0 & 1 \end{pmatrix}, \tag{3.25}
\]

- Rotation by Euclidean rotator \(R\) (see section 2.3)

\[
S = \begin{pmatrix} R & 0 \\ 0 & R \end{pmatrix} \tag{3.26}
\]

- Uniform scaling by factor \(\exp(\gamma), \gamma \in \mathbb{R}\)

\[
S = \begin{pmatrix} \exp(\gamma/2) & 0 \\ 0 & \exp(-\gamma/2) \end{pmatrix} \tag{3.27}
\]

- Transversion parametrized by Euclidean vector \(v\)

\[
S = \begin{pmatrix} 1 & 0 \\ v & 1 \end{pmatrix} \tag{3.28}
\]

The first three transformations represent similarity transformations in Euclidean space, which preserve lines but may alter Euclidean distances uniformly by a scalar factor. Geometrically, a transversion can be seen as a hypersphere inversion followed by a translation and another
Determining Conformal Transformations from Minimal Correspondence Data

hypersphere inversion or, alternatively, as the successive inversion in two hyperspheres of equal radius touching in a single point. Either way, in \( n \)-dimensional Euclidean space a transversion has \( n \) degrees of freedom, which is why it can be parametrized by a single Euclidean vector. This transformation — while preserving angles locally — can transform lines in Euclidean space into circles and vice versa.

These simple structures make it easy to predict the interaction between basic transformations or analyze a composite transformation. Moreover, this representation facilitates the interpretation of the algebraic result of a transformation in terms of its effects on the Euclidean parts of a geometric object represented in CGA. Note that these elementary transformations trivially fulfill the Vahlen matrix conditions (3.9), (3.10) and (3.11).

We would like to remark that the scaling and the transversion act on an object by changing its weight. The weight is a homogeneous degree of freedom. In order to interpret a representative element as a geometric object, it has to be normalized, the weight “divided out”. The effect on a transformed object in Euclidean space manifests fully only after this interpretation step. As a simple example, we will consider the action of a scaling versor (3.27) on a conformal point (3.7).

\[
\frac{1}{\Delta(S)} S P \hat{S}^{-1} = \begin{pmatrix} p & -\exp(\gamma)p^2 \\ \exp(-\gamma) & -p \end{pmatrix} 
= \exp(-\gamma) \begin{pmatrix} \exp(\gamma)p & -(\exp(\gamma)p)^2 \\ 1 & -\exp(\gamma)p \end{pmatrix}
\]

(3.29) (3.30)

We see that the absolute Euclidean coordinates of the (matrix representation of the) conformal vector (3.29) are left unchanged. However, changing the \( n_0 \) and \( n_\infty \)-coordinate (i.e. the bottom left and top right matrix entries) changes the interpretation (3.30) of the resulting (matrix representation of the) conformal vector. The rewriting as in (3.30) shows that the matrix actually represents the point at Euclidean location \( \exp(\gamma)p \) with weight \( \exp(-\gamma) \).

3.3. Exact Conformal Transformations and Minimal Data

Geometrically, a conformal transformation of \( n \)-dimensional Euclidean space can be seen as a series of inversions in Euclidean hyperspheres (see section 2.4). Since every such inversion preserves the magnitude of angles locally, but reverses their handedness, only an even number of hypersphere inversions performs a conformal transformation representable as an even versor in CGA. An odd number of hypersphere inversions performs an angle reversing anti-conformal transformation. In the following, without loss of generality, we will restrict our considerations to even versors and conformal transformations and only briefly return to anti-conformal transformations in section 3.4.1.
3.3 Exact Conformal Transformations and Minimal Data

3.3.1. Related Methods

Before we consider the problem of finding a conformal transformation from correspondence data, we need to address the question of what kind of input data is available. Different kinds of input data lead to different approaches.

The probably most straightforward and most efficient method for finding the orthogonal transformation between two corresponding sets of vectors can be found in [FD11]. It constructs a composite orthogonal transformation by successively aligning vectors by a reflection in the respective mid-plane between corresponding vectors. (For more details on the generalized concept of mid-planes, please refer to section 2.4.) Assuming that two sets of input vectors \{x_i\} and \{x'_i\} are indeed related by a common orthogonal transformation (i.e. a transformation that preserves the inner product between vectors), successive reflections do not change previously aligned vectors. In that respect the input data needs to be exact. Figure 3.1 visualizes the method given in [FD11].

![Figure 3.1](image)

**Figure 3.1:** A sequence of reflections in respective mid-planes of corresponding vectors successively aligns these vectors. As an example, here the vectors are conformal vectors representing unit weight conformal points. The second reflection in s2 does not change the already aligned vector \(x'_1\) when \{x_i\} and \{x'_i\} are related by an orthogonal transformation, i.e. \(x_i \cdot x_j = x'_i \cdot x'_j\) for all \(i, j\). For details see [FD11].

The method generalizes to higher-dimensional spaces and spaces with more complicated metrics, in particular Minkowski spaces. However, technically, it recovers an assumed orthogonal transformation between corresponding vectors. By the representational power of CGA, orthogonal transformations of the Minkowski space \(R^{n+1,1}\) induce conformal transformations on the underlying Euclidean space \(R^n\). The main characteristic that sets the method from [FD11] apart from the method introduced in the present chapter is that it needs vector correspondences to be known in the space on which the orthogonal transformation acts. More specifically, it is not enough to know correspondences between geometrical objects, but the concrete conformal representation vector needs to be known — including the magnitude of its weight, or the homogeneous coordinate.

An alternative method is presented in [Per08], which is based on finding a least squares solu-
tion to a system of $2^n$ simultaneous linear equations. The input data need not be exact for this method to produce a result. Possible noise or uncertainty is accounted for by the least squares criterion. However, the method does not guarantee to return a versor. Rather, the versor conditions need to be enforced by additional constraints. Moreover, the least squares criterion is imposed on the $2^n$ components of the involved multivectors rather than on geometrically meaningful quantities, which makes it difficult to evaluate the geometric interpretation of the apparent optimality of that method. Additionally, the weights of the corresponding conformal (multi-)vectors need to be known.

A third method [RFC10] reconstructs an assumed rotation (i.e. an orthogonal transformation) between corresponding vectors by finding a set of mutually orthogonal two-dimensional rotation planes, each being determined by a pair of chords between corresponding vectors. In order to obtain these planes from the chords the authors employ singular value decomposition (SVD), which makes their method viable for non-exact and non-minimal correspondence data. Just like the other two methods it requires full vector correspondences with the weights known. Also, this method performs poorly in special cases termed (pseudo-)isoclinic rotations. These rotations are characterized by the fact that two or more of the recovered rotations in the mutually orthogonal planes have the same rotation angle.

### 3.3.2. Input Data Requirements

In this chapter we focus on geometric configurations in Euclidean $n$-dimensional space. In applications this is often the only information available, because Euclidean properties can be measured, estimated or designed, while representational properties (such as the magnitude of homogeneous coordinates, e.g. interpretable as point weights) may be inaccessible or irrelevant.

In order to answer the question what information is needed to specify a conformal transformation of Euclidean space, we resort to a counting argument for the number of degrees of freedom of an even versor. It has been shown [HS84] that every even versor in the geometric algebra over a Euclidean or a Minkowski space can be written as the exponential of a pure bivector (i.e. a multivector containing only components of grade 2), $S = \exp(b)$, with $\langle b \rangle_2 = b$. Conversely, the exponential of any pure bivector results in an even versor. Note that in general $b$ is not a 2-blade, but may be written as the sum of 2-blades. The number of degrees of freedom of a pure bivector in a given CGA $\mathcal{G}(\mathbb{R}^{n+1,1})$ is therefore equal to the number of linearly independent 2-blades in that CGA.

\[ \text{#DOF of an even versor} = \binom{n+2}{2} \quad (3.31) \]

This limits the choice of exact correspondence data to specify a conformal transformation. To explain our basic method we take a suitably chosen minimal set of data as follows.

We start out with a Euclidean vector frame $\{ \vec{t}_i \}, i = 1, \ldots, n$. Without loss of generality we can assume it to be orthonormal (indicated by the check over the vectors), because every non-orthogonal frame can be orthonormalized using Gram-Schmidt orthogonalization in Euclidean
space $\mathbb{R}^n$. Now, the first direction vector provides $(n-1)$ degrees of freedom, because — since it is normalized — only its direction matters, but not its scale. The second direction vector provides only $(n-2)$ degrees of freedom, because it has to be orthogonal to the first one and it is normalized, too. Proceeding like this, the last direction vector is completely determined by the former ones because of the orthonormalization assumption we made. It does not provide any degrees of freedom. One may argue that the handedness of the frame provides another degree of freedom, but this is not so. Since the handedness is preserved by a conformal transformation (as opposed to an anti-conformal transformation), it does not matter which one is picked.

By taking this frame and anchoring it at a given point, the frame’s Euclidean location in space, $p_1$, clearly provides us with $n$ more degrees of freedom.

We know that conformal transformations preserve the mutual angles between the frame vectors, but that uniform scaling is a valid conformal transformation. All vectors of one frame can be assumed to have a common length, $\sigma$, i.e. $t_i = \sigma \tilde{t}_i$, $i = 1, \ldots, n$, which accounts for one more degree of freedom.

In addition to the localized frame, we specify another Euclidean point, $p_2$, providing an additional $n$ degrees of freedom. This makes for a total of

$$\#\text{DOF} = (n-1) + \cdots + 1 + n + 1 + n$$

$$= 1 + n + \frac{n(n+1)}{2}$$

$$= \frac{(n+2)(n+1)}{2}$$

$$= \left( \frac{n+2}{2} \right), \quad (3.32)$$

which is precisely the number of degrees of freedom accounted for by an even CGA versor and therefore by a conformal transformation of $n$-dimensional Euclidean space.

In summary, to specify a conformal transformation in Euclidean $n$-space, as data we assume a frame made up of $n$ orthogonal direction vectors of a common length, $t_1, t_2, \ldots, t_n$, anchored at a common point, $p_1$, and an additional point $p_2$, all being mapped to another such orthogonal common length frame, $t'_1, t'_2, \ldots, t'_n$, anchored at a common point, $p'_1$, and an additional point $p'_2$. See Figure 3.2 for an example in 3-dimensional Euclidean space.

**Figure 3.2.** A frame of three direction vectors with a common scale, anchored at a common point plus an additional point constitute minimal correspondence data for a conformal mapping in 3-dimensional Euclidean space. Note that a conformal mapping can rotate and uniformly scale the frame, locally, but it cannot change the mutual angles between the frame vectors.
Finally, besides the correspondences \( p_1 \leftrightarrow p_1', p_2 \leftrightarrow p_2'\) and \( t_i \leftrightarrow t_i' \) to be known, we require the mutual angles between the vectors \( t_i \) and between the vectors \( t_i' \) to be the same. In that sense the correspondence data has to be \textit{exact}. Given this prerequisite, the above correspondences determine a conformal transformation precisely and any conformal transformation is determined by such a set of correspondences.

Note that, even though we will use the conformal model of geometric algebra to represent the correspondences, the input data only consists of purely Euclidean quantities. We are therefore able to determine a conformal transformation on purely geometric grounds. No abstract algebraic or representational characteristics enter our requirements. More specifically, unlike in the methods discussed in section 3.3.1 the weights of the conformal points \( p_1 \) and \( p_2 \) do not have to be known.

3.3.3. Solution

We represent the additional Euclidean point \( p_2 \) by a conformal point in matrix representation as in (3.7) and the localized frame \( \{ t_i \} \) by a number of conformal tangents as in (3.8), anchored at a common Euclidean point \( p_1 \). Then, in order to recover the conformal transformation, we would have to solve a set of simultaneous equations (3.16) for the common even versor \( S \).

Whether we use the matrix representation or the vector basis representation, it is not trivial to solve these equations, because of the non-commutative nature of the geometric product and because of interactions between the different multivectors involved. In order to reduce the complexity of the equations to solve we will introduce a set of \textit{transfer objects}, which have very simple matrix representations. Then we find the Vahlen matrices \( S_j = \begin{pmatrix} a_j & b_j \\ c_j & d_j \end{pmatrix} \), \( j = 1, 2 \) which take each of the corresponding sets of input data, \( \{ p_1, p_2, t_1, \ldots, t_n \} \) and \( \{ p_1', p_2', t_1', \ldots, t_n' \} \), respectively, to these transfer objects. Finally, we recover the conformal transformation that maps \( \{ p_1, p_2, t_1, \ldots, t_n \} \) to \( \{ p_1', p_2', t_1', \ldots, t_n' \} \) as \( S = S_2^{-1} S_1 \).

In particular, we introduce the following transfer objects,

\[
0 T_i = \begin{pmatrix} 0 & 0 \\ \hat{0}_{t_i} & 0 \end{pmatrix}, \quad i = 1, \ldots, n \tag{3.33}
\]

\[
\infty P = n_\infty = \begin{pmatrix} 0 & -2 \\ 0 & 0 \end{pmatrix}, \tag{3.34}
\]

i.e. an orthonormal localized frame at the origin with directions \( 0 \hat{t}_i \) and the point at infinity. Note that, in addition to being orthonormal, we assume that the frame \( \{ 0 \hat{t}_i \} \) has the same handedness as \( \{ t_i \} \) and \( \{ t'_i \} \). The handedness of a frame \( \{ t_i \} \) is computed as the sign of the outer product of its vectors relative to the chosen Euclidean pseudoscalar, \( (t_1 \wedge \cdots \wedge t_n) I_n^{-1} \), where \( I_n = e_1 \wedge e_2 \wedge \cdots \wedge e_n \), so the assumption is easy to check and assert.

Since \( S \) represents an even versor, either \( S_1 \) and \( S_2 \) both represent even versors or both represent odd versors. Because we assumed equal handedness of the original frame, the target

\[3\text{This idea is similar to the "projective frame" technique using homogeneous coordinates to calculate Möbius transformations in the 2D (complex) plane.}\]
Now, we will find a matrix $S_1$ that maps the original data to the transfer objects. Step by step we will make sure that this matrix is a Vahlen matrix. Consider $S_1$ acting on a conformal tangent (3.8) via the twisted adjoint action (3.18). Since the location of the tangents $p_1$ is mapped to the conformal point at the origin by $S_1$, (3.19) implies that $a_1 p_1 + b_1 = 0$, so that

$$b_1 = -a_1 p_1.$$  
(3.35)

Furthermore, because $S_1$ maps the additional point $p_2$ to the conformal point at infinity, (3.23) implies that $c_1 p_2 + d_1 = 0$, and therefore

$$d_1 = -c_1 p_2.$$  
(3.36)

Next, we ensure that $S_1$ is a Vahlen matrix by asserting conditions (3.9), (3.10) and (3.11).

We rewrite the general condition (3.11) for our specific matrix $S_1$ to

$$\mathbb{R} \setminus \{0\} \ni \Delta(S_1) = a_1 \hat{d}_1 - b_1 \hat{c}_1$$

$$= -a_1 \hat{c}_1 \hat{p}_2 + a_1 \hat{p}_1 \hat{c}_1$$

$$= a_1 \hat{p}_1 \hat{c}_1 - a_1 \hat{p}_2 \hat{c}_1$$

$$= a_1 (\hat{p}_1 - \hat{p}_2) \hat{c}_1.$$  
(3.37)

It follows that

$$a_1^{-1} = \frac{1}{\Delta(S_1)} (p_1 - p_2) \hat{c}_1.$$  
(3.38)

Note that this is not a proof that this condition holds. It will allow us to choose a scalar $\Delta(S_1)$ to establish the relationship between $a_1$ and $c_1$ and ensure that the matrix we will later find for $S_1$ is indeed a Vahlen matrix. The simplest choice would be for example $\Delta(S_1) = 1$, yielding a Vahlen matrix $S_1$ which represents an even unit versor.

By (3.37), conditions (3.10) are fulfilled, as we see when we write out the four individual
equations.

\[
\begin{align*}
\hat{a}_1 \hat{b}_1 &= -a_1 \hat{a}_1 \hat{p}_1 \quad \in \mathbb{R}^n, \quad (3.39) \\
\hat{b}_1 d_1 &= \hat{a}_1 \hat{p}_1 c_1 \hat{p}_2 \\
&= \hat{p}_1 \hat{a}_1 c_1 \hat{p}_2 \\
&= \hat{p}_1 \hat{a}_1 a_1^{-1} c_1 \hat{p}_2 \\
&= \frac{1}{\Delta(S_1)} (p_1 - p_2) \hat{c}_1 c_1 \hat{p}_2 \\
&= \frac{\hat{a}_1 \hat{a}_1 \hat{c}_1}{\Delta(S_1)} (p_1^2 - p_2^2) \hat{c}_1 c_1 \hat{p}_2 \\
&= \Delta(S_1) (p_1 - p_2) \hat{c}_1 c_1 \hat{p}_2 \quad \in \mathbb{R}^n, \quad (3.40)
\end{align*}
\]

\[
\begin{align*}
d_1 \hat{c}_1 &= -c_1 p_2 \hat{c}_1 \quad \in \mathbb{R}^n, \quad (3.41)
\end{align*}
\]

\[
\begin{align*}
\hat{c}_1 a_1 &= \Delta(S_1) (p_1 - p_2)^{-1} \quad \in \mathbb{R}^n, \quad (3.42)
\end{align*}
\]

where we used (3.38), as well as the fact that \( \hat{s} s \in \mathbb{R} \), for any \( s \) that is the geometric product of any number of vectors. Recall that the latter fact is (2.36), and follows from the (anti)automorphism properties of the Clifford conjugation and the grade involution, as well as axiom (2.10) of the geometric product.

So far, our matrix \( S_1 \) is of the form

\[
S_1 = \begin{pmatrix}
a_1 & -a_1 p_1 \\
c_1 & -c_1 p_2
\end{pmatrix},
\]

which, together with (3.38), leaves \( c_1 \) as the only unknown. In order for \( S_1 \) to be a Vahlen matrix representing an (even) versor, it needs to fulfill condition (3.9). That is, \( a_1 \) and \( c_1 \) have to be (even, respectively odd) Euclidean versors.

We can find a solution using the transformation of the Euclidean frame vectors \( \hat{t}_i \) to the transfer frame (3.20), which is equivalent to

\[
\begin{align*}
o \hat{t}_i &= (c_1 p_1 + d_1) \frac{t_i (c_1 p_1 + d_1)}{c_1 (p_1 - p_2)} \\
&= (c_1 p_1 - c_1 p_2) \frac{t_i (c_1 p_1 - c_1 p_2)}{c_1 (p_1 - p_2)} \\
&= c_1 (p_1 - p_2) t_i c_1 (p_1 - p_2).
\end{align*}
\]

(3.44)

Before we proceed to the solution we would like to remark that for any versor \( S = \prod_{i=1}^{k} s_i = s_1 s_2 \ldots s_k, \ k \in \mathbb{R} \), the transformation \( S x S^{-1} \) of any vector \( x \) leaves the length of the transformed vector unchanged. By contrast, the operation \( S x S^\perp \) of (3.44) does not. It changes the length by a scalar factor of \( S S^\perp \). In (3.44) we are transforming a set of non-unit vectors \( \hat{t}_i \) of a common length to a set of unit vectors \( \hat{0} \hat{t}_i \). We will do so by first matching the vectors’ directions and then recovering the common scaling factor.

Recall that we assumed both the frame \( \{ \hat{t}_i \} \) and the frame \( \{ \hat{0} \hat{t}_i \} \) to be orthonormal with the same handedness (see section 3.3.2). Therefore, they are \( n \)-dimensional Euclidean vectors
with known lengths related by an orthogonal transformation.

We have therefore reduced the determination of the conformal transformation to finding a Euclidean \( n \)-dimensional rotation of \( n \) vectors. We can use any of the methods introduced in section 3.3.1 to recover this rotation. In the following we derive a recursive formula using the approach in [FD11].

We introduce the even Euclidean versor \( c_1' = c_1 (p_1 - p_2) \) which will align the two frames. Following [FD11] we find this versor as the sequence of mid-planes (see section 2.3) between corresponding vectors. Without loss of generality we take the first mid-plane to be the one between \( \vec{t}_1 \) and \( 0\vec{t}_1 \), the next one between the vector \( \vec{t}_2 \) reflected in the first mid-plane and \( 0\vec{t}_2 \) and so on. By circumstance it may occur that a single reflection aligns more than one pair of corresponding vectors. In this case the mid-plane between two equal vectors will evaluate to zero. If that happens, we assume the corresponding inversion operator to be the identity instead of the mid-plane. Recall that already aligned vectors do not change by successive reflections [FD11]. We obtain an explicit recursively defined formula for \( i = 1, \ldots, n \).

\[
\begin{align*}
c_1'^{(0)} & = 1, \\
X^{(i)} & = (-1)^{i-1} \left( \prod_{k=1}^{i} c_1'^{(i-k)} \right) \vec{t}_i \left( \prod_{k=1}^{i} c_1'^{(i-k)} \right)^{-1} - 0\vec{t}_i, \\
c_1'^{(i)} & = \begin{cases} 
X^{(i)}, & \text{if } X^{(i)} \neq 0 \\
1, & \text{if } X^{(i)} = 0
\end{cases}
\end{align*}
\]

Then we find

\[
c_1' = \prod_{i=0}^{n} c_1'^{(n-i)},
\]

where the product means successive multiplication from the right.

As a simple example consider the three-dimensional Euclidean case, i.e. \( n = 3 \). We get

\[
\begin{align*}
X^{(1)} & = \vec{t}_1 - 0\vec{t}_1, \\
X^{(2)} & = - (\vec{t}_1 - 0\vec{t}_1) \vec{t}_2 (\vec{t}_1 - 0\vec{t}_1)^{-1} - 0\vec{t}_2, \\
X^{(3)} & = 0, \\
c_1' & = 1 X^{(2)} X^{(1)} \\
& = \left( - (\vec{t}_1 - 0\vec{t}_1) \vec{t}_2 (\vec{t}_1 - 0\vec{t}_1)^{-1} - 0\vec{t}_2 \right) (\vec{t}_1 - 0\vec{t}_1) \\
& = - (\vec{t}_1 - 0\vec{t}_1) \vec{t}_2 - 0\vec{t}_2 (\vec{t}_1 - 0\vec{t}_1),
\end{align*}
\]

where we assumed that there is an orthogonal transformation relating \( \{\vec{t}_1\} \) and \( \{0\vec{t}_1\} \) (i.e. \( \vec{t}_3 \) and \( 0\vec{t}_3 \) are automatically aligned), and that \( (\vec{t}_1 - 0\vec{t}_1) \) is invertible (i.e. non-zero in this Euclidean case).

The versor obtained in this way needs to be rescaled in order to change the scale of the frame \( \{\vec{t}_i\} \) in the desired way when substituted into the non-versor transformation (3.44). We
calculate
\[
c_1 = \sqrt{\frac{1}{\sigma c'_1 c'_1}} c'_1 (p_1 - p_2)^{-1},
\]
where \(\sigma \in \mathbb{R}\) denotes the common scale of the frame \(\{t_i\}\).

Now, substituting (3.53) and (3.38) back into (3.43) gives us a full closed form solution for \(S_1\). A solution for \(S_2\) can be found in a perfectly analogous way substituting \(p'_1, p'_2, \{t'_i\}\) and \(\sigma'\) for \(p_1, p_2, \{t_i\}\) and \(\sigma\), respectively. We find the Vahlen matrix \(S = S_2^{-1} S_1\) by straightforward application of (3.15).

### 3.4. Generalization and Limitations

In this section we generalize our method to odd versors in arbitrary dimensions. We analyze the important special case of rigid body motions and present a way of extracting minimal correspondence data from Euclidean point correspondences using the conformal model of geometric algebra. Finally, we will consider the case of non-exact correspondence data.

#### 3.4.1. Even vs. Odd Versors

Throughout section 3.3 we have assumed that the correspondence data is related by a conformal transformation, which can be represented by an even versor. If the two configurations are related by a locally angle reversing anti-conformal transformation, the method has to be modified slightly.

In this case the handedness of only one of the localized vector frames will match the handedness of our transfer frame \(\{\tilde{t}_i\}\). We can compute the handedness, for example by the sign of the expression \((t_1 \wedge \cdots \wedge t_n) I_n^{-1}\), where \(I_n = e_1 \wedge \cdots \wedge e_n\). On the other frame (without loss of generality let this be \(\{t'_i\}\)) we introduce an inversion in an arbitrary but fixed hypersphere represented by an arbitrarily chosen conformal vector \(r\) and obtain \(\{p''_1, p''_2, t''_1, \ldots, t''_n\} = -r \{p'_1, p'_2, t'_1, \ldots, t'_n\} r^{-1}\). Now we compute the even conformal versor \(S'\) relating \(\{p_1, p_2, t_1, \ldots, t_n\}\) and \(\{p''_1, p''_2, t''_1, \ldots, t''_n\}\) using the method introduced in section 3.3.3, and find the desired odd conformal versor relating \(\{p_1, p_2, t_1, \ldots, t_n\}\) and \(\{p'_1, p'_2, t'_1, \ldots, t'_n\}\) by undoing the arbitrary reflection introduced before:

\[
S = -r^{-1} S' r.
\]

#### 3.4.2. Minimal Data from Exact Point Correspondences

In section 3.3.2 we argued that a localized vector frame and an additional point are minimal data determining a conformal transformation. However, sometimes it may be inconvenient to give the correspondence data in this format, e.g. because of the way such data is acquired. In this section we will introduce a way to acquire a minimal set of correspondence data from point correspondences.

Recall that our method works with purely Euclidean geometric information, i.e. point locations in Euclidean space. The weights of the conformal points do not need to be known.
A crucial prerequisite is that corresponding points are indeed related by a conformal transformation. For example, in three dimensions the counting argument shows that a conformal transformation has 10 degrees of freedom. Three point correspondences would provide only 9 DOF, leaving us one degree of freedom short, whereas four point correspondences provide 12 DOF, thus overdetermining the problem. As we will show now, if we assume the existence of a common conformal transformation, we can use the properties of conformal transformations and the representational power of CGA to find that transformation from point correspondences.

We have noted that conformal transformations preserve hyperspheres; especially, circles are mapped to circles. Also, conformal transformations preserve angles locally, i.e. if two circles intersect in some point, their images under the conformal mapping will intersect in the image of that point under the same angle. Therefore, in $n$-dimensional Euclidean space, $(n-1)$ circles intersecting in a common point at Euclidean position $p$ are enough to specify a frame of tangent vectors localized at $p$ up to scale. Note that $(n-1)$ vectors are given as the tangent directions of the respective circles in $p$ and the $n$-th vector can be determined as being orthogonal to all of those. For a reminder of how to represent circles and tangents in CGA we refer the reader to section 2.4.

In order to specify $(n-1)$ independent circles, we need at least $n+1$ conformal points $\{p_1, \ldots, p_{n+1}\}$. Without loss of generality we pick the circles

$$C_i = p_1 \wedge p_2 \wedge p_{i+2}, \quad i = 1, \ldots, n-1$$

(intersecting in $p_1$ and $p_2$). The conformal tangents to these circles in $p_1$ are given by

$$u_i = p_1 \cdot C_i, \quad i = 1, \ldots, n-1.$$  

(3.56)

Their Euclidean directions can be recovered by

$$u_i = -\infty \cdot u_i, \quad i = 1, \ldots, n-1,$$

(3.57)

$$u_n = (u_1 \wedge \cdots \wedge u_{n-1}) \cdot I_n,$$

(3.58)

where $I_n$ denotes the Euclidean $n$-dimensional pseudoscalar $I_n = e_1 \wedge \cdots \wedge e_n$. See Figure 3.3 for a visualization of the obtained setup in three dimensions.

**Figure 3.3.:** With four points, two independent circles, $C_1$ and $C_2$, can be determined. They intersect in two common points, $p_1$ and $p_2$. Without loss of generality, in $p_1$ we obtain a non-orthonormal frame of tangent vectors $\{u_i\}$, while $p_2$ will serve in the additional point correspondence needed for our method.
Due to the preservation properties of conformal transformations the corresponding vector frame \( \{ u'_i \} \) can be determined perfectly analogously by replacing the conformal points by their images \( \{ p'_1, \ldots, p'_{n+1} \} \).

Note that the localized frames, \( \{ u_i \} \) and \( \{ u'_i \} \), obtained in this way have the same mutual angles between frame vectors, if the corresponding points are related by a conformal transformation. But the frames are not orthonormal, their mutual angles and individual vector lengths depending on the chosen point correspondences. At this point we have two options to ensure the matching of the angles in the localized frames and in the transfer frame required by our method. The inner product provides us with a notion of intersection angles, and we could use it to measure the mutual angles between the \( \{ u_i \} \) and create a transfer frame with matching angles. However, we prefer to employ an orthonormalization procedure (see [HS84], section 1-3. Frames and Matrices) very similar to Gram-Schmidt orthonormalization, yielding \( \{ t_i \} \) and \( \{ t'_i \} \), respectively. This enables us to pick a Cartesian frame \( \{ e_1, \ldots, e_n \} \) as the transfer frame. With this and an additional point correspondence — without loss of generality we can pick any point other than \( p_1 \), say, \( p_2 \) — we have all the information to compute the Vahlen matrix \( S \) up to the scale parameters \( \sigma \) and \( \sigma' \) which get lost in the orthonormalization process.

![Diagram](image_url)

**Figure 3.4.:** Using a Cartesian transfer frame \( \{ e_i \} \), we assume that the original frame \( \{ t_i \} \) is related to it by scale \( \sigma \), i.e. \( \| t_i \| = \sigma \| e_i \| \), \( i = 1, \ldots, n \), while the image frame \( \{ t'_i \} \) is related to it by scale \( \sigma' \). Therefore, \( \{ t_i \} \) and \( \{ t'_i \} \) are related by scale factor \( \mu = \sigma' / \sigma \). Since it will not change the composite transformation \( S \), we can perform this scaling in part or in total in either one of the component transformations \( S_1 \) and \( S_2 \).

If the original frame and its image are related to the transfer frame by scaling factors \( \sigma \) and \( \sigma' \), respectively, they are related to each other by a scaling factor \( \mu = \sigma' / \sigma \) (see Figure 3.4). Without loss of generality we let \( \sigma' = \mu \) and \( \sigma = 1 \). Recalling the form (3.43) of the partial solution to the Vahlen matrix \( S_1 \), as well as the fact that scaling of the corresponding frame vectors is achieved by the scale of the versor \( c_1 \) (see (3.53)) we can make the required scaling factor explicit and express \( S_1 \) in terms of it as a 1-parameter family of Vahlen matrices

\[
S_1(\mu) = \begin{pmatrix}
a_1 & b_1 \\
\mu c_1 & \mu d_1
\end{pmatrix}.
\]  

(3.59)

Tracking the scaling factor through the whole procedure it turns out that the final overall
Vahlen matrix takes the form

\[
S(\mu) = S_2^{-1}S_1(\mu) = \left(\begin{array}{cc}
\hat{d}_2a_1 - \mu\hat{b}_2c_1 & \hat{d}_2b_1 - \mu\hat{b}_2d_1 \\
-\hat{c}_2a_1 + \mu\hat{a}_2c_1 & -\hat{c}_2b_1 + \mu\hat{a}_2d_1
\end{array}\right).
\]  

(3.60)

Its effect on the Euclidean position \(p\) of a point is that it moves to

\[
p' = \left(\left(\hat{d}_2a_1 - \mu\hat{b}_2c_1\right)p + \hat{d}_2b_1 - \mu\hat{b}_2d_1\right)\left(\left(\mu\hat{a}_2c_1 - \hat{c}_2a_1\right)p + \mu\hat{a}_2d_1 - \hat{c}_2b_1\right)^{-1}. 
\]  

(3.61)

Knowing a correspondence between \(p\) and \(p'\) we can solve for \(\mu\), yielding

\[
\mu = \left(p'\hat{c}_2a_1p + p'\hat{c}_2b_1 + \hat{d}_2a_1p + \hat{d}_2b_1\right)\left(p'\hat{a}_2c_1p + p'\hat{a}_2d_1 + \hat{b}_2c_1p + \hat{b}_2d_1\right)^{-1}. 
\]  

(3.62)

In order to solve (3.62) we need to know the entries of \(S_1\) and \(S_2\) up to the value of \(\mu\). We obtain them using the normalized frames \(\{t_i\}\), \(\{e_i\}\) and \(\{t'_i\}\), yielding \(S_1(1)\) and \(S_2\). Then we determine the correct scale \(\mu\) using another point correspondence in (3.62), as follows.

Note that by design of (3.55) the points \(p_1\) and \(p_2\) will be aligned with \(p'_1\) and \(p'_2\) by the 1-parameter family of Vahlen matrices (3.60). This means that (3.61) is fulfilled for these points no matter which value we choose for \(\mu\). Therefore, these correspondences do not contribute enough information to recover the scale. So, for an additional point — without loss of generality we choose \(p_3\) — we need to enforce the condition

\[
p'_3 = S(\mu)p_3S(\mu)^{-1}. 
\]  

(3.63)

Using (3.62) for \(p'_3\) and \(p_3\) with the matrix entries from the preliminary solutions \(S_1(1)\) and \(S_2\), gives us the correct \(\mu\). Substituting that back into (3.60) gives us the final, full conformal transformation.

3.4.3. Special Case: Rigid Body Motions

As mentioned before, an important special case of conformal transformations is the set of rigid body motions or Euclidean transformations, which preserve Euclidean distances. Euclidean transformations are fully characterized by the fact that they are conformal transformations that preserve the point at infinity. If it is known beforehand that the transformation we seek is a rigid body motion, the equations from section 3.3.3 simplify significantly.

A convenient way to specify the point at infinity as being preserved is to require \(p_2 = p'_2 = \eta_\infty\). But our method depends on the Euclidean position vectors \(p_1, p_2\) and \(p'_1, p'_2\) of the given points, respectively. The point at infinity cannot be specified explicitly in terms of Euclidean position vectors, which breaks down equations (3.17) and (3.19). Therefore, we must replace equations (3.35), (3.36), (3.38) and (3.53) by the following considerations.
The action of a Vahlen matrix $S$ on the point at infinity (3.34) is given by

$$S \infty P \hat{S}^{-1} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} 0 & -2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \bar{d} & \bar{b} \\ \bar{c} & \bar{a} \end{pmatrix} = \begin{pmatrix} -2a\bar{c} & -2a\bar{a} \\ -2c\bar{c} & -2c\bar{a} \end{pmatrix}. \quad (3.64)$$

Thus, if we want the point at infinity to be preserved, our conditions for $S_1$ simplify to

$$c_1 = 0, \quad (3.65)$$
$$a_1 = \Delta (S_1) \hat{d}_1^{-1}, \quad (3.66)$$
$$b_1 = -a_1 p_1, \quad (3.67)$$
$$d_1 t_1 \hat{d}_1 = \hat{t}_1, \quad (3.68)$$

and analogously for $S_2$. Note that $p_2$ has been eliminated from the equations. We therefore only require a localized vector frame and its image as correspondence data when we know that we are determining a Euclidean transformation. A scale parameter is no longer required, since it is unity for all rigid body motions.

We obtain a solution for $S = S_2^{-1} S_1$ by determining the versor $d_1$ (and analogously $d_2$) using the above methods for Euclidean frame alignment and substituting it into (3.66) and (3.67), respectively.

### 3.4.4. Non-Exact Correspondence Data

Whenever one is trying to fit a model to minimal data in general, the fit is necessarily very sensitive to noise. Assume that the correspondence data arises from some true data degraded by noise. Then the noisy data will still exactly define a set of model parameters, only they will not be the true ones. Since there is no redundancy in the data, the fitting method cannot be aware of the noise. In our specific case of fitting a conformal transformation to correspondence data, things are slightly different.

Given minimal correspondence data in the form of a localized frame and an additional point, the crucial step of our method is to determine the rotation and scaling between the corresponding Euclidean frames of vectors. This step — and only this step — introduces redundancy. Assume that the correspondence data (consisting of a localized frame of vectors and an additional point) arises from true data degraded by noise in all parameters. Our method cannot be aware of noise in the location of the frame of vectors or in the additional point. It will always find a transformation that aligns the two with their images. But our method is aware of noise in the frame vectors. If the mutual angles between vectors of the original and the image frame do not match or the original vectors and their images are not related by a common scale, the correspondence data does not define a conformal transformation.

Therefore we have to make the assumption that there is a conformal transformation taking place, i.e. that the correspondence data is exact. If that is not the case, all approaches intro-
duced in section 3.3.1 offer work-arounds\textsuperscript{4}. The recursive formula (3.48) derived from [FD11], for example, will always yield a solution, because it is the geometric product of a number of invertible Euclidean vectors. However, in case of non-exact correspondence data, only the last vector pair $t_n$ and $t'_n$ will be guaranteed to be aligned. Therefore the quality of the solution is affected by the order in which the data is processed. Moreover, if the frame vectors are not related by a common scale, some scale has to be chosen in (3.53).

By contrast, both methods introduced in [Per08] and [RFC10] give algebraic least squares solutions to the orthogonal transformation aligning two vector frames. While [Per08] enforces the orthogonality of the recovered transformation explicitly (e.g. by Lagrange multipliers), the method from [RFC10] composes the full rotation of planar rotations, fitting only the rotation planes using the SVD. Either way, once the orthogonal transformation is recovered a common scaling factor has to be chosen as well.

Note that none of these methods give a geometrically motivated measure for the error made or the uncertainty of the result, even though such a measure may exist. While both the points $p_1$ and $p_2$ will align exactly with their images $p'_1$ and $p'_2$ under the obtained conformal transformation, the effect on any other geometric object has not been quantified.

### 3.5. Conclusion

We have derived a closed form solution to a conformal versor given exact correspondences between two sets of data in $n$ dimensions. We have shown that a minimal set of data needed is a localized Euclidean vector frame and an additional point in Euclidean space. It turns out that this provides exactly the number of degrees of freedom necessary to completely specify a conformal (respectively anti-conformal), i.e. locally angle preserving (respectively angle reversing) transformation.

In order to derive this solution we have made use of the representation of the $(n+2)$-dimensional conformal model of geometric algebra by $2 \times 2$-matrices with entries from the $n$-dimensional Euclidean geometric algebra. When these matrices fulfill a number of additional constraints they are called Vahlen matrices. They correspond to the CGA versors and form a group representing conformal transformations. However, relaxing these constraints allows the matrices to represent a number of geometrical objects on which conformal transformations can act in a structure preserving manner. This enables them to represent the full conformal model of geometric algebra. Using this representation on our minimal data, we reduce the problem of calculating a general even conformal versors to that of finding an even Euclidean rotator in $n$ dimensions, which has known solutions.

The method works with purely Euclidean geometric information on point locations and direction vectors. As opposed to other methods [Per08, RFC10, FD11] which one may attempt to extend to this problem, knowledge of the weights of conformal points is not required. This is of

\textsuperscript{4}The approaches introduced in section 3.3.1 offer work-arounds only to the sub-step of finding the Euclidean orthogonal transformation between frames of Euclidean vectors that are entirely known. None of the approaches can replace the method presented here for finding a conformal transformation from geometric data.
advantage when information about the concrete homogeneous representation of the geometric objects is not available — a very common situation in practice.

If the correspondence data is not exact (for example, if the mutual angles between the vectors of the localized frames are not preserved), our method can still yield a conformal transformation. Assuming that the non-exactness is due to noise degrading the true (and exact) data, the resulting transformation will deviate from the true transformation. To date, the degree of this deviation is not quantified. In [RFC10] one may find an approach that yields a solution to the final step of our method which recovers the orthogonal transformation between two frames even for noisy correspondence data using singular value decomposition, although an analysis of the quality of the recovered rotation is missing. Geometrically motivated methods for representing or propagating noise are not available (see [Per08] for an approach to noise propagation on an algebraic level). Those would be a prerequisite for obtaining approximate solutions under non-exact correspondences or optimization of these solutions under certain aspects, such as minimization of Euclidean distances.

Regardless, for minimal data our method of reducing the problem of finding a conformal transformation in $n$ dimensions to that of finding a Euclidean orthogonal transformation in $n$ dimensions remains valid, even if the data correspondence is not exact. This means that the full problem of finding a conformal transformation becomes amenable to various methods of dealing with noise or uncertainty that are available for determining Euclidean orthogonal transformations.
Finding Articulated Structures from Motion of the End-Frame

Recovering an articulated structure from the observation of moving features is a variant of the structure-from-motion problem. It finds application, for example, in computer vision and motion capturing, where a sequence of video images is used to reconstruct a moving human body.

Often the foundation for this articulated motion recovery is formed by observations of features on all links of an articulated structure moving over time. In this chapter we show that under certain reasonable assumptions observations of features on a single link are enough to recover the articulated structure responsible for the observed motion. The assumptions we make here are constant speed of the motions of the individual links, observation intervals that are equally long and non-degeneracy of the articulated structure to be inferred.

By tracking a set of features moving through space over instants in time, it is possible to determine the instantaneous motion between them. If the features are part of an (unknown) open kinematic chain, the instantaneous motions between consecutive instants change. From these changes it is possible to reconstruct the constituent elemental transformations that each of the links of the structure undergoes, without observing (features on) all the links. Moreover, very few time instances — in the exact case, roughly $N$ instants for $N$ links — are necessary for solving for the motion parameters of the individual links.

We formulate the problem using the conformal model of geometric algebra (CGA). Then
— lacking the required tools in pure geometric algebra — we find matrix formulations for the
motion equations. We show that these equations are solvable for the motion parameters in a
least squares sense for up to three joints. The least squares solution makes the approach viable
for settings where observations are degraded by noise. The method we present generalizes to
cases with arbitrarily many joints, but finding an explicit solution for more than three joints
seems to become computationally prohibitive. In practice, additionally observing the motion
of intermediate links can often be used to subdivide an articulated structure into parts with
up to three links.

Even though we focus on the recovery of articulated structures, the decomposition method
presented in this chapter can be used on composite motions whose basis is no physical artic-
ulated structure. A simple example would be a rolling wheel, whose points perform an axis
rotation composed with a forward translation.

4.1. Introduction

Structure-from-motion has come to refer to a wide class of problems. Recovering an articu-
lated structure from feature observations is an important variant which finds applications in
calibration, motion capturing and computer vision.

4.1.1. Related Work

In this section we give a brief historical cross section through the literature related to re-
covering articulated motion from image sequences. We only regard what we may call model
free approaches, which try to recover the full articulated structure. By contrast, model based
approaches either assume a full-fledged model (e.g. of a human body) and only learn the pa-
rameters (e.g. joint angles, limb dimensions etc.) from image sequences or incorporate a strong
prior for humanoid motion (e.g. using hard coded probabilistic constraints).

Instead of giving an exhaustive survey of related literature we mention what to the best
of our knowledge we regard as seminal work. Often the approaches have been picked up and
developed further by other authors.

Several approaches to the articulated motion recovery problem exist. Historically the older
ones employ algebraic geometry. That means they use geometric intuition and various heuris-
tics to set up a number of simultaneous equations that can be solved for parameters describing
the articulated motion that generated the observations. A classical example is [WA82], in
which the authors assume that points on a rigid object perform a fixed axis rotation. The
trajectories traced out by these points are circles around the axis. Circles are projected onto
ellipses when viewed by some imaging device. After recovering the parameters of the ellipses
on which the 2D observations lie, a 3D reconstruction algorithm is used to recover the rotation
axis in three-dimensional space. For articulated structures, the authors of [WA82] assume that
a correct segmentation of the set of all observed features into several rigid links is known.

A somewhat more recent approach [KMB94] starts out with fitting the model of a de-
formable geometric object to all observed point features. The model is then split into multiple
such models under physically motivated conditions: the authors impose an energy function that determines when it is reasonable to assume that points belong to different parts of an articulated structure rather than to the same deformable object. The models and constraints are based on complicated polynomial equations and their particular approach works in 2D only.

In [HHNQ97] Holt et al. present a very extensive analysis of the problem of determining an articulated structure from perspective projections of points on the structure in motion. As an example they assume a simplified and fixed model of the human body. They enforce linear constraints for coplanarity of the moving points and rigidity of the links. From this they derive a number of polynomial equations in the positions of certain body parts which they solve using numerical solvers such as MACAULAY, MAPLE and POLSYS. They distinguish a number of special cases and determine in which of these cases unique solutions are possible and for how many instants the moving features have to be observed. The authors claim that their assumed model for the human body is just an example and their analysis extends to more general structures.

Modern development is driven by a more computational approach called factorization. In [CK98] Costeira and Kanade collect a number of 2D point observations into a matrix which they factorize into a motion matrix and a shape matrix using singular value decomposition. The shape matrix describes the constellation of the original points while the motion matrix describes the transformation of points over time. Ambiguities and constraints (such as the fact that the motion matrix is to describe a rigid body motion) are dealt with by setting up an overdetermined system of linear equations that can be solved for the motion matrix components in a least squares sense. A big obstacle in this approach is that the shape and motion matrices interact with each other (i.e. only the product of shape and motion matrix explains the transformation of observed features) due to noise and non-rigidity of the observed objects. Costeira and Kanade overcome this by introducing a shape interaction matrix which determines whether features lie on the same or on different objects. They cluster this matrix into independent blocks by employing some energy function on the shape interaction matrix. The main limitation of their approach is that the recovered motions are independent — meaning that articulated motion cannot be recovered properly — and that they assume the observed features to be the result of an orthographic projection. As a consequence their approach works only in 2D.

Others have generalized Costeira and Kanade’s approach in different directions. For example, Yan and Pollefeys [YP05] put emphasis on dealing with articulated motion based on the idea that the interdependence of joined links leads to a loss of degrees of freedom for dependent motion. By finding the subspace in which the lost degrees of freedom lie, they also recover the nature and geometric configuration of the joints. Fayad et al. [FDBA09] introduce weights on the different motions to strike a balance between deformable models and articulated rigid body motions. Paladini et al. [PDBS+09] focus on dealing with missing data. They introduce an iterative algorithm which estimates an initial motion matrix, projects it onto the motion manifold and then alternates between estimating the shape matrix from the motion matrix
Finding Articulated Structures from Motion of the End-Frame

and vice versa until convergence.

An alternative approach is presented by Ross et al. [RTZ08] who use unsupervised maximum likelihood learning to infer 2D stick figure models from 2D point observations. The explicit likelihood formula for the stick figure models involves many parameters and iteratively finding its maximum requires many iteration cycles. They report results of varying quality.

4.1.2. Our Approach

In this chapter we show that it is possible to recover articulated motion from observations of the last link of a kinematic chain alone. This presents an alternative to other model free approaches as introduced in the previous section and is complementary to those.

Instead of assuming particular features, such as points, lines etc. we assume that a set of features is observed that is sufficient to recover the motion of the last link. For different classes of motion different sets of features may be used. For example, to recover a general conformal transformation in \( n \) dimensions a localized frame of vectors and an additional point may be observed; for recovering a 2D rigid body motion observing a set of two points is enough.

Tracking a set of sufficient correspondence data moving through space over multiple instants in time, it is possible to determine the instantaneous motion between them. If the correspondence data are part of an unknown articulated structure — for example, an open kinematic chain — the changes of instantaneous motions from one instant to the next allow us to infer the constituent elemental motion that each of the links of the chain undergoes, even without observing correspondence data on all the links. We can recover kinematic information, such as joint velocities, as well as geometric information, such as translation vectors and location of rotation axes in space. Figure 4.1 shows a three-dimensional articulated structure with three links moving in space.

After an initial general formalization of the problem statement we will focus on three-dimensional rigid body motions. More specifically, in this chapter we will assume that we observe correspondence data sufficient for recovering an instantaneous 3D rigid body motion of the last link of an open kinematic chain. We track this correspondence data over multiple equally spaced instants in time in order to obtain a number of instantaneous 3D rigid body motions.

Furthermore, we assume that this correspondence data is part of an open kinematic chain. An open kinematic chain is a number of rigid bodies linked by joints (in the general case they can be rotational and prismatic) with one of the rigid bodies at rest with respect to some world coordinate system. The assumption of an open kinematic chain is usually not fulfilled when observing mobile platforms such as living creatures, especially when they are running, jumping or flying. However, an open kinematic chain is much more convenient to model and we will adhere to it for now because often the assumption is fulfilled locally — for example, when the subject is moving slowly and has at least one link on the ground or when the base of the kinematic chain is at rest relative to some known motion (e.g. the shoulder with respect to the torso).

Finally, we assume that each of the links of the kinematic chain is moving with constant
4.1 Introduction

Figure 4.1: An articulated structure with three links. Each link is connected to the next by a joint that allows rotation about the link’s axis and a translation along it. The solid grey lines depict the start and end configuration, the dotted lines show intermediate instants in time. The end frame is shown in solid red, green and blue lines.

speed over the whole sequence of observations. With our approach only a small number of observations is necessary. As a consequence the whole sequence of observations extends over a small time span. We believe that the assumption of constant speed over a very small time span is reasonable.

We formulate the geometrical problem in a coordinate free manner using CGA versors to represent the geometrical rigid body relationships. In order to obtain the equations for the versor components and solve them we revert to the linear algebra analogue of CGA briefly discussed in chapter 2. We use it to set up a number of linear equations describing the instantaneous transformations between correspondence data as well as the transformations between different instantaneous transformations, which eventually can be solved for the components of the versors describing the elemental motions of the links of the assumed kinematic chain.

The rest of this chapter is structured as follows. In section 4.2 we give a formal problem statement and introduce the notation used in the rest of this chapter. Then we describe the matrix representation for geometric algebra versors representing 3D rigid body motions using a subspace of $\mathcal{G}(\mathbb{R}^{4,1})$. In section 4.3 we present a method to decompose a nested rigid body motion into its elemental components given some observations of instantaneous motions. We reflect on the assumptions we have made to derive this solution in section 4.4. Here, we also remark on geometric degeneracies which may make the problem ill posed and may preclude a unique solution. Conclusions are drawn in section 4.5.
4.2. Preliminaries

In this section we first formalize our assumptions and the problem statement in the language of geometric algebra versors. The considerations in section 4.2.1 are so general that they do not only pertain to composite rigid body motions. Any transformation that can be represented by even versors yields to the formalism presented there. However, in the context of this chapter we are only interested in rigid body motions and we will focus on these in section 4.2.2 and later. Essentially, we are trying to fit the parameters of transformations to observed data. These transformations can be represented by matrices as well and we explain that representation in detail. Later, in section 4.3, we will use it to set up a system of equations linear in the transformation parameters for which solutions can be found employing well-known linear algebra techniques.

4.2.1. Notation and Glossary

We assume sets of correspondence data sufficient to recover the transformation of a certain class between two data sets. For the class of conformal transformations such a set may consist of a localized frame and an additional point, for rigid body motions it might be a set of points or a localized frame etc. In general, we will call such a set a constellation. Observing a constellation at two different instants in time allows us to determine the instantaneous motion between those instantiations. We denote a constellation observed at instant $i$ by $X(t_i)$, abbreviated as $X_i$.

![Image of a kinematic chain](image)

Figure 4.2.: In 2D, the last link of a kinematic chain consisting of one prismatic and one rotational joint may trace out the trajectory described by the true motion $^0R(t)$ over continuous time $t$. By contrast, the instantaneous motions between two discrete time instants can be described by different rotations in the plane. Here the instantaneous motion $^1R_1$ between instants 0 and 1 (respectively, $^1R_2$ between instants 1 and 2) is a rotation about the point $c_1$ (respectively, $c_2$).

The instantaneous motion is the apparent motion that the constellation performs from one
4.2 Preliminaries

instant to the next. It does not necessarily describe the motion that the constellation undergoes in the continuous time between two instants. See Figure 4.2 for the example of a rigid body motion in 2D. In the following we focus only on motions that can be represented by (even) versors. Let \( R(t) \) be an even versor parametrized by time \( t \). Then we can write it as the exponential of a bivector \( B(t) \) parametrized by time (see (2.59)).

\[
R(t) = \exp(B(t)) \quad (4.1)
\]

If the parametrization of \( B \) by \( t \) is linear we call \( R \) an \textit{elemental versor} and say that it is generated by a \textit{constant speed bivector} \( B \). In that case we have

\[
R(t) = \exp(B(t)) = \exp(tB) = (E)^t, \quad (4.2)
\]

where \( E = \exp(B) \) is a constant versor and \( R \) can be written as a power of \( E \).

The motion of the last link of a kinematic chain can be written as a composite motion consisting of all the elemental motions performed by the links between the base and the last link. Let the kinematic chain consist of \( N \) links, each performing an elemental motion represented by an elemental versor \( (E_j)^t \), \( j = 1, \ldots, N \). We number the elemental motions from the base of the kinematic chain outward, i.e. the first link at the base performs an elemental motion described by \( E_1 \). The composite motion of the last link is then represented by the versor

\[
0 R(t) = (E_N)^t (E_{N-1})^t \ldots (E_1)^t. \quad (4.3)
\]

We will call this motion the \textit{true motion}, as it represents the continuous motion that the last link actually performs with time \( t \). The true motion cannot be observed directly. All that can be observed are snapshots of the constellation at discrete time instants. The apparent motions between those instants are described by the instantaneous motions.

We assume that the intervals between the instants at which we observe our constellation are \textit{equally long} — without loss of generality we take them to be of \textit{unit length}. Furthermore, we assume that the true motion from instant 0 to any instant \( i \) is \textit{structurally the same}. By this we mean that the constellation at time \( i \) is obtained from the constellation at time 0 by the composite motion (4.3), i.e. for all \( i \in \mathbb{N} \)

\[
X_i = 0 R(i) X_0 \left(0 R(i)^{-1}\right)^{-1}, \quad (4.4)
\]

with

\[
0 R(i) = \exp(iB_N) \exp(iB_{N-1}) \ldots \exp(iB_1) = (E_N)^i (E_{N-1})^i \ldots (E_1)^i, \quad (4.5)
\]
that is, the true motion $0R(i)$ is a composite (or nested) transformation relative to some starting constellation $X_0$ where each of the $N$ component transformations $(E_j)^i$ is an elemental versor generated by a constant speed bivector $B_j$.

Note that the true motion cannot be observed directly, but it can be inferred from the instantaneous motions between consecutive instants. The instantaneous motion of the constellation between instants $i - 1$ and $i$ can be written as the difference between the true motion from instant 0 to instant $i - 1$ and the true motion from instant 0 to instant $i$ as follows. We will use the superscript index of a versor to indicate the difference between transformations (and refer to them as meta-motions below). Since $0R$ denotes the true motion we write $1R$ for the instantaneous motion. Later we will use $2R$, $3R$ etc. to denote the differences between the instantaneous motions, the differences between those differences etc. (see Figure 4.3). Furthermore we will use subscripts to indicate the time instants between which the differences act (see Figure 4.3). For the instantaneous motion we have

$$1R_i = 0R(i) (0R(i - 1))^{-1}, \quad (4.6)$$

and we find that the constellation at time $i$ can be obtained from that at time $i - 1$ by

$$X_i = 1R_i X_{i-1} (1R_i)^{-1}. \quad (4.7)$$

If the true motion $0R(i)$ consists of only one elemental component motion $E_1$, the instantaneous motion $1R_i$ remains the same at any instant $i$, as we can see by

$$1R_i = 0R(i) (0R(i - 1))^{-1} = (E_1)(E_1)^{-1(i-1)} = E_1. \quad (4.8)$$

If the true motion $0R(i)$ consists of $N > 1$ component motions, the observed instantaneous motion will change with time (Figure 4.2 shows an example in the 2D plane). Such a change of the instantaneous motion can be modeled by a linear transformation of the versor describing it. We will call this linear transformation a \textit{meta-motion} and denote the level at which a meta-motion acts upon other meta-motions by its superscript.

For a composite motion consisting of a general number $N$ of elemental component motions the progression of meta-motions can be described by a recursive formula. We let $kR_i$ denote the meta-motion of meta-level $k$. The subscript loosely indicates the time instants between which motions are observed, but with increasing meta-level $i$ becomes more and more disassociated with the actual time instant $i$, see Figure 4.3.
Figure 4.3: As time $t$ progresses, the last link undergoes true motion $^0R(t)$. Directly observable is only the instantaneous motion at instant $i$, $^1R_i$. The change of the instantaneous motions over time is described by meta-motions $^2R_i$, their change by meta-motions $^3R_i$ etc.

More specifically, we let

$$^kR_1 = E_NE_{N-1} \cdots E_k,$$  \hspace{1cm} (4.9)

$$^kR_{i+1} = ^{k+1}R_{i+1}^kR_i\left(^{k+1}R_i\right)^{-1}, \quad \text{for } k = 1, \ldots, N-1,$$  \hspace{1cm} (4.10)

$$^N R_i = E_N, \quad \text{for all } i \in \mathbb{N}.$$  \hspace{1cm} (4.11)

For example, for a composite motion consisting of only $N = 2$ components we have

$$^1R_{i+1} = ^2R_{i+1}^1R_i\left(^2R_i\right)^{-1} = E_2^1R_i(E_2)^{-1},$$  \hspace{1cm} (4.12)

that is

$$^2R_i = E_2, \quad \text{for all } i \in \mathbb{N}.$$  \hspace{1cm} (4.13)

An obvious consequence of (4.10) and (4.11) is that

$$^N^{-1}R_{i+1} = (E_N)^iN^{-1}R_i(E_N)^{-i}.$$  \hspace{1cm} (4.14)

The recursive definitions (4.9), (4.10) and (4.11) are crucial for solving for the component motion parameters in section 4.3, but first we need to prove that they accurately describe the change of observed instantaneous motions between time instants. The idea is to derive (4.4) using (4.9) – (4.11). We begin by writing out the instantaneous motion between instants $i$ and $i+1$ in terms of the elemental motions $(E_j)^i$ comprising the true motion $^0R(i)$. We start out
with (4.3) and factor out the deepest non-elemental meta-motion $N^{-1}R$ at different instants.

\begin{align*}
0R(i + 1) &= (E_N)^{i+1} \ldots (E_i)^{i+1} \\
&= (E_N)^{i} \underbrace{E_N E_{N-1} (E_{N-1})^i (E_{N-2})_{i+1} \ldots (E_1)^{i+1}}_{N^{-1}R_1} \\
&= (E_N)^{i} \underbrace{N^{-1}R_1 (E_N)^{-i} (E_N)^i (E_{N-1})^i (E_{N-2})_{i+1} \ldots (E_1)^{i+1}}_{N^{-1}R_{i+1}} \\
&= N^{-1}R_{i+1} (E_N)^{i-1} \underbrace{N^{-1}R_1 (E_N)^{-i} (E_N)^i (E_{N-1})^{i-1} (E_{N-2})_{i+1} \ldots (E_1)^{i+1}}_{N^{-1}R_{i}} \\
&\vdots \\
&= N^{-1}R_{i+1} \ldots N^{-1}R_1 (E_{N-2})^{i+1} \ldots (E_1)^{i+1}, \tag{4.15}
\end{align*}

where we repeatedly applied the trick of inserting ever decreasing powers of $(E_N)^{-1}$ and $E_N$ and using (4.9) as well as (4.14). Effectively, we have split off the elemental versors $E_N$ and $E_{N-1}$ and incorporated them in meta-motion $N^{-1}R$. From here we split off the next elemental versor $E_{N-2}$ and get

\begin{align*}
0R(i + 1) &= \\
&= N^{-1}R_{i+1} \ldots N^{-1}R_1 E_{N-2} (E_{N-2})^i (E_{N-3})_{i+1} \ldots (E_1)^{i+1} \\
&= N^{-1}R_{i+1} \ldots N^{-1}R_1 \left(N^{-1}R_1\right)^{-1} \left(N^{-1}R_2\right)^{-1} \underbrace{(E_{N-1})_{i+1}}_{N^{-2}R_1} \\
&= N^{-1}R_{i+1} \ldots N^{-1}R_2 N^{-2}R_3 (E_{N-2})^i (E_{N-3})_{i+1} \ldots (E_1)^{i+1} \\
&= N^{-1}R_{i+1} \ldots N^{-1}R_3 N^{-2}R_2 N^{-3}R_1 \left(N^{-1}R_1\right)^{-1} \left(N^{-1}R_2\right)^{-1} \left(N^{-1}R_3\right)^{-1} N^{-1}R_1 (E_{N-2})^i (E_{N-3})_{i+1} \ldots (E_1)^{i+1} \\
&\vdots \\
&= N^{-2}R_{i+1} \ldots N^{-1}R_1 \left(N^{-1}R_3\right)^{-1} \left(N^{-2}R_2\right)^{-1} \left(N^{-3}R_1\right)^{-1} N^{-1}R_1 (E_{N-2})^i (E_{N-3})_{i+1} \ldots (E_1)^{i+1}. \tag{4.16}
\end{align*}

Here we have applied (4.10) in the form

\begin{align*}
kR_{i+1} R_{i+1} = k+1 R_{i+1} k R_{i}, \tag{4.17}
\end{align*}

which permits “swapping” consecutive superscripts between versors with consecutive subscripts. Effectively we have “shifted” $N^{-2}R$ to the left while increasing its subscript by one with every shift. We can repeat that trick for all factors $E_{N-2}$ until all the $N^{-1}R$ have been
replaced by $N^{-2}R$. We obtain

$$0R(i + 1) = N^{-2}R_{i+1}N^{-2}R_{i} \ldots N^{-2}R_{1}(E_{N-3})^{i+1} \ldots (E_{1})^{i+1}, \quad (4.18)$$

and repeat the procedure in (4.16) for the next elemental versor $E_{N-3}$ and so on, decreasing the meta-level superscript of the meta-motions by one each time. Eventually, we arrive at

$$0R(i + 1) = 1R_{i+1}1R_{i} \ldots 1R_{1}, \quad (4.19)$$

If we write out $(0R(i))^{-1}$ using the structure of (4.3) the order of (inverse) elemental versors is reversed. We can apply a procedure analogous to the one above from the right. Since the (inverse) elemental versor appear with one power less we can only convert them to (meta)versors with a time subscript that is smaller by one. We have

$$0R(i + 1)(0R(i))^{-1} = 1R_{i+1}1R_{i} \ldots 1R_{1}(1R_{1})^{-1} \ldots (1R_{i})^{-1} = 1R_{i+1}, \quad (4.20)$$

which is in agreement with (4.6). Results (4.19) and (4.20) may seem intuitively obvious, but we reached them using our recursive definitions (4.9), (4.10) and (4.11). This shows that those accurately describe the change of instantaneous motion of an observed constellation between instants, and that we can use them as a full description of the structure in what follows.

### 4.2.2. Transformations of Rigid Body Motion Versors

The true motion and the instantaneous motion transform geometric objects by some group action — e.g. rotations, translations, screw motions or conformal transformations. Essentially, these transformations are coordinate transformations acting on objects in a structure preserving way by the sandwiching versor product (see (4.4) and (4.7), respectively). With the description of meta-motions as in (4.10), however, it has become clear that the transformations within the group need to be worked out in more detail.

For a number of reasons, within the scope of this chapter we will consider the group of special Euclidean transformations in three dimensions $SE(3)$. Firstly, since our focus is on inferring an open kinematic chain consisting of rigid bodies, we only need to consider rigid body motions for the component transformations. Secondly, due to group closure any nested rigid body motion is a rigid body motion and — according to Chasles’ theorem — can be described by a screw motion. Therefore all instantaneous motions between observed constellations can be written as screw motions. Thirdly, the group $SE(3)$ is a Lie group and affords an exponential map from the tangent space at the identity to group elements. Any screw motion is continuously connected to the identity and yields to our definitions in the previous section, e.g. that of elemental versors (4.2). And finally, screw motions can be parametrized by geometrically meaningful, intuitive quantities such as translation vectors and rotation axes and angles.
Finding Articulated Structures from Motion of the End-Frame

More specifically, every 3D screw motion is characterized by three geometric quantities: firstly, a screw axis/angle $\omega$ whose direction is the direction of the rotation axis and whose magnitude is the rotation angle; secondly, a translation vector $t = \frac{p}{2\pi} \omega$ along that axis, where $p$ is the screw motion’s pitch, i.e. the amount of translation per revolution; thirdly, an offset of the screw axis from the origin encoded as follows. If $u$ is any point on the screw axis, then we define $c = u \times \omega$, which is also called the moment of the line which coincides with the screw axis. The moment is independent of the particular choice of $u$. Figure 4.4 visualizes these relationships. To sum up, a 3D screw motion can be characterized by

$$\text{rotation axis/angle: } \omega \quad (4.21)$$
$$\text{pitch: } p \quad (4.22)$$
$$\text{translation vector: } t = \frac{p}{2\pi} \omega \quad (4.23)$$
$$\text{point on offset screw axis: } u \quad (4.24)$$
$$\text{moment vector: } c = u \times \omega \quad (4.25)$$

We note that these quantities are not independent of each other. A 3D screw motion has only 6 degrees of freedom. To support geometric intuition, however, we find it convenient to maintain the quantities (4.21) – (4.25) separately.

In the conformal model of geometric algebra a screw motion is generated by a bivector $B$ which can be written in terms of the screw parameters as follows.

$$B = -\frac{1}{2} \omega \cdot I_3 - \frac{1}{2} (t - c) \wedge n_\infty, \quad (4.26)$$

where $I_3$ denotes the Euclidean 3D pseudoscalar. The generated screw motion versor may be

---

**Figure 4.4:** A screw motion in 3D is described by its screw axis, determined by the rotation axis/angle $\omega$, the pitch $p$ which defines a translation vector $t$ along the screw axis, and a moment vector $c$. We show the coordinate origin $n_0$ for reference. Note that not all parameters are independent. A 3D screw motion has 6 degrees of freedom.
found by writing out the power series of the exponential map (2.59) from chapter 2. The result is

\[
\exp(B) = \cos \left( -\frac{1}{2} \|\omega\| \right) + \frac{1}{\|\omega\|} \sin \left( -\frac{1}{2} \|\omega\| \right) \omega \cdot I_3 \\
+ \left( -\frac{1}{2} \cos \left( -\frac{1}{2} \|\omega\| \right) t + \frac{1}{\|\omega\|} \sin \left( -\frac{1}{2} \|\omega\| \right) c \right) \wedge n_\infty \\
- \frac{1}{2} \|t\| \sin \left( -\frac{1}{2} \|\omega\| \right) I_4,
\]

(4.27)

where \( I_4 \) denotes the grade-4 part of a screw motion versor, \( I_4 = e_1 \wedge e_2 \wedge e_3 \wedge n_\infty \). Both (4.26) and (4.27) are geometric algebra representations of concepts introduced in, for example, [SB01] and [Sel04]. With this we can write any 3D screw motion versor as a vector on the 8-dimensional blade basis

\[
\{1, e_2 \wedge e_3, e_3 \wedge e_1, e_1 \wedge e_2, e_1 \wedge n_\infty, e_2 \wedge n_\infty, e_3 \wedge n_\infty, I_4\},
\]

(4.28)

obtaining

\[
\exp(B) \equiv \begin{pmatrix}
\cos \left( -\frac{1}{2} \|\omega\| \right) \\
\frac{1}{\|\omega\|} \sin \left( -\frac{1}{2} \|\omega\| \right) \omega \\
- \frac{1}{2} \cos \left( -\frac{1}{2} \|\omega\| \right) t + \frac{1}{\|\omega\|} \sin \left( -\frac{1}{2} \|\omega\| \right) c \\
- \frac{1}{2} \|t\| \sin \left( -\frac{1}{2} \|\omega\| \right)
\end{pmatrix}.
\]

(4.29)

This is a vector representation consisting of 8 scalar components. In (4.29) we shortened the notation by writing one scalar two three-dimensional vectors and another scalar. When we talk about the components of the vector representation of a versor we mean 8 scalar components.

The translation vector \( t \) and the vector \( \omega \) characterizing the screw axis are parallel. By convention they are also taken to have the same orientation. This is not a hard requirement, though, since algebraically any rotation about a vector \( \omega \) through an angle \( \|\omega\| \) can also be described by a rotation about a vector \(-\omega\) through an angle of \(2\pi - \|\omega\| \). This ambiguity in the rotational part manifests in a sign change of the screw motion’s pitch \( p \) and therefore of the translation vector \( t \) which is defined in terms of \( p \) and \( \omega \) by (4.23). In the most general case we would need to write the last (i.e. grade-4) component of (4.27), respectively (4.29), as follows.

\[
\langle \exp(B) \rangle_4 = -\frac{1}{2} \|\omega\| t \cdot \omega \sin \left( -\frac{1}{2} \|\omega\| \right).
\]

(4.30)

But the formulation used in (4.27) and (4.29) will be more convenient in later considerations.

The form (4.29) shows that the first four scalar components (i.e. the top two lines) are responsible for the purely rotational part of the transformation because they only depend on the rotation axis/angle \( \omega \). The last four components (i.e. the bottom two lines in (4.29)) depend on the translation vector \( t \) as well as on the moment \( c \). For pure rotations about lines through the origin they are zero.

The versor product of a screw motion versor \( \exp(B) \) by another screw motion \( G \in SE(3) \) is
of the same form (4.27)

\[ G \exp(B)G^{-1} = \exp(GBG^{-1}) = \exp(B'), \quad (4.31) \]

with

\[ \omega' = [R_G] \omega, \quad (4.32) \]

\[ t' = [R_G] t, \quad (4.33) \]

\[ c' = [R_G] c + ([I] - [R_G]) u_G + t_G \times [R_G] \omega \]
\[ = [R_G] c + k_G \times [R_G] \omega, \quad (4.34) \]

where \([R_G]\) denotes the matrix of purely Euclidean rotation characterizing \(G\), \([I]\) denotes the identity matrix, \(t_G\) denotes the translation vector characterizing \(G\) and \(u_G\) denotes some point on the screw axis of \(G\). Therefore

\[ k_G = ([I] - [R_G]) u_G + t_G \quad (4.35) \]

is a fixed vector with respect to \(\exp(B)\), depending only on \(G\). Moreover, \(k_G\) is independent of the particular choice of \(u_G\) (see Figure 4.5).

**Figure 4.5:** Since \([R_G]\) is a rotation about the vector \(\omega_G\) and \(u_G\) denotes a point on the screw axis \(L\), the chord between \(u_G\) and its rotated version \([R_G] u_G\) is independent of the choice of \(u_G\). In this figure we see that \(([I] - [R_G]) u_G = ([I] - [R_G]) u_G'\).

Using (4.32), (4.33) and (4.34) the versor product of a screw motion versor \(\exp(B)\) by some screw motion versor \(G\) can be written as a linear transformation of the components of \(\exp(B)\) (note that because of (4.31) the same linear transformation acts on \(B\) and \(\exp(B)\)). The matrix that represents this transformation is called the adjoint representation of \(G\). An introduction to this representation can be found in [Sel04] and here we write it out in more detail. In matrix representation we have

\[
G \exp(B)G^{-1} = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & [R_G] & 0 & 0 \\
0 & [k_G] \times [R_G] & [R_G] & 0 \\
0 & 0 & 0 & 1
\end{pmatrix} \exp(B), \quad (4.36)
\]
where it is understood that \( \exp(B) \) is represented as in (4.29) and \( [k_G]_x \) denotes the skew symmetric matrix generated by the components of the vector \( k_G \), that is if \( k_G = (k^1, k^2, k^3)^T \) then

\[
[k_G]_x = \begin{pmatrix}
0 & -k^3 & k^2 \\
k^3 & 0 & -k^1 \\
-k^2 & k^1 & 0
\end{pmatrix}.
\] (4.37)

Note that the scalar part as well as the grade-4 part of \( \exp(B) \) are preserved by such a transformation (4.36), which is obvious considering that (4.32) and (4.33) imply that \( \|\omega'\| = \|\omega\| \) and \( \|t'\| = \|t\| \).

The geometric product of a versor \( G \) with an element \( F \) of the geometric algebra is a linear transformation of the components of \( F \) and can be written as a \( G \)-matrix acting on an \( F \)-vector. As mentioned briefly in section 2.5 the exact nature of the linear transformation depends on whether we consider multiplication from the left or from the right. Let \( F, G \in SE(3) \) be screw motion versors. Then the composite versor can be expressed as a linear transformation of \( F \) by

\[
GF = \begin{pmatrix}
\|R_G\|_l & 0 \\
\|M_G\|_l & \|R_G\|_l
\end{pmatrix} F.
\] (4.38)

Note that the matrix in (4.38) is not identical to the adjoint representation of \( G \) mentioned in [Sel04] because it does not act on the Lie algebra of \( SE(3) \). Instead of representing the double-sided adjoint action as in (4.31) it represents the one-sided multiplicative group action. We need this in order to formulate our recursive definition (4.10) in matrix representation.

In (4.38), \( \|R_G\|_l \) is an orthogonal \( 4 \times 4 \) matrix with additional symmetries, \( \widehat{[R_G]}_l \) is a version of \( \|R_G\|_l \) with some sign changes and switched rows and columns and \( \|M_G\|_l \) is a multiple of \( \|R_G\|_l \) by a skew symmetric matrix with some sign changes and switched rows and columns. The details of the switches and sign changes depend on the ordering of the blade basis chosen to represent the screw motions versors in (4.29). If the screw motion versor represented as a vector on the above blade basis (4.28) is \( G = (g^1, \ldots, g^8)^T \), then we have

\[
\|R_G\|_l = \begin{pmatrix}
g^1 & -g^2 & -g^3 & -g^4 \\
g^2 & g^1 & g^4 & -g^3 \\
g^3 & -g^4 & g^1 & g^2 \\
g^4 & g^3 & -g^2 & g^1
\end{pmatrix},
\] (4.39)

\[
\widehat{[R_G]}_l = \begin{pmatrix}
g^1 & g^4 & -g^3 & -g^2 \\
-g^4 & g^1 & g^2 & -g^3 \\
g^3 & -g^2 & g^1 & -g^4 \\
g^2 & g^3 & g^4 & g^1
\end{pmatrix},
\] (4.40)

\[
\|M_G\|_l = \begin{pmatrix}
g^5 & -g^8 & g^7 & -g^6 \\
g^6 & -g^7 & -g^8 & g^5 \\
g^7 & g^6 & g^5 & -g^8 \\
g^8 & g^5 & g^6 & g^7
\end{pmatrix}.
\] (4.41)
For multiplication from the right the equations look similar with subtle differences in the sign changes and in the rows and columns switched. We have

\[
FG = \begin{pmatrix}
[R_G]_r & 0 \\
[M_G]_r & [R_G]_r
\end{pmatrix} \begin{pmatrix}
F
\end{pmatrix}, \quad (4.42)
\]

with

\[
[R_G]_r = \begin{pmatrix}
g^1 & -g^2 & -g^3 & -g^4 \\
g^2 & g^1 & -g^4 & g^3 \\
g^3 & g^4 & g^1 & -g^2 \\
g^4 & -g^3 & g^2 & g^1
\end{pmatrix}, \quad (4.43)
\]

\[
[R_G]_r = \begin{pmatrix}
g^1 & -g^4 & g^3 & -g^2 \\
g^4 & g^1 & -g^2 & -g^3 \\
-g^3 & g^2 & g^1 & -g^4 \\
g^2 & g^3 & g^4 & g^1
\end{pmatrix}, \quad (4.44)
\]

\[
[M_G]_r = \begin{pmatrix}
g^5 & -g^8 & -g^7 & g^6 \\
g^6 & g^7 & -g^8 & -g^5 \\
g^7 & -g^6 & g^5 & -g^8 \\
g^8 & g^6 & g^5 & g^7
\end{pmatrix}. \quad (4.45)
\]

The structure of equations (4.38) and (4.42) implies that we need to consider only the purely Euclidean, rotational part of the transformations represented by \( F \) and \( G \) in order to describe the rotational part of their product. The zero in the upper right corner of the matrices means that the translational part of \( G \) does not influence the result of the rotational part of \( FG \) or \( GF \).

Furthermore, the structure of the sub-matrices implies that

\[
\det([R_G]) = \det([R_G]) = ((g^1)^2 + (g^2)^2 + (g^3)^2 + (g^4)^2)^2. \quad (4.46)
\]

Requiring that to be equal to 1 means enforcing the condition that \( G \) be a unit versor (see (2.36)) and reduces the number of degrees of freedom for the rotational part to 3.

### 4.3. Solution to Component Versors

In this section we will present a way of inferring the component elemental motions comprising a kinematic chain. At the heart of our method are the observations of the instantaneous motion of the last link of the kinematic chain from one instant to the next. For a given number \( N \) of links specific solutions exist which we will present in the following. In all cases our solution is based on solving systems of linear equations, which can be done exactly or — in the presence of observation noise and numerical calculation errors — approximately, using least squares methods such as the singular value decomposition (SVD).
4.3 Solution to Component Versors

4.3.1. Zero Elemental Motions

For the trivial case of \( N = 0 \) we will observe no motion at all and the instantaneous motion \( ^1R_i \) equals the identity transformation for all instants \( i \).

4.3.2. One Elemental Motion

If \( N = 1 \) we need two instantaneous motions, say \( ^1R_1 \) and \( ^1R_2 \). Since we assume the elemental motions to be generated by constant speed bivectors, (4.11) implies that \( ^1R_1 = ^1R_2 = E_1 \), i.e. we observe the only component elemental versor directly. This was also shown as an example in (4.8).

4.3.3. Two Elemental Motions

For \( N = 2 \) we start out using observations at two instants. In equation (4.10) we let \( k = 1, i = 1 \) and obtain

\[
^1R_2 = 2^2R_2^1R_1 (2^1R_1)^{-1},
\]

\[
^1R_2^2R_1 = 2^2R_2^1R_1,
\]

(4.47)

with

\[
2^1R_1 = 2^2R_2 = E_2,
\]

(4.48)

by (4.11). Now we subtract the left hand side of (4.47) and use matrix formulations (4.38) and (4.42) to get

\[
0 = ^1R_2E_2 - E_2^1R_1
\]

\[
= \begin{pmatrix}
[R_1R_2]_l & 0 \\
[M_1R_2]_l & [R_1R_2]_l
\end{pmatrix}
E_2 - \begin{pmatrix}
[R_1R_1]_r & 0 \\
[M_1R_1]_r & [R_1R_1]_r
\end{pmatrix}
E_2
\]

\[
= \begin{pmatrix}
[R_1R_2]_l - [R_1R_1]_r & 0 \\
[M_1R_2]_l - [M_1R_1]_r & [R_1R_2]_l - [R_1R_1]_r
\end{pmatrix}
E_2.
\]

(4.49)

We focus on the rotational part of \( E_2 \) first and use the upper left corner of the matrix (4.49) to solve for it. The upper left \( 4 \times 4 \)-sub-matrix of (4.49) only has rank 2. This follows from the fact that \( ^1R_1 \) and \( ^1R_2 \) have the same grade-0 component (see (4.36)) and the form of (4.39) and (4.43). Geometrically, solving the upper left part of (4.49) is equivalent to finding the rotation between two 3D vectors from their chord. See Figure 4.6 for a visualization of the situation. Figure 4.2 depicts a similar problem in 2D. There is a 1-parameter family of possible rotations, which — together with the scaling freedom (we have not yet enforced that \( E_2 \) should be a unit versor) — leads to a 2-parameter family of possible solutions for the rotational part of \( E_2 \).

We obtain an additional independent condition by observing an additional instantaneous motion for the next instant \( i = 3 \). We can stack two matrices of the same form and obtain the
Figure 4.6.: Because the grade-0 parts of $^{1}R_{1}$ and $^{1}R_{2}$ are equal, we only have the bivector chord between their Euclidean parts (here visualized by the chord between Euclidean vectors) to infer the rotation relating them. There is a 1-parameter family of possible rotations, indicated here as the family of rotation planes containing the chord.

rotational part of the true $E_{2}$ as lying in the one-dimensional null space of the $8 \times 4$-matrix

$$
\begin{pmatrix}
[R_{1}R_{2}]_{l} - [R_{1}R_{1}]_{r} \\
[R_{1}R_{3}]_{l} - [R_{2}R_{2}]_{r}
\end{pmatrix},
$$

(4.50)

whose rank is 3. The last degree of freedom in the solution can be resolved by normalizing the vector representing the rotational part of $E_{2}$.

Now that we know the rotational part of $E_{2}$ we know its rotation axis/angle $\omega_{E_{2}}$ and therefore the rotation matrix $[R_{E_{2}}]$. We use (4.10) and (4.11) with $k = 1$ and $i = 1$, as well as (4.36) in order to write

$$
^{1}R_{2} = 2R_{2}^{1}R_{1}(^{2}R_{1})^{-1}
$$

$$
^{1}R_{2} = E_{2}^{1}R_{1}(E_{2})^{-1}
$$

$$
^{1}R_{2} = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & [R_{E_{2}}] & 0 & 0 \\
0 & [k_{E_{2}}] \times [R_{E_{2}}] & [R_{E_{2}}] & 0 \\
0 & 0 & 0 & 1
\end{pmatrix}^{1}R_{1},
$$

(4.51)

which we can re-write as a non-homogeneous system of linear equations in the unknown components of $k_{E_{2}}$. Introducing the notations $^{1}R_{i}|_{Eu}$ for the vector of purely Euclidean 2-blades
of \(1^R_i\) and \(1^R_i|_{n_\infty}\) for the vector of 2-blades of \(1^R_i\) which contain \(n_\infty\), we let

\[
a_i = [R_{E_2}|1^R_i|_{E_1}], \quad (4.52)
\]
\[
b_i = [R_{E_2}|1^R_i|_{n_\infty}], \quad (4.53)
\]
\[
c_i = 1^R_{i+1}|_{n_\infty}, \quad (4.54)
\]

and, using the fact that \([k_{E_2}] \times [R_{E_2}|1^R_i|_{E_1}] = - ([R_{E_2}|1^R_i|_{E_1}] \times k_{E_2})\), we obtain the 6 \(\times\) 4-matrix

\[
\begin{pmatrix}
- [a_1]_x & b_1 - c_1 \\
- [a_2]_x & b_2 - c_2
\end{pmatrix},
\quad (4.55)
\]

whose rank is 3 and whose one-dimensional null space provides a solution for \(k_{E_2}\) up to a homogeneous degree of freedom. We normalize the last coordinate to obtain a unique solution.

With (4.35) and the known quantity \(\omega_{E_2}\) we can solve for the pitch \(p_{E_2}\), the translation vector \(t_{E_2}\) and the offset \(u_{E_2}\).

\[
p_{E_2} = -\frac{2\pi}{\|\omega\|} k_{E_2} \cdot \omega_{E_2}, \quad (4.56)
\]
\[
t_{E_2} = p_{E_2} \omega_{E_2}, \quad (4.57)
\]
\[
u_{E_2} = ([I] - [R_{E_2}])^{-1} (k_{E_2} - t_{E_2}), \quad (4.58)
\]

from which we can reconstruct the full versor \(E_2\).

Note that in practice the matrix \(([I] - [R_{E_2}])\) may not be invertible, and therefore \(u_{E_2}\) may not be uniquely defined by (4.58) — a consequence of the fact that \(u_{E_2}\) can be any point on the screw axis of \(E_2\) (see Figure 4.5). With \(k_{E_2}\) and \(t_{E_2}\) known, a practical way to obtain \(u_{E_2}\) is to find the one-dimensional null space of the matrix

\[
\begin{pmatrix}
[I] - [R_{E_2}] & -(k_{E_2} - t_{E_2}) \\
[0] & 0
\end{pmatrix},
\quad (4.59)
\]

which encodes (4.58) as well as the requirement that \(u_{E_2}\) is orthogonal to \(t_{E_2}\).

### 4.3.4. Three Elemental Motions

Now we let \(N = 3\). Again, we start with two instantaneous motions, by rewriting (4.10) as

\[
1^R_2 2^R_1 = 2^R_2 1^R_1
\]
\[
0 = 1^R_2 2^R_1 - 2^R_2 1^R_1
\]
\[
= \left( [R_{1R_2}]_l \quad 0 \\
[M_{1R_2}]_l \quad [R_{1R_2}]_l
\right) 2^R_1 - \left( [M_{1R_1}]_r \quad 0 \\
[R_{1R_1}]_r \quad [R_{1R_1}]_r
\right) 2^R_1
\]
\[
= \left( [R_{1R_2}]_l \quad 0 \\
[M_{1R_2}]_l \quad [R_{1R_2}]_l
\right) 2^R_1 - \left( [M_{1R_1}]_r \quad 0 \\
[R_{1R_1}]_r \quad [R_{1R_1}]_r
\right) 2^R_1
\]
\[
= \left( [R_{1R_2}]_l \quad 0 \\
[M_{1R_2}]_l \quad [R_{1R_2}]_l
\right) 2^R_1 - \left( [M_{1R_1}]_r \quad 0 \\
[R_{1R_1}]_r \quad [R_{1R_1}]_r
\right) 2^R_1
\]
\[
= \left( [R_{1R_2}]_l \quad 0 \\
[M_{1R_2}]_l \quad [R_{1R_2}]_l
\right) 2^R_1 - \left( [M_{1R_1}]_r \quad 0 \\
[R_{1R_1}]_r \quad [R_{1R_1}]_r
\right) 2^R_1
\]
\[
= \left( [R_{1R_2}]_l \quad 0 \\
[M_{1R_2}]_l \quad [R_{1R_2}]_l
\right) 2^R_1 - \left( [M_{1R_1}]_r \quad 0 \\
[R_{1R_1}]_r \quad [R_{1R_1}]_r
\right) 2^R_1
\]
\[
= \left( [R_{1R_2}]_l \quad 0 \\
[M_{1R_2}]_l \quad [R_{1R_2}]_l
\right) 2^R_1 - \left( [M_{1R_1}]_r \quad 0 \\
[R_{1R_1}]_r \quad [R_{1R_1}]_r
\right) 2^R_1
\]
\[
= \left( [R_{1R_2}]_l \quad 0 \\
[M_{1R_2}]_l \quad [R_{1R_2}]_l
\right) 2^R_1 - \left( [M_{1R_1}]_r \quad 0 \\
[R_{1R_1}]_r \quad [R_{1R_1}]_r
\right) 2^R_1
\]
\[
= \left( [R_{1R_2}]_l \quad 0 \\
[M_{1R_2}]_l \quad [R_{1R_2}]_l
\right) 2^R_1 - \left( [M_{1R_1}]_r \quad 0 \\
[R_{1R_1}]_r \quad [R_{1R_1}]_r
\right) 2^R_1
\]
\[
= \left( [R_{1R_2}]_l \quad 0 \\
[M_{1R_2}]_l \quad [R_{1R_2}]_l
\right) 2^R_1 - \left( [M_{1R_1}]_r \quad 0 \\
[R_{1R_1}]_r \quad [R_{1R_1}]_r
\right) 2^R_1
\]
\[
= \left( [R_{1R_2}]_l \quad 0 \\
[M_{1R_2}]_l \quad [R_{1R_2}]_l
\right) 2^R_1 - \left( [M_{1R_1}]_r \quad 0 \\
[R_{1R_1}]_r \quad [R_{1R_1}]_r
\right) 2^R_1
\]
\[
= \left( [R_{1R_2}]_l \quad 0 \\
[M_{1R_2}]_l \quad [R_{1R_2}]_l
\right) 2^R_1 - \left( [M_{1R_1}]_r \quad 0 \\
[R_{1R_1}]_r \quad [R_{1R_1}]_r
\right) 2^R_1
\]
\[
= \left( [R_{1R_2}]_l \quad 0 \\
[M_{1R_2}]_l \quad [R_{1R_2}]_l
\right) 2^R_1 - \left( [M_{1R_1}]_r \quad 0 \\
[R_{1R_1}]_r \quad [R_{1R_1}]_r
\right) 2^R_1
\]

but now \(2^R_1 \neq 2^R_2\). We write \(2^R_i\) as a vector on the blade basis (4.28) as \(2^R_i = (2^r_i, 2^r_i, \ldots, 2^r_i)^T\) and observe that \(2^R_{i+1}\) emerges from \(2^R_i\) through the versor product by the elemental versor \(E_3\) (see (4.10), (4.11)), which means that its scalar part and its grade-4
part are preserved (4.36), i.e. $r^1_i = r^1_{i+1}$ and $r^8_i = r^8_{i+1}$.

As before, we focus on the rotational part first — and therefore the upper left sub-matrices in (4.60) — and obtain

$$0 = \left[ R_2 \right]_i \begin{pmatrix} r^1_i \\ r^2_i \\ r^3_i \\ r^4_i \\ r^5_i \\ r^6_i \\ r^7_i \\ r^8_i \end{pmatrix} - \left[ R_1 \right]_r \begin{pmatrix} r^1_r \\ r^2_r \\ r^3_r \\ r^4_r \\ r^5_r \\ r^6_r \\ r^7_r \\ r^8_r \end{pmatrix}. \quad (4.61)$$

This can be rewritten as a homogeneous system of linear equations in the components of $R_i$.

Let $[R]_c$ denote the $c$-th column of the matrix $[R]$. Then we have

$$0 = \left( \left[ R_i \right]_{i+1} \right)_l - \left[ R_i \right]_r \begin{pmatrix} 3 \cdot 4 \\ 3 \cdot 5 \\ 3 \cdot 6 \\ 3 \cdot 7 \end{pmatrix} \begin{pmatrix} r^1_i \\ r^2_i \\ r^3_i \\ r^4_i \end{pmatrix}. \quad (4.62)$$

The above $4 \times 7$-matrix has full row-rank and can be stacked for additional instants (while padding it with zeros appropriately). Since $r^1_i = r^1_r$ for all $i$, every additional observation adds three columns and four rows to the matrix, which continues to have full row-rank. Therefore, if we use only the matrix in (4.62) its null space is 3-dimensional. If we stack two matrices the null space is 2-dimensional and if we stack three matrices the system can be solved uniquely up to scale. The solution vector $\begin{pmatrix} r^1_1, r^2_1, r^3_1, r^4_1, r^5_1, r^6_1, r^7_1, r^8_1 \end{pmatrix}^T$ lies in the one-dimensional null space of the $12 \times 13$-matrix

$$\begin{pmatrix} \left[ R_2 \right]_l \begin{pmatrix} r^1_i \\ r^2_i \\ r^3_i \\ r^4_i \end{pmatrix} - \left[ R_1 \right]_r \begin{pmatrix} r^1_r \\ r^2_r \\ r^3_r \\ r^4_r \end{pmatrix} & 0 & 0 \\ \left[ R_3 \right]_l \begin{pmatrix} r^1_i \\ r^2_i \\ r^3_i \\ r^4_i \end{pmatrix} & 0 & \left[ R_1 \right]_r \begin{pmatrix} r^1_r \\ r^2_r \\ r^3_r \\ r^4_r \end{pmatrix} & 0 \\ \left[ R_4 \right]_l \begin{pmatrix} r^1_i \\ r^2_i \\ r^3_i \\ r^4_i \end{pmatrix} & 0 & \left[ R_1 \right]_r \begin{pmatrix} r^1_r \\ r^2_r \\ r^3_r \\ r^4_r \end{pmatrix} & 0 \end{pmatrix}. \quad (4.63)$$

Because the $[R_i]_l$ are orthogonal matrices the versors $R_i$ will have a common scale which can simply be adjusted to unity. With solutions to the rotational parts of $R_1$, $R_2$ and $R_3$ we can recurse to the case of $N = 2$ above (now substituting $R_i$ for the instantaneous motions) and find a solution to the rotational part of $R_i = E_3$, too.

We note that we have used 4 observations to derive the solution for the rotational parts of $R_i$ and $E_3$. But a counting argument for the degrees of freedom in the involved motions suggests that three observations should suffice. Both the rotational parts of $R_i$ and $E_3$ represent 3D rotations with 3 degrees of freedom each, while every correspondence between rotational parts of observed instantaneous motions $R_i$ and $R_{i+1}$ provides three conditions. Therefore we
present an alternative approach which works without knowledge of \(1R_4\). Deleting the last row and column block from matrix (4.63) we obtain a full rank \(8 \times 10\)-matrix with a 2-dimensional null space. Knowing that \(2R_1\) and \(2R_2\) are related by the same rotation as \(2R_2\) and \(2R_3\) we can impose an additional constraint on the solution. The bivector chords between the Euclidean bivector parts of \(2R_i\) and \(2R_{i+1}\) should have the same length, i.e.

\[
0 = \|2R_2 - 2R_1\|^2 - \|2R_3 - 2R_2\|^2
= \left(2R_1|_{Eu} - 2R_3|_{Eu}\right) : 2R_2|_{Eu}.
\]  

(4.64)

Algebraically, the last step can be verified by writing it out in terms of the components of the vector representation of \(2R_i\). Geometrically, it means that \(2R_2\) has to lie in the mid-plane between \(2R_1\) and \(2R_3\) (see Figure 4.7). Unfortunately, condition (4.64) is multilinear in the components of \(2R_i\) and cannot simply be stacked with matrix (4.62). After determining the 2-dimensional null space of the modified matrix (4.63), enforcing (4.64) boils down to finding the roots of a second degree polynomial, which yields two solutions. Both pose algebraically feasible solutions. The correct one has to be selected using additional constraints (which are not inherent in the problem description) or the observation of at least one additional instantaneous motion.

Now, with the rotational parts of \(2R_i\) and \(3R_i = E_3\) known, we focus on the remaining translational part, using only three instantaneous motions. Consider the last row block in

\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{figure4.7.png}
\caption{We visualize the Euclidean 2-blade parts of \(1R_i\) as 3D vectors. If the chords between \(1R_1|_{Eu}\) and \(1R_2|_{Eu}\), respectively between \(1R_2|_{Eu}\) and \(1R_3|_{Eu}\), have equal length then \(1R_2|_{Eu}\) (here a couple of candidates are shown) must lie on the mid-plane between \(1R_1|_{Eu}\) and \(1R_3|_{Eu}\).}
\end{figure}
Finding Articulated Structures from Motion of the End-Frame

\[ (4.60) \]

\[ 0 = \left( \left[ M_{1R_2} \right]_t \right) \left[ R_{1R_2} \right]_t \right)^2 R_1 - \left( \left[ M_{1R_1} \right]_r \left[ R_{1R_1} \right]_r \right)^2 R_2. \]  

(4.65)

At the same time, with (4.10), (4.11) and (4.36) we have

\[ 2R_2 = 3R_1 R_1^{-1} \]

\[ = E_3^2 R_1 E_3^{-1} \]

\[ = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & [R_{E_3}] & 0 & 0 \\ 0 & [k_{E_3}] \times [R_{E_3}] & [R_{E_3}] & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}^2 R_1. \]  

(4.66)

Of this equation we also focus on the translational part only and obtain

\[ 2R_2 |_{n_\infty} = \left( \left[ k_{E_3} \right] \times [R_{E_3}] \left[ R_{E_3} \right] \right) \left( \begin{array}{c} 2R_1 |_{E_4} \\ 2R_1 |_{n_\infty} \end{array} \right). \]  

(4.67)

We let \([R]^z\) denote the \(z\)-th power of the matrix \([R]\) and use the following property of the cross product: for an orthogonal \(3 \times 3\)-matrix \([R]\) and 3D vectors \(a\) and \(b\), we have

\[ [R] (a \times b) = ([R] a) \times ([R] b). \]  

(4.68)

Then we can write the part of \(2R_i\) containing \(n_\infty\) more generally, for all \(i > 1\)

\[ 2R_i |_{n_\infty} = \left( \sum_{z=0}^{i-2} [R_{E_3}]^z \left[ k_{E_3} \right] \times [R_{E_3}]^{i-1-z} \left[ [R_{E_3}]^{i-1} \right] \right) \left( \begin{array}{c} 2R_1 |_{E_4} \\ 2R_1 |_{n_\infty} \end{array} \right) \]

(4.69)

which allows us to express \(2R_2 |_{n_\infty}\) in terms of the unknown quantities \(k_{E_3}\) and \(2R_1 |_{n_\infty}\) as follows. We let

\[ d_i = [R_{E_3}]^{i-1} 2R_1 |_{E_4}, \]  

(4.70)

and get

\[ 2R_2 |_{n_\infty} = \left( - [d_2] \times [R_{E_3}] \right) \left( \begin{array}{c} k_{E_3} \\ 2R_1 |_{n_\infty} \end{array} \right), \]  

(4.71)

or, more generally,

\[ 2R_i |_{n_\infty} = \left( \sum_{z=0}^{i-2} - [d_1] \times [R_{E_3}]^z \left[ [R_{E_3}]^{i-1} \right] \right) \left( \begin{array}{c} k_{E_3} \\ 2R_1 |_{n_\infty} \end{array} \right). \]  

(4.72)
For a more compact notation we define the $4 \times 7$-matrices

\[
B_1 = \begin{pmatrix} [0]_{4 \times 3} & [I]_{4 \times 4} \end{pmatrix},
\]
\[
B_i = \begin{pmatrix} -[d_i]_x - \sum_{z=1}^{i-2} [d_i]_x [R_{E_3}]^z \end{pmatrix} \begin{pmatrix} [R_{E_3}]^{i-1} \end{pmatrix}, \quad \text{for } i > 1.
\]

Substituting (4.71) into (4.65) and collecting the unknown quantities we obtain

\[
0 = [M_{12}]_l \begin{pmatrix} 2r_1^1 \\ 2r_2^1 \\ 2r_3^1 \\ 2r_4^1 \end{pmatrix} + \begin{pmatrix} 2r_1^5 \\ 2r_2^5 \\ 2r_3^5 \\ 2r_4^5 \end{pmatrix} - [M_{12}]_r \begin{pmatrix} 2r_1^2 \\ 2r_2^2 \\ 2r_3^2 \\ 2r_4^2 \end{pmatrix} - \begin{pmatrix} 2r_1^6 \\ 2r_2^6 \\ 2r_3^6 \\ 2r_4^6 \end{pmatrix} - [R_{E_3}]_r \begin{pmatrix} 2r_1^7 \\ 2r_2^7 \\ 2r_3^7 \\ 2r_4^7 \end{pmatrix}
\]

\[
= \begin{pmatrix} 2r_1^1 \\ 2r_2^1 \\ 2r_3^1 \\ 2r_4^1 \end{pmatrix} - \begin{pmatrix} 2r_1^2 \\ 2r_2^2 \\ 2r_3^2 \\ 2r_4^2 \end{pmatrix} + \begin{pmatrix} 2r_1^5 \\ 2r_2^5 \\ 2r_3^5 \\ 2r_4^5 \end{pmatrix} - \begin{pmatrix} 2r_1^6 \\ 2r_2^6 \\ 2r_3^6 \\ 2r_4^6 \end{pmatrix} - \begin{pmatrix} 2r_1^7 \\ 2r_2^7 \\ 2r_3^7 \\ 2r_4^7 \end{pmatrix}
\]

\[
= ([M_{12}]_l - [M_{12}]_r [R_{E_3}]) \begin{pmatrix} 2r_1^1 \\ 2r_2^1 \\ 2r_3^1 \\ 2r_4^1 \end{pmatrix} + ([R_{E_3}]_l B_1 - [R_{E_3}]_r B_2) \begin{pmatrix} k_{E_3} \\ 2r_1^5 \\ 2r_2^5 \\ 2r_3^5 \\ 2r_4^5 \end{pmatrix}.
\]

(4.75)

an inhomogeneous system of linear equations in the unknown $k_{E_3}$ and the translational part of $^2R_1$.

To solve this system, we need to stack two matrices of the form

\[
\begin{pmatrix}
[R_{E_3}]_l B_1 - [R_{E_3}]_r B_{i+1} \\
[M_{12}]_l [R_{E_3}]^{i-1} - [M_{12}]_r [R_{E_3}]^i
\end{pmatrix}
\]

and find their joint null space. It turns out, however, that this null space is two-dimensional. This is a consequence of the fact that the linear correspondence equations do not yet account for the versor nature of the solution. In order to make sure that the solution represents a screw motion versor, we need to constrain it further. In order to do that we place an internal consistency constraint on the vector representation of the versor $^2R_1$, which relates its translational part to its rotational part. From (4.29) we observe that for any 3D screw motion versor $R$ we have

\[
(R|_n)^2 = \left(-\frac{1}{2} \cos \left(-\frac{1}{2} \|\omega\|\right) t + \frac{1}{\|\omega\|} \sin \left(-\frac{1}{2} \|\omega\|\right) \epsilon \right)^2
\]

\[
= \frac{1}{4} \cos^2 \left(-\frac{1}{2} \|\omega\|\right) \|t\|^2 + \frac{1}{\|\omega\|^2} \sin^2 \left(-\frac{1}{2} \|\omega\|\right) \|\epsilon\|^2,
\]

(4.77)
because \( t \) and \( c \) are orthogonal to each other. On the other hand

\[
\|c\|^2 = \left( R|_{n_{\infty}} + \frac{1}{2} \cos \left( -\frac{1}{2} \|\omega\| \right) t \right) \frac{\|\omega\|}{\sin \left( -\frac{1}{2} \|\omega\| \right)}
\]

\[
= \frac{\|\omega\|^2}{\sin^2 \left( -\frac{1}{2} \|\omega\| \right)} \left( R|_{n_{\infty}} \right)^2 + \frac{1}{4} \cos^2 \left( -\frac{1}{2} \|\omega\| \right) \|t\|^2 + \cos \left( -\frac{1}{2} \|\omega\| \right) (R|_{n_{\infty}} \cdot t).
\]

Substituting this into (4.77) gives us

\[
\left( R|_{n_{\infty}} \right)^2 = \frac{1}{2} \cos^2 \left( -\frac{1}{2} \|\omega\| \right) \|t\|^2 + \left( R|_{n_{\infty}} \right)^2 + \cos \left( -\frac{1}{2} \|\omega\| \right) (R|_{n_{\infty}} \cdot t) \quad (4.78)
\]

\[
0 = \frac{1}{2} \cos^2 \left( -\frac{1}{2} \|\omega\| \right) \|t\|^2 + \cos \left( -\frac{1}{2} \|\omega\| \right) (R|_{n_{\infty}} \cdot t). \quad (4.79)
\]

Since \( t \) is a multiple of \( \omega \) it can be written as \( t = \|t\| \omega \). We use (4.29) to solve the last component, denoted \( r^8 \), for \( \|t\| \) and obtain

\[
\|t\| = -2 r^8 / \sin \left( -\frac{1}{2} \|\omega\| \right). \quad (4.80)
\]

Substituting this into (4.79) yields

\[
0 = \frac{2 \cos^2 \left( -\frac{1}{2} \|\omega\| \right)}{\sin^2 \left( -\frac{1}{2} \|\omega\| \right)} (r^8)^2 + 2 \frac{\cos \left( -\frac{1}{2} \|\omega\| \right)}{\sin \left( -\frac{1}{2} \|\omega\| \right)} \|\omega\| (R|_{n_{\infty}} \cdot \omega) r^8 + \frac{1}{\|\omega\|} (R|_{n_{\infty}} \cdot \omega). \quad (4.81)
\]

Note that the last step in (4.81) is possible only if \( \frac{\cos \left( -\frac{1}{2} \|\omega\| \right)}{\sin \left( -\frac{1}{2} \|\omega\| \right)} r^8 \neq 0 \). In that case (4.81) is an independent linear constraint on the components of the translational part of \( ^2 R_1 \) and therefore can be appended to matrix (4.76). Otherwise, two cases are possible. One is that \( R \) is a pure axis rotation, i.e. \( t = 0 \). The second case is that \( \cos \left( -\frac{1}{2} \|\omega\| \right) = 0 \). Both cases have the same consequence. Because of (4.29) they mean that

\[
R|_{n_{\infty}} = \frac{1}{\|\omega\|} \sin \left( -\frac{1}{2} \|\omega\| \right) c. \quad (4.82)
\]

Therefore \( R|_{n_{\infty}} \cdot \omega = 0 \) and (4.81) follows automatically, implying that \( r^8 = 0 \). Consequently, we can incorporate (4.81) in any case.

The final solution — up to a homogeneous degree of freedom — to \( k_{E_3} \) and the translational
part of $^2R_1$ arises from the null space of the matrix

\[
\begin{pmatrix}
[R_1 R_2]_l B_1 - [R_1 R_1]_r B_2 \\
[R_1 R_2]_l B_2 - [R_1 R_2]_r B_3 \\
0 0 0 \frac{1}{\omega^2 R_1} (\omega^2 R_1)^T \cos\left(-\frac{1}{2} \|\omega^2 R_1\|\right) \sin\left(-\frac{1}{2} \|\omega^2 R_1\|\right) \\
0
\end{pmatrix}
\begin{pmatrix}
[M_1 R_2]_l - [M_1 R_1]_r [R E_3]
\end{pmatrix}
\begin{pmatrix}
(2 r_1^1) \\
(2 r_1^2) \\
(2 r_1^3) \\
(2 r_1^4)
\end{pmatrix}.
\]

From it we can reconstruct the full elemental versor $E_3$ using the equivalents of (4.56), (4.57) and (4.58).

### 4.3.5. More Than Three Elemental Motions

In principle, solutions for the case that $N > 3$ are possible, but they become computationally very expensive. Starting out with equation (4.60) we consider the rotational part of $^2R_i$ only and obtain

\[
0 = [R_1 R_2]_l \begin{pmatrix}
2 r_1^1 \\
2 r_1^2 \\
2 r_1^3 \\
2 r_1^4
\end{pmatrix} - [R_1 R_1]_r \begin{pmatrix}
2 r_1^1 \\
2 r_1^2 \\
2 r_1^3 \\
2 r_1^4
\end{pmatrix}
\]

\[
= \left( [R_1 R_2]_l \begin{pmatrix}
2 r_1^1 \\
2 r_1^2 \\
2 r_1^3 \\
2 r_1^4
\end{pmatrix} - [R_1 R_1]_r \begin{pmatrix}
2 r_1^1 \\
2 r_1^2 \\
2 r_1^3 \\
2 r_1^4
\end{pmatrix}\right).
\]

The solution for the rotational parts of $^2R_i$ lie in the null space of the $8 \times 4$-matrix in (4.85) which has full rank. Stacking more correspondence equations does not reduce the dimension of the null space below 4, because every equation adds 4 rows and 4 columns. We can obtain a 4-parameter family of solutions to $^2R_i$, where $i = 1, \ldots, k + 1$ with $k$ correspondence equations. With this we can establish a 4-parameter family of correspondence equations for the next lower $N$ and keep recursing until we can impose constraints that let us solve for the built up number (in the order of $O(4^N)$) of parameters. Such constraints can be preservation of grade-0 and grade-4 components of one of the versors, the bivector chord constraint (4.64) or — in case
of the translational parts of the solution versors — the ability to express components of one versor in terms of components of another versor as in (4.72).

However, the exponential increase of the number of parameters with increasing $N$ makes computations prohibitively complicated very quickly. Moreover — and more importantly — finding the null space of a family of matrices involves finding the roots of a high degree polynomial in the parameters. In general, polynomial equations of degree five or higher do not have exact solutions. Given exact correspondence data, we suspect that the equations arising from our method do have exact solutions, but the practical relevance of that case is questionable.

4.4. Examining Assumptions and Degeneracies

In the previous sections we made a number of assumptions and presented a method to decompose an articulated motion into constituent elemental motions. Now we analyze the limitations of this method and consider the impact of those assumptions.

4.4.1. Theoretical Remarks

A very basic constraint is that we only considered rigid body motions. For an important class of practical problems such as motion capturing — and in view of the DASIS project — this poses little limitation. However, the conformal model of geometric algebra allows us to describe conformal transformations as versors, and our considerations in section 4.2.1 remain valid for this wider class of transformations. So, one could try and generalize our approach to composite general conformal transformations. But when it comes to analyzing the matrix algebra representation of the linear transformations that conformal versors represent, matters are more complicated. First of all, full conformal transformations of 3-dimensional Euclidean space have 10 degrees of freedom, so the vector representation of bivectors is 10-dimensional; versors generated by exponentiating such bivectors are represented by 16-dimensional vectors. Moreover the interactions between components of these versors are more involved. For example, a separation of the purely rotational part as suggested by (4.38) or (4.42) is not possible. The exact details of the matrix representations of conformal motion versors are outside the scope of this thesis.

Presuming an open kinematic chain is a rather weak constraint. When observing a rigid body undergoing a composite motion it is always possible to interpret it as part of an open kinematic chain. In a next iteration of applying our method this kinematic chain may change — for example, if the “base” of the chain is not at rest but in motion — and it is a matter of interpretation what this change signifies: whether it is a result of measurement noise, a systematic error or the result of a violation of other assumptions. In practice, additional observations (e.g. of other links) or prior knowledge may be used to factor additional motions out of the observed instantaneous motions before decomposing it using our method.

On the other hand, assumptions such as the structural invariance of the true motion and constant joint velocities are critical to our approach. All our derivations in sections 4.2 and 4.3 are tailored to these assumptions. If they are violated — that is, if the joints undergo an
accelerated motion or the structure of the kinematic chain changes — our method will not yield the correct solution. The case of kinematic chains containing an accelerated joint was tested with simulated data. The results are presented in appendix A and discussed briefly in section 4.4.2.

The assumption of equally long time intervals between observations is very probably fulfilled in most practical situations. In case of time intervals of different lengths involves bivector exponentials with possibly fractional exponents. Such bivector exponentials are easily obtained (see (4.2)). The progression of meta-motions that we have given for discrete, integral instants in (4.9), (4.10) and (4.11) is then structurally the same, but more involved and more difficult to verify and we omit it here.

A bigger practical obstacle arises from the fact that we obtained specialized solutions for a specific number \( N \) of assumed links. Given the observation of a constellation at two instants, for example, we can always find the instantaneous motion, i.e. a decomposition of the true motion into a single elemental motion. This is an exact algebraic solution to the problem of finding the transformation between the two constellations. But this solution is wrong if the true motion is more complicated. Given the observation of the constellation at an additional instant, we can apply our method to yield a decomposition into two elemental motions. This may still not reflect the true motion between instants, if that true motion is more complicated. Thus, if we apply the solution for \( N = 2 \) (section 4.3.3) to a case where \( N = 3 \) we will fail to obtain the correct elemental motions comprising the true motion. Does this mean that we need to know \( N \) beforehand?

In fact, the correspondence equations carry enough structure to determine \( N \) from them. If the true \( N \) is larger than the assumed \( N \), the null space obtained from the respective solution matrices (e.g. (4.50) or (4.63)) has more than one dimension, while if the true \( N \) is smaller than the assumed \( N \) the null space is empty.

We have presented exact cases above. This means that if the correspondence equations are known exactly and to infinite precision, then (barring degeneracies, see below) they exactly determine a unique solution. It is well known that observation noise or numerical inaccuracies can result in an inconsistent set of equations which yields no simultaneous solution. In those cases the well understood singular value decomposition (SVD) gives a least squares approximation to the solution. Finding those singular values which are close to zero (and the corresponding singular vectors) gives an approximation even to the higher-dimensional null space of a matrix.

Finally, our method is sensitive to geometric degeneracies. We have tacitly assumed that the kinematic chain we recover is in a general configuration. That is, each link performs a general screw motion over time, whose parameters are independent of the parameters of all other links. In reality, for example when trying to recover the physical configuration of a robotic arm, this may not be the case.

In general, geometric degeneracies arise when some degrees of freedom are not expressed in

\footnote{It is possible, even though extremely unlikely, that observation noise or numerical inaccuracies lead to two or more equations becoming linearly dependent. In that case the system of equations would have multiple linearly independent solutions.}
an observation or when degrees of freedom in one elemental motion can be subsumed in another elemental motion. A simple concrete example for the latter is the case $N = 2$ when both $E_2$ and $E_1$ are pure translations, say, along translation vectors $t_{E_2}$ and $t_{E_1}$, respectively. The composite motion is a pure translation along translation vector $t_{E_2} + t_{E_1}$ and appears as the case $N = 1$, which could be factored into an arbitrary number of pure translations. Consider a similar case (see Figure 4.8 for a visualization) where $N = 2$, $E_1$ is a screw motion with screw axis/angle given by $\omega_{E_1}$ and $E_2$ is a pure translation along translation vector $t_{E_2}$. Then any component of $t_{E_2}$ along $\omega_{E_1}$ can be subsumed in the translational part of $E_1$ and the result will be indistinguishable from the true motion $^0R = E_2E_1$. Moreover, since the rotational part of $E_2$ is the identity transformation, the solution will appear as the case $N = 1$ when restricted to the rotational part of the instantaneous motions but as the case $N = 2$ when solving for the translational parts (given a correct partial solution for the single rotational part). Note that — rather than invalidating our method — such geometric degeneracies cause the problem to be ill posed in the specific cases when they occur. In these cases they make a unique solution fundamentally impossible.

![Figure 4.8: A simple geometric degeneracy. The two composite motions $^0R = E_2E_1$ on the left and $^0R' = E_2'E_1'$ on the right are completely indistinguishable from each other. $E_1'$ subsumes the translational component of $E_2$ in the direction of $\omega_{E_1}$.](image)

### 4.4.2. Empirical Evaluation

In order to evaluate the operating range of our method and to validate its practical applicability we have run experiments with simulated data. We used point features from which instantaneous motions were estimated using the Procrustes method. These estimates were used as input for the decomposition method presented above. A detailed description of the experimental setup along with a brief discussion and quantitative results can be found in appendix A.

For random configurations, including a fair amount of (near) degeneracies, with three links the method works reasonably well for small observation noise of up to one percent on point features, if enough points are used to ensure stable Procrustes estimates. Random configurations of only two links extend the working range to observation noise levels of up to two percent on point features. If particularly stable configurations are chosen, such that geometric degener-
cies are precluded, the working range of the method extends even further to observation noise levels of up to five percent on point features. However, with high noise levels it is advisable to only use few (e.g. up to 12) instantaneous motions.

We simulated cases where the $N$-th joint undergoes an accelerated rather than a constant speed motion. For small acceleration parameters up to $a = 0.1$ and up to 6 observed instantaneous motions (but no observation noise on the point features) the results are comparable to the cases with small noise but no acceleration. For more details, we refer the reader to appendix A.

What is notable is that even though the absolute error made in inferring the individual elemental motions may be large, the alignment of true (i.e. simulated) points with points moved by the composite of the estimated elemental motions can be fairly accurate. A deeper understanding of the mutual relationships between feature observations, instantaneous motions, true motion and elemental motions as well as an appropriate distance metric between rigid body motions would improve the interpretation of those results but is left as future research.

4.5. Conclusion

We have introduced a method to decompose a nested 3D rigid body motion into its constituent elemental screw motions. The method infers the structure of the composite motion from observations of instantaneous motions of a constellation of correspondence data, which is assumed to be rich enough to determine a unique rigid body motion from it.

In order to derive the method we place a number of assumptions on the problem. The most limiting constraints are the assumption of constant joint velocities over the whole period of observation and the assumption of an invariant structure of the true motion. Because our method works with a very small number of observations — if exact data is available, roughly $N$ observations are needed to infer a chain of $N$ elemental motions — the period over which these parameters need to remain invariant is small.

In non-degenerate cases, the structure of the correspondence equations allows us to completely infer the structure of the composite motion as well as the parameters of the component motions. However, observation noise and numerical calculation errors may obfuscate this structure. For example, a homogeneous system of equations may turn out not to have a solution or the null space of a matrix may lose a number of dimensions. In those cases the singular value decomposition (SVD) can yield an algebraic least squares approximation to the solution. Note that the least squares criterion applies to components of versors and is not directly geometrically motivated.

With the recursive method introduced in this chapter we are able to infer up to three component motions. Beyond that, solutions are possible but computationally expensive as they involve symbolically solving for (or approximating) null spaces of multi-parameter families of matrices. The number of parameters is exponential in the number of components of the composite motion and an explicit solution involves finding the roots of high degree polynomials.

Finally, we have exposed a number of geometric degeneracies to which our method is vulner-
able. Such degeneracies arise with fundamental ambiguities — such as the fact that there is no unique decomposition of a vector into a sum of vectors — or when a degree of freedom is not expressed in the observations. They cause the problem to be ill posed and preclude a unique solution irrespective of the method used. In practice, observations of intermediary motions, e.g. of links between the base and the last link of a kinematic chain, may be used to resolve those degeneracies.

Our method can be used to approximate a complicated motion by the product of a number of screw motions. In computer vision and motion capturing it may provide a way to deal with missing data by inferring articulated structures from few observations.

Beyond the theoretical derivations presented in this chapter, we have validated our method and quantified its accuracy in a number of preliminary experiments with simulated data. The results of these experiments along with a brief discussion of their results is presented in appendix A.
In this chapter we present a Bayesian approach to the parameter estimation of a hypersphere, given noisy point observations on that sphere. The Bayesian approach forces to make explicit all assumptions and beliefs about the problem at hand in the form of prior probabilities. We identify these assumptions — in our specific case isotropic Gaussian zero mean noise with given standard deviation on the point observations, as well as uniform distribution of the observations over the estimated hyperspherical surface — and give mathematical, geometrical and practical justifications for them. As a result we obtain an explicit, closed form formula for the likelihood of the parameters “location of center” and “radius” of a hypersphere in arbitrary dimensions.

In a Bayesian setting such a likelihood formula would be used as a departure for further inference or reasoning. Here we calculate the likelihood’s gradient and apply gradient descent to obtain a maximum likelihood estimate (MLE) for 3D spheres. We compare this estimator to a non-linear least squares estimator (which differs from our maximum likelihood estimator in that it implicitly minimizes the observation noise) and analyze its behavior in various setups, including varying degrees of violation of our assumptions.
5.1. Motivation

The maximum likelihood estimator (MLE) for hyperspheres that we propose here includes very classical results in 2D circle estimation but extends to spheres in 3D and beyond. In fact, by giving a closed form likelihood function, we enable extension to a full Bayesian approach to the problem. Instead of singling out a maximum, the Bayesian approach retains the likelihood function, which can be used for further reasoning. For instance, one can infer typical parameter values using probability mass.

More precisely, we assume $N$ data points in an $n$-dimensional Euclidean space. As these points may themselves be the result of some measurement or estimation process we take them to be observations degraded by noise, their true positions being unknown. Out of all possible hyperspheres on which the true points could lie, a maximum likelihood estimate selects the one that has the highest probability of having generated the observed data, given certain characteristics of the observation noise. We will assume this noise to be Gaussian and isotropic, i.e. the observations are normally distributed around the true points with no preferred direction. Furthermore, we assume the variance of the observation noise to be known (though not necessarily the same for all points). This is not an unrealistic assumption because — whether the observations are measurements by a physical sensor or the result of some precursory calculation — often, one has some idea about its magnitude.

This chapter is structured as follows. In section 5.2 we give a short historically motivated overview about the subject at hand, including a brief review of relevant literature. In section 5.3 we will try to explain what is meant here by the “Bayesian approach” and place our method within that context. The main contribution is presented in section 5.4 while in section 5.5 we discuss its results. Finally, section 5.6 contains concluding remarks and an outlook to future work. Appendices B and C at the end of this thesis provide some detailed numerical results as well as mathematical background for expressing derivations in this chapter.

5.2. Background

Maximum likelihood estimation provides an intuitive framework for estimating model parameters from observed data. However, care has to be taken when it comes to modeling the random variables involved or describing the statistical behavior of the obtained estimates.

An early step on the way to a hypersphere estimator is a circle estimator. A circle is a hypersphere in two dimensions and a number of researchers have addressed the problem of estimating it from uncertain point data. Much disagreement seems to exist, however, regarding how to set up a maximum likelihood circle estimator or how to interpret its properties. We will give a brief overview and comparison of different approaches.

Kiefer and Wolfowitz [W56] present a very systematic and general approach to maximum likelihood estimation. We will adopt their notation in this chapter. Let $X_i$, $i = 1, \ldots, N$, be vectors of random variables. Each of them has $n$ components $X_{ij}$, $j = 1, \ldots, n$. Specific values (or realizations) $x_{ij}$ of these will be called observations or data. Let their probability density
function be \( f(x|\theta, \alpha_i) \), where the so-called structural parameters \( \theta \) are used to parametrize some model that the data adheres to, while the vectors of incidental parameters \( \alpha_i \) determine the distribution of the data.

To further structure the problem, one can assume that the \( \alpha_i \) are not arbitrary constants but independently and identically distributed (i.i.d.) random variables with distribution function \( G(\alpha_i|\nu) \), where \( \nu \) parametrizes the distribution. Formally, \( G \) is a collection of random variables and is called a stochastic process. See Figure 5.1 for a Bayesian network visualizing these notions. The way one chooses to view the \( \alpha_i \) — whether to regard them as unknown constants, as i.i.d. random variables or something else — determines the behavior and properties of the resulting maximum likelihood estimator.

Kiefer and Wolfowitz come up with a number of regularity assumptions on \( f \), which have to hold in order for the maximum likelihood estimator to be consistent. Proving these assumptions depends on the problem at hand and properties of the space in which \( G \) is assumed (or allowed) to lie.

An early step on the way to a hypersphere estimator is a circle estimator. A circle is a hypersphere in two dimensions and a number of researchers have addressed the problem of estimating it from uncertain point data. Much disagreement seems to exist, however, regarding how to set up a maximum likelihood circle estimator or how to interpret its properties. We will give a brief overview and comparison of different approaches.

Having introduced the very general notation used in \( [W56] \) we will now focus on other authors’ work with maximum likelihood circle estimation in 2D. Obviously, the model in this case is the common circle on which the true data points lie. Its parameters (i.e. radius and location of the center) are the structural parameters \( \theta \). There exists any number of possible incidental parameters \( \alpha \), but the most common ones are those describing the observation noise, e.g. mean and variance of the respective random variables modeling the observations.

### 5.2.1. Structural Model vs. Functional Model

Given a circle on which some true 2D points lie, their observations can be considered as realizations of random variables, which can be described in the established terminology by

\[
X_{i1} = \theta_1 + \theta_3 \cos \alpha_{i3} + \alpha_{i1}, \quad (5.1)
\]

\[
X_{i2} = \theta_2 + \theta_3 \sin \alpha_{i3} + \alpha_{i2}, \quad (5.2)
\]
A Bayesian Approach to Hypersphere Estimation

Figure 5.2: The Gaussian Cartesian functional model for circle estimation

where $\alpha_{i3}$ parametrize the distribution of the true points over the circle with center $(\theta_1, \theta_2)$ and radius $\theta_3$, while $\alpha_{i1}$ and $\alpha_{i2}$ parametrize the noise on the point coordinates (see Figure 5.2). Berman and Culpin [C86] classify different approaches to the incidental parameters. Firstly, they assume that $\alpha_{i1}$ and $\alpha_{i2}$ are Gaussian random variables with zero mean.

Then they cite Chan [Ch65], who assumed that $\alpha_{i3}$ were fixed but unknown angles. This they call the Gaussian Cartesian functional model, since the observations are Cartesian coordinates bound by the functional relationship based on $\alpha_{i3}$. An estimation method would attempt to estimate the $\alpha_{i3}$. Instead of taking them to have fixed but unknown values, the $\alpha_{i3}$ can be regarded as independently distributed random variables. Berman and Culpin [C86] call this the Gaussian Cartesian structural model, because nothing is assumed about the distribution of the true points and only the structural parameters $\theta$ are estimated, not the $\alpha_{i3}$.

The difference between the two is the difference between a random variable and a realization. The structural model describes the true points by a probability distribution which may require any number of additional parameters. It can be implemented in different manners by constraining the possible distribution of $\alpha_{i3}$ in certain ways [An81, Be83]. In the functional model each true point is treated as a single (albeit unknown) realization of that random variable which is usually estimated along with the structural parameters.

We observe here that the Bayesian approach blurs the boundaries between the two models as both of them arise out of different assumptions about prior distributions and the decision which variables are treated as data instead of parameters and which variables are marginalized over. We will treat these issues in more detail in section 5.3.

5.2.2. Cartesian Model vs. Radial Model

Another way of classifying the problem statement is by assuming noise not on the Cartesian coordinates of the observations but on their polar coordinate representation. The following is called the radial model [C86].

\[
\begin{align*}
X_{i1} &= \theta_1 + \alpha_{i1} \cos \alpha_{i2}, \\
X_{i2} &= \theta_2 + \alpha_{i1} \sin \alpha_{i2},
\end{align*}
\]
with various constraints on the incidental parameters. For example, Berman and Culpin claim that it is usually assumed that $E[\alpha_i] = \theta_3$, the circle’s radius (which is saying that the radius can be modeled as a true value $\theta_3$ with additive zero mean noise). If the $\alpha_i$ are regarded as independently distributed random variables, Berman and Culpin call it the radial structural model.

5.2.3. Maximum Likelihood vs. Least Squares

When it comes to fitting a model to observed data, least squares methods are very popular, because they are straightforward to implement, intuitive and often give good results. Moreover, it is a common misconception that least squares estimates are equivalent to maximum likelihood estimates under the assumption of independent Gaussian noise. This is not generally true but rather depends very much on the model used.

A maximum likelihood estimator $\{\hat{\theta}_{ML}, \hat{\alpha}_{ML}\}$ tries to maximize the joint likelihood of the structural and possible incidental parameters $\theta$ and $\alpha$ that explain all the observations.

$$\{\hat{\theta}_{ML}, \hat{\alpha}_{ML}\} = \arg\max\{\theta, \alpha\} L_{x_1, \ldots, x_N}(\theta, \alpha), \quad (5.5)$$

where the $L_{x_1, \ldots, x_N}(\theta, \alpha)$ denotes the conditional probability $p(x_1, \ldots, x_N|\theta, \alpha)$ and depends on the specific functional relationship between the structural and possible incidental parameters $\theta$ and $\alpha$, and the observations $x_i$.

By contrast, a least squares estimator $\{\hat{\theta}_{LS}, \hat{\alpha}_{LS}\}$ tries to minimize the sum of squared distances between the observations $x_i$ and the model $f$, which depends on the structural parameters $\theta$, but also on the assumed fixed but unknown true values of the incidental parameters $\alpha_i$. The true values $\alpha$, which are constrained by the structural parameters $\theta$, are estimated along with the structural parameters.

$$\{\hat{\theta}_{LS}, \hat{\alpha}_{LS}\} = \arg\min\{\theta, \alpha\} \sum_i^N (x_i - f(\theta, \alpha))^2 \quad (5.6)$$

For 2D circle estimation, the least squares estimator setting is described by the functional models above. A classical relevant example of this procedure is found in [Ch65], where a functional relationship between Cartesian coordinates $X_{i1}(\theta, \alpha_i) = \theta_1 + \theta_3 \cos \alpha_i$ and $X_{i2}(\theta, \alpha_i) = \theta_2 + \theta_3 \sin \alpha_i$ is used to describe a circle, whose parameters $\theta_i$ are then estimated along with the true points (or rather their true angular parameters $\alpha_i$).

Under the assumption of Gaussian isotropic noise, the above makes for equivalence of max-
imum likelihood and least squares methods in a certain sense as the sum of squared distances
between observations \( x_i \) and estimated true points \((X_{i1}, X_{i2})\) is minimized. However, the least
squares estimator also *implicitly* minimizes the amount of noise in the observations. This is
behavior that the user of a least squares estimator may not have asked for. If the amount of
noise can be known, this behavior is even detrimental to the estimate. Additionally, as each
observation \( x_i \) adds a parameter (or degree of freedom) \( \alpha_i \) to the least squares estimate, it
becomes more vulnerable to overfitting the data.

5.2.4. Consistency of MLE in General and in Particular

A very common quality measure for an estimator is *consistency*. That means that the expect-
ation of the estimator tends towards the true value of the estimated quantity as the amount
of data \( N \) tends towards infinity. As mentioned above, Kiefer and Wolfowitz [W56] propose a
number of regularity assumptions under which the maximum likelihood estimator is consistent.
Their proof pertains to maximum likelihood estimation in general and is not limited to circle
or hypersphere estimation.

Chan [Ch65] points out that for the Gaussian Cartesian functional model of circle estima-
tion, where \( X_{i1} \) and \( X_{i2} \) are normally distributed, a necessary and sufficient condition for the
maximum likelihood estimator to be consistent is that both \( 1/N \sum \cos \alpha_{i3} \) and \( 1/N \sum \sin \alpha_{i3} \)
tend to zero as \( N \to \infty \). Geometrically, this assumption is fulfilled if the data is uniformly dis-
tributed over the whole circle. Anderson [An81] remarks that attempts to establish a consistent
estimator for the incidental parameters \( \alpha_{i3} \) introduces instability of the maximum likelihood
estimator if the support of \( \alpha_{i3} \) is less than half the circle. She, too, reverts to the assumption
that the observations are approximately uniformly distributed over the circle.

5.2.5. Efficiency of the MLE

Also, Anderson [An81] establishes formulas for the variance of the estimator for the circle’s
radius and for the variance of the noise in the observations. Moreover, she brings up the
asymptotic relative efficiency (ARE) of the maximum likelihood estimator for the circle’s cen-
ter. This ratio compares the capability of two estimators to establish the same confidence
in the estimate. We will omit the mathematical details here and only report on Anderson’s
results.

Under the assumption of uniform distribution an alternative consistent estimator for the
circle’s center is the arithmetic mean of the observations’ locations. Both estimators, the
MLE estimator and the arithmetic mean, would eventually (i.e. given an infinite number of
observations) find the true value. But as long as the amount of data is not infinite they are
both afflicted by uncertainty which depends on the amount of data \( N < \infty \) seen so far.

Comparing the uncertainty of these two estimators, Anderson comes to the conclusion that
the ARE of the maximum likelihood estimator depends on the ratio of observation noise \( \sigma^2 \)
to the circle’s radius \( \rho^2 \). Anderson reports an ARE of 0.991 at \( \sigma^2/\rho^2 = 1.0 \), dropping to 0.810 at
\( \sigma^2/\rho^2 = 0.3 \) then to 0.358 at \( \sigma^2/\rho^2 = 0.1 \) and down to 0.039 at \( \sigma^2/\rho^2 = 0.01 \). By definition of
the ARE, this means that the maximum likelihood estimator needs only 99.1% (respectively 81%, 35.8% and 3.9%) of the data to achieve the same level of uncertainty/confidence as the estimator derived from the observation mean.

5.3. A Bayesian Approach

We now assume a Bayesian standpoint. It should have become clear that, from a Bayesian point of view, there are a number of decisions that every author of a statistical parameter estimator makes, explicitly or — more often — implicitly. In this section we will introduce the general Bayesian approach while pointing out the decisions we make along the way for our specific hypersphere estimator and explain why we make them. Finally we will arrive at a likelihood formula which can be computed explicitly, yielding an estimate for the hypersphere parameters.

Formally, the likelihood maps a set of structural and incidental parameters to the conditional probability density of the observations given these parameters, i.e.

\[(\theta, \alpha) \mapsto L_x(\theta, \alpha) = p(x|\theta, \alpha).\] (5.7)

We would like to stress the fact that the likelihood is a function of the parameters \(\theta\) and \(\alpha\), not of the data \(x\) — whereas \(p(x|\theta, \alpha)\) denotes the probability of observing \(x\) given the parameters \(\theta\) and \(\alpha\). Here we will use the notation of \(L_x(\theta, \alpha)\) (read: the likelihood of the parameters \(\theta\) and \(\alpha\), given the data \(x\)) instead of \(L(\theta, \alpha)\). Sometimes — especially in the context of maximum likelihood estimation — the likelihood function is only of interest up to a multiplicative constant.

Now we regard the incidental parameters \(\alpha\) not as arbitrary given constants but as random variables governed by some (possibly unknown) stochastic process \(G(\alpha|\nu)\) parametrized by \(\nu\). In the terminology above this means that we are assuming a structural model, where the true values can only be described by a random variable of which the observations are realizations. Taking this into account the likelihood is described by

\[L_{x,\alpha}(\theta, \nu) = p(x|\theta, \alpha)G(\alpha|\nu) \quad = \quad p(x, \alpha|\theta, \nu).\] (5.8)

We assume a structural model because treating the incidental parameters as fixed values which need to be estimated would introduce additional degrees of freedom to our estimator which then would become more prone to overfitting the data. Note that at this point (5.8) we have abandoned the idea of estimating the true values \(\alpha_i\). Instead we estimate the parameters \(\nu\) of the process governing their probability distribution.

Anderson [An81] differentiates four cases of estimation problems with respect to the process \(G\). Firstly, the process \(G\) could be (assumed to be) a uniform distribution. Secondly, \(G\) could be known except for a fixed number of parameters \(\nu\), for instance \(G\) could be a normal distribution with unknown mean and variance. Thirdly, \(G\) could be unknown, in which case it
lies in a function space parametrized by $\nu$ and can be estimated consistently \[W56\]. In practice, however, it may be infeasible to estimate $\nu$ — and thereby $G$ — without further constraining the space in which $G$ is allowed to lie. Finally, if nothing is known about the incidental parameters, one could try to eliminate them from the functional relationship and end up with a likelihood that does not involve a $G$. For the functional circular model, Anderson says that it is provably impossible to eliminate the incidental parameters. Therefore, this approach is also not an option for our hypersphere estimator. Alternatively, in order to avoid introducing $G$ altogether, one could treat the incidental parameters as unknown constants to be estimated along with the structural parameters, which brings us back to the functional model and to the initial formulation of the likelihood (5.7).

In what we would like to call the “full Bayesian approach” it is even possible to introduce prior probabilities on the possible distributions $G$. This is very similar to the third case quoted above. Instead of strictly constraining the function space, these prior probabilities are used to guide the search for $G$. If one is interested in estimating only the structural parameters, though, one will marginalize the prior distribution out of the likelihood formula.

$$L_{x,\alpha}(\theta) = \int p(x|\theta, \alpha)G(\alpha|\nu)p(\nu)\,d\nu$$
$$= \int p(x|\theta, \alpha)G(\alpha, \nu)\,d\nu$$
$$= p(x|\theta, \alpha)G(\alpha)$$
$$= p(x, \alpha|\theta),$$

where, for brevity’s sake, we extended the notation of probability densities to that of the stochastic process in a straightforward way.

With these remarks we are now finally in a position where we can formally specify the model we use for our hypersphere estimator. Instead of estimating the distribution of the incidental parameters or even marginalizing it out of our likelihood function, we will assume a certain distribution $G$.

Our goal is to fit a hypersphere to point data. We assume independent, isotropic Gaussian noise on the Cartesian coordinates of $N$ point observations. In many real world applications, a Gaussian distribution is a realistic — or at least very common — assumption, validated by the central limit theorem. The means and variances of these Gaussian distributions make up the incidental parameters of our estimator. We assume the variances to be known. As said before, in many applications this is the case, and in our setting we only want to estimate the structural (hypersphere) parameters. With the noise characteristics known we have the variance(s) $\sigma_i^2$, $i = 1, \ldots, N$ given. The incidental parameters that are left are the means of the point observations. Without any further information we assume that they are uniformly distributed over the surface of the hypersphere-to-be-estimated; we do not have an a priori reason to assume a concentration around some point. Because we are not interested in estimating the incidental parameters, we will marginalize them out of our likelihood formula.

For a single point observation $x_i$, given the hypersphere $s(\theta_c, \theta_r)$ with structural parameters
radius $\theta_r$ and location of center $\theta_c$, the true point $\alpha_i$ could lie anywhere along $s$ with equal probability. The observation is a realization of a normally distributed random variable with mean $\alpha_i$ and known variance $\sigma^2_i$. Then (5.10) becomes

$$L_X(\theta) = \int p(X, \alpha | \theta) \, d\alpha.$$  

(5.11)

In our specific case, we start out from (5.9) and marginalize out the incidental parameters.

$$L_X(\theta) = \int L_{x_i, \alpha_i, \sigma}(\theta_c, \theta_r, \alpha_i) \, d\alpha_i \, d\sigma$$

$$= \int p(x_i | \theta_c, \theta_r, \alpha_i, \sigma) \, G(\alpha_i, \sigma) \, d\alpha_i \, d\sigma$$

$$= \int p(x_i | \theta_c, \theta_r, \alpha_i, \sigma) \, U_s(\theta_c, \theta_r)(\alpha_i) \, \delta(\sigma - \sigma_i) \, d\alpha_i \, d\sigma$$

$$= \int \int \left[ \mathcal{N}(x_i | \alpha_i, \sigma^2_i) \right] \, d\alpha_i \, d\sigma.$$  

(5.12)

Note that one fundamental assumption we make is that there is a common hypersphere to be estimated. This means, given a set of structural parameters $\theta$, the probability of the true points lying anywhere but on the indicated hypersphere is zero. We use $U_s(\theta_c, \theta_r)(\alpha_i)$ to denote the uniform distribution of the $\alpha_i$ restricted to the hypersphere $s$ parametrized by $\theta_c$ and $\theta_r$. Therefore, the integral over the whole observation space becomes the integral over that hypersphere. We use the Dirac delta $\delta(\sigma - \sigma_i)$ as a probability distribution that is zero for all $\sigma \neq \sigma_i$ to model the fact that we know the value of $\sigma_i$. We will explicitly evaluate the integral (5.12) in the next section. But first, having placed our estimator in the context of Bayesian reasoning, some remarks are in order.

Given a closed form formula for the likelihood of the structural (and incidental) parameters, one can employ standard methods — such as gradient descent or the Newton-Raphson method — to single out its maximum. However, the Bayesian approach has several advantages. First of all, depending on the further data processing chain, it is not always necessary to come up with a single estimate for the involved parameters. Rather the likelihood formula provides a means to assign a probability value to each possible set of parameters. Additionally, in many cases the maximum likelihood estimate may not represent a typical outcome. In this respect probability mass is more indicative than the magnitude of the likelihood (which may have a very high, yet very narrow peak), and it can only be determined by integrating the likelihood over a certain range. Furthermore, even if the main interest lies in a specific estimate, one can compute the variance of the estimate, which gives a clue about the confidence in it. This can be very useful in order to base further decisions on the result or in keeping track of the error build-up in processing uncertain data. Also, given a closed form of the likelihood, it is possible to marginalize out certain parameters that may be uninteresting in specific applications (known as nuisance parameters) or to incorporate prior knowledge about the distribution of parameters.
Figure 5.3: All 3D points $\alpha_i$ lying on the indicated 1D sub-hypersphere (circle) with radius $\theta_r \sin \tau_i$ have the same distance to the observation $x_i$ and contribute the same probability of having generated it. Note that the triangle joining $\theta_c$, $x_i$ and $\alpha_i$ does not, in general, contain any right angles.

5.4. Evaluation of the Closed Form Likelihood

Before we write out the integral (5.12) we shall give some geometrical intuition regarding the derivation of the result. See Figure 5.3 for an example in three dimensions. Imagine a straight line from the hypersphere’s center $\theta_c$ through the observed point $x_i$. The intersection point of that line with the hypersphere is one of many candidates for the true point that generated the observation. From $\theta_c$, a candidate $\alpha_i$ and the observed point $x_i$ are viewed under an angle $\tau_i$. Now, because the assumed observation noise is isotropic, we can express the probability of $\alpha_i$ having generated $x_i$ purely in terms of their mutual distance $d$, which only depends on $\tau_i$. This means that a rotation of the configuration about the imagined line joining $\theta_c$ and $x_i$ does not change the probability. More specifically, every point on a hypersphere with one dimension less and a radius that only depends on $\tau_i$ contributes the same probability. Therefore, as we let the viewed angle $\tau_i$ run from 0 to $\pi$ we integrate the probabilities over the whole hypersphere as in (5.12).

Indeed, this observation, which greatly simplifies calculations, is the main reason why we assume both isotropic noise on the observations, as well as uniform distribution of the true points around the hypersphere. With anisotropic noise the probability of an $\alpha_i$ having generated $x_i$ would not only depend on their distance, but also on their direction. While — even under isotropic noise — without uniform distribution of $\alpha_i$ a rotation about the imagined line would change that probability.

The squared distance, $d(\tau_i)^2$, between the observation $x_i$ and $\alpha_i$ is

$$d(\tau_i)^2 = (\|\theta_c - x_i\| - \theta_r \cos \tau_i)^2 + (\theta_r \sin \tau_i)^2$$

$$= \|\theta_c - x_i\|^2 + (\theta_r \cos \tau_i)^2 - 2 \|\theta_c - x_i\| \theta_r \cos \tau_i + (\theta_r \sin \tau_i)^2$$

$$= \|\theta_c - x_i\|^2 + \theta_r^2 - 2 \|\theta_c - x_i\| \theta_r \cos \tau_i. \quad (5.13)$$

Let $n$ be the dimensionality of the space of observed points. The hypersphere we are trying to estimate will have dimension $n - 1$, whereas the hypersphere of true points which could have generated the observation for fixed $\tau_i$ has dimension $n - 2$ and radius $(\theta_r \sin \tau_i)$. The surface
area of a hypersphere of co-dimension 2 is given by the following expression.

\[
A(n - 2, \theta_r \sin \tau_i) = (n - 1) \frac{\pi^{n-1}}{\Gamma\left(\frac{n+1}{2}\right)} (\theta_r \sin \tau_i)^{n-2}
\]

\[
= \frac{2\pi^{n-1}}{\Gamma\left(\frac{n-1}{2}\right)} (\theta_r \sin \tau_i)^{n-2},
\]

(5.14)

where \(\Gamma\) denotes the gamma function and we used the fact that \(\Gamma\left(\frac{n+1}{2}\right)/(n-1) = \frac{1}{2}\Gamma\left(\frac{n-1}{2}\right)\).

Since we assume the observation noise to be Gaussian, our integral (5.12) evaluates to

\[
L_{x_i, \sigma_i}(\theta_c, \theta_r) = \int_0^{\pi} \frac{1}{\sqrt{(2\pi \sigma_i^2)^n}} \exp\left(\frac{-d(\tau_i)^2}{2\sigma_i^2}\right) A(n - 2, \theta_r \sin \tau_i) d\tau_i
\]

\[
= \int_0^{\pi} \frac{1}{\sqrt{(2\pi \sigma_i^2)^n}} \exp\left(-\frac{||\theta_c - x_i||^2 + \theta_r^2}{2\sigma_i^2} - 2\sqrt{||\theta_c - x_i||^2 \theta_r \cos \tau_i}\right)
\]

\[
\cdot \frac{2\pi^{n-1}}{\Gamma\left(\frac{n-1}{2}\right)} (\theta_r \sin \tau_i)^{n-2} d\tau_i
\]

\[
= \frac{1}{\sqrt{(2\pi \sigma_i^2)^n}} \exp\left(-\frac{||\theta_c - x_i||^2}{2\sigma_i^2} - \frac{\theta_r^2}{2\sigma_i^2}\right) \frac{2\pi^{n-1}}{\Gamma\left(\frac{n-1}{2}\right)} \theta_r^{n-2}
\]

\[
\cdot \int_0^{\pi} \exp\left(\frac{\sqrt{||\theta_c - x_i||^2 \theta_r \cos \tau_i}}{\sigma_i^2} \sin \tau_i\right) (\sin \tau_i)^{n-2} d\tau_i
\]

\[
= \frac{\sqrt{\pi}}{\Gamma\left(\frac{n-1}{2}\right)} \frac{\Gamma\left(\frac{n-1}{2}\right)}{\Gamma\left(\frac{n}{2}\right)} \theta_r^{n-2}
\]

\[
\cdot \frac{\sqrt{\pi}}{\Gamma\left(\frac{n}{2}\right)} \frac{\Gamma\left(\frac{n}{2}\right)}{\Gamma\left(\frac{n-1}{2}\right)} \theta_r^{n-2}
\]

\[
= \frac{2\theta_r^{n-2}}{\sqrt{(2\pi \sigma_i^2)^n}} \exp\left(-\frac{||\theta_c - x_i||^2 + \theta_r^2}{2\sigma_i^2}\right)
\]

\[
\cdot \frac{n!}{\pi^{n/2} (4\sigma_i^2)^n} \theta_r^{n-2}
\]

(5.15)

where \(\theta F_1\) denotes the confluent hypergeometric limit function (see Appendix C).

We will assume the observations \(x_i\) to be statistically independent. In order to investigate a simplified case of practical relevance in detail, we will also assume the noise level for each observation to be the same, i.e. \(\sigma_i = \sigma\), \(i = 1, \ldots, N\). The joint likelihood for \(N\) point
observations is therefore given by

$$L_{x,\sigma}(\theta_c, \theta_r) = \prod_{i=1}^{N} L_{x_i}(\theta_c, \theta_r)$$

$$= \left( \frac{2\theta_r^{n-2}}{(2\sigma^2)^n \Gamma \left( \frac{n}{2} \right)} \right)^N \exp \left( -\frac{N\theta_r^2 + \sum_{i=1}^{N} \|\theta_c - x_i\|^2}{2\sigma^2} \right) \prod_{i=1}^{N} \frac{n}{\sigma^2} F_1 \left( \frac{n}{2}, \frac{\|\theta_c - x_i\|^2}{\sigma^2} \right).$$ \hspace{1cm} (5.16)

Maximum likelihood estimation in this case is obviously highly non-linear in the structural parameters to be estimated. In order to employ, say, a gradient descent method to search for the maximum, one needs to know the gradient of the likelihood function. We therefore calculate the first derivatives with respect to \(\theta_c\) and \(\theta_r\), respectively.

$$\nabla_{\theta_c} L_{x,\sigma}(\theta_c, \theta_r) = L_{x,\sigma}(\theta_c, \theta_r) \sum_{i=1}^{N} \left( -\frac{\theta_c - x_i}{\sigma^2} + \frac{(\theta_c - x_i)\theta_r^2}{n\sigma^4} F_1 \left( 1 + \frac{n}{2}, \frac{\|\theta_c - x_i\|^2\theta_r^2}{4\sigma^4} \right) \right),$$ \hspace{1cm} (5.17)

$$\nabla_{\theta_r} L_{x,\sigma}(\theta_c, \theta_r) = L_{x,\sigma}(\theta_c, \theta_r) \left( \frac{N(n-2)}{\theta_r} - \frac{N\theta_r}{\sigma^2} \right) \right. \left. + \sum_{i=1}^{N} \frac{\theta_r \|\theta_c - x_i\|^2}{n\sigma^4} F_1 \left( 1 + \frac{n}{2}, \frac{\|\theta_c - x_i\|^2\theta_r^2}{4\sigma^4} \right) \right),$$ \hspace{1cm} (5.18)

where we have used \((\partial/\partial x) F_1(n/2, x) = 2 F_1(1 + n/2, x)/n\).

The fact that the gradient is a multiple of the joint likelihood works to our advantage. Firstly, when looking for zeros of (5.17) and (5.18), we are looking for parameters for which either the sum term or \(L_{x,\sigma}(\theta_c, \theta_r)\) is zero. Since the likelihood is essentially a probability it will not be zero at its maximum, so we only need to find parameters for which the sum term is zero. From a computational point of view, focusing only on the sum yields a numerically much more well behaved gradient. Additionally, the joint likelihood is very small, such that multiplying by it could introduce numerical instability. Usually this problem is tackled by taking the logarithm of the joint likelihood (yielding the log-likelihood), such that, instead of having to evaluate a large product as in (5.16), one evaluates a large sum.

5.5. Results

The first test that our formula has to pass is the following. Does it agree with the previously known likelihood formula for 2D circle estimation? If we substitute \(n = 2\) into (5.16) we get

$$L_{x,\sigma}(\theta_c, \theta_r) = \left( \frac{1}{2\sigma^2} \right)^N \exp \left( -\frac{N\theta_r^2 + \sum_{i=1}^{N} \|\theta_c - x_i\|^2}{2\sigma^2} \right) \prod_{i=1}^{N} I_0 \left( \frac{\sqrt{\|\theta_c - x_i\|^2\theta_r^2}}{\sigma^2} \right).$$ \hspace{1cm} (5.19)
where \( I_0 \) denotes the modified Bessel function of the first kind (see also Appendix C). This agrees — up to a multiplicative constant — with the likelihood formula cited by Anderson, who gives the log-likelihood for the circle’s center in [An81].

Of special interest to us is the case of \( n = 3 \), because it is of geometric importance in many applications and the highest dimension which is still easily visualizable. The explicit likelihood formula becomes

\[
L_{x,\sigma}(\theta_c, \theta_r) = \left( \frac{4\theta_r}{\sqrt{(2\sigma^2)^3\sqrt{\pi}}} \right)^N \exp \left( -\frac{N\theta_r^2 + \sum_{i=1}^{N} \| \theta_c - x_i \|^2}{2\sigma^2} \right) \prod_{i=1}^{N} \frac{\sigma^2 \sinh \left( \frac{\sqrt{\| \theta_c - x_i \|^2 \theta_r}}{\sigma^2} \right)}{\sqrt{\| \theta_c - x_i \|^2 \theta_r}}
\]

\[= \left( \frac{4\sigma^2 \theta_r}{\sqrt{2^4 \pi \sigma^6 \theta_r}} \right)^N \exp \left( -\frac{N\theta_r^2 + \sum_{i=1}^{N} \| \theta_c - x_i \|^2}{2\sigma^2} \right) \prod_{i=1}^{N} \frac{\sinh \left( \frac{\sqrt{\| \theta_c - x_i \|^2 \theta_r}}{\sigma^2} \right)}{\sqrt{\| \theta_c - x_i \|^2 \theta_r}} \right) \prod_{i=1}^{N} \frac{\sinh \left( \frac{\sqrt{\| \theta_c - x_i \|^2 \theta_r}}{\sigma^2} \right)}{\sqrt{\| \theta_c - x_i \|^2 \theta_r}}.
\]

(5.20)

Using (5.20) we implemented a maximum likelihood estimator for the sphere parameters in order to investigate its accuracy and its efficiency under varying conditions. Among other things we are interested in the behavior of the maximum likelihood estimator when the assumption of uniform distribution which we used to derive (5.20) is not reflected by observations on a true sphere. To this end we have run a number of experiments with simulated data. The simulated true points were either uniformly distributed over the surface of the sphere or they were distributed according to a normal distribution with varying standard deviations. In detail, we set up our experiments as follows.

We assume a true 3D sphere with radius 10 centered at the origin. On this sphere we sample a number of points by introducing a reference plane which can rotate around the \( y \)-axis through an angle that we randomly sample from a uniform distribution between \(-\pi/2\) and \(\pi/2\). From this plane we determine an azimuth angle sampled from a normal distribution with zero mean and a standard deviation which we call the spread parameter \( q \). Should this latter sample lie outside the interval \([-\pi, \pi]\), we just discard it. The fact that the resulting points are not samples of a normal distribution does not matter. Our main focus is on obtaining some concentration of the sampled points, not on their specific distribution. If it falls inside the interval we accept the resulting point and add isotropic Gaussian noise with zero mean and variance \( \sigma^2 \) to its three Cartesian coordinates. See Figure 5.5 for typical sample distributions.

We repeat this process \( N \) times and take the \( N \) resulting points as input for the likelihood formula (5.20). We find a maximum by gradient descent using the gradient formulas (5.17) and (5.18).

In addition, we perform an experiment where the reference plane is rotated through a random angle uniformly distributed between \(-\pi\) and \(\pi\) and the azimuth angle distributed according to \( \cos \varphi \) with \( \varphi \) between 0 and \( \pi/2 \). This results in a distribution of sample points over the
A solid sphere that is imaged under (approximately) parallel projection. Points on the half sphere turned away from the sensor are not visible due to self occlusion. Points at the “apex”, closest to the sensor, have a higher chance to be observed than points on the “equator”. More precisely the probability of a point being observed is distributed according to \( \cos \varphi \) on the interval of \( \varphi \in [0, \pi/2] \).

By rotating this figure out of the drawing plane we obtain the setup in 3D described in the text.

surface of half the sphere with higher chance of occurring near the “apex” and lower chance of occurring near the equator. The setup emulates a situation that occurs in practice when a solid sphere is imaged under (approximately) parallel projection, for example, by a camera, a laser range finder or a similar device. In these cases half the sphere — namely the part turned away from the sensor — is not visible and it is more likely to observe points on the sphere surface where the sphere normal points in the direction of the sensor. See Figure 5.4 for a visualization of this setup.

For our experiments we vary \( N \in \{4,10,20,100\} \), \( q \in \{0.1,0.2,0.5,1\} \) and \( \sigma \in \{0.1,0.5,1,2,5\} \). For each of the resulting \( 4 \times 5 \times 5 \) cases we perform 5000 independent runs. Additionally, we perform the same experiment with sample points uniformly distributed around the true sphere, varying \( N \) and \( \sigma \).

As a measure of the quality of our estimator we calculate its bias, i.e. the difference between the expectation of the estimated parameters \( \hat{\theta} \) and their true value \( \theta_0 \).

\[
\text{bias}[\hat{\theta}] = E[\hat{\theta}] - \theta_0
\]

Since we are estimating multiple parameters, this will be a vector. We will calculate the magnitude, \( b \), of that vector using the Euclidean metric. Other measures of quality are the trace, \( \text{tr} \), as well as the determinant, \( \text{det} \), of the covariance matrix \( \Sigma(\hat{\theta}) = \text{cov}(\hat{\theta}) \). Instead of calculating them analytically, we approximate all these values by the sample mean and the sample covariance of the 5000 runs.

For comparison we implemented a non-linear least squares estimator for the sphere parameters \( \theta \), which uses gradient descent to minimize the sum of squared orthogonal distances between the observed points and the estimated sphere. We will denote results pertaining to the maximum likelihood estimator (respectively, the non-linear least squares estimator) by a subscript MLE (respectively, subscript NLS). For detailed numerical results see Appendix B.
5.5 Results

(a) point spread $q = 0.1$

(b) point spread $q = 0.2$

(c) point spread $q = 0.5$

(d) point spread $q = 1.0$

Figure 5.5.: 20 points sampled from a sphere at the origin with radius 10. The points’ spread around the point $(0, 10, 0)$ on the sphere varies. The sampled points have added Gaussian noise with standard deviation $\sigma = 0.1$ on their Cartesian coordinates.

In the tables summarizing the results, we find several things worth remarking on. Firstly, we see that both estimators exhibit a bias that is more severe the more concentrated the sample points are on the sphere. As the spread parameter increases, the bias reduces significantly with an increasing number of points. We would like to point out that this is not an artifact of the type of estimators we used (i.e. maximum likelihood vs. least squares estimator), but of the assumptions that we made. More precisely, we assumed that the sample points are uniformly distributed around the sphere and therefore our estimator will try to estimate a sphere around
A Bayesian Approach to Hypersphere Estimation

Figure 5.6: Some estimates. The blue sphere is the true sphere of radius 10 from which the observed points were sampled. The red sphere indicates the non-linear least squares estimate, while the green one shows the maximum likelihood estimate. Images are scaled to fit the respective plot.

which the points are uniformly distributed. In Figure 5.6 on the right we see the effect of this assumption.
5.5 Results

Figure 5.7.: The estimators' bias as a function of the point spread parameter for 20 points around a sphere at the origin with radius 10. The bias has been calculated using the sample mean of 5000 independent estimations.

The images on the left of Figure 5.6 do not exhibit this behavior that drastically, because we have very low noise. The observed points provide stronger evidence for the true hypersphere they were sampled from. The amount of noise is another assumption that we make, and in Figure 5.6 we can clearly see those two assumptions compete. On the left the low noise assumption dominates the estimate, while on the right the assumption of uniform distribution takes over.

Note that while the bias of the two estimators can differ by as much as a factor of five (Table B.1, spread 0.1, 100 points, $\sigma = 0.5$) they are usually of the same order of magnitude. In general, the maximum likelihood estimator seems to lead to a slightly higher bias. This trend reverses (and is particularly pronounced when the number of point observations is small), when the points are indeed uniformly distributed around the sphere (compare the b-values in individual cells of Table B.5), which serves to show the influence of the violated assumption in the other cases. Figure 5.7 shows the bias as a function of the degree to which the assumption is violated. The bias has been calculated empirically, using the sample mean of 5000 runs.

The determinant of the covariance matrix is a measure for the hypervolume of the covariance ellipsoid. Therefore we expect it to be growing with increasing dimensionality, in general. The least extreme difference between the two compared estimators we found is a factor of about 1.17 in favor of the maximum likelihood estimator (Table B.4, spread 1.0, 4 points, $\sigma = 0.5$), but usually the two differ by many orders of magnitude, which can be as extreme as $10^{28}$ (Table B.1, spread 0.1, 100 points, $\sigma = 0.5$).

Numerical results for the experiment with points distributed according to a cosine function (recall Figure 5.4) can be found in table B.6. As one might expect, in practice as well as in our simulation experiments this particular distribution is similar to our artificial distribution described earlier, but with a spread parameter between $q = 0.5$ and $q = 1.0$. The numerical results support this intuition.

The empirical results seem to indicate that the more extreme the concentration of sample points the larger the difference between the determinants of the covariance matrices becomes. But this explanation is a little too simplistic. Since — due to the assumption of a uniform
distribution of points — the concentration is not “seen” by either estimator, we believe that the influencing parameter is the ratio of assumed noise versus the radius of the estimated hypersphere. As mentioned before, higher noise reduces the evidence of the observed points in favor of the true sphere. This would also reflect Anderson’s results regarding the asymptotic relative efficiency (ARE) which we summarized in section 5.2.5.

The relationship becomes most apparent in Figure 5.8, where we plot the determinants of the respective estimator’s covariance matrix on a logarithmic scale. We ordered the plot by ascending quotient \(\sigma^2/\rho^2\), where \(\rho\) determines the mean radius of the estimated sphere. Figure 5.9 shows the asymptotic relative efficiency (ARE), \(\sqrt{\text{det}_{\text{NLS}}/\text{det}_{\text{MLE}}}\), of the maximum likelihood estimator on a logarithmic scale, ordered by the same criterion. Even though the horizontal axis is not linear, a trend becomes apparent. With an increasing noise-to-radius ratio, the absolute determinant of the covariance matrix decreases along with the ARE of the maximum likelihood estimator with respect to a least squares estimator.

5.6. Conclusion

In this chapter we have applied the Bayesian approach towards inference problems to the estimation of hyperspheres from point data. We have derived an explicit likelihood formula for the hypersphere parameters “radius” and “location of center” in \(n\) dimensions and calculated the likelihood gradient, which we used to obtain a maximum likelihood estimate for 3D spheres.

The Bayesian approach has several advantages, one of which is to force the author to make explicit all assumptions in the form of prior probabilities. Our assumptions about the esti-
5.6 Conclusion

Figure 5.9: The ratio of determinants $\sqrt{\frac{\det_{NLS}}{\det_{MLE}}}$, ordered by ascending quotient $\sigma^2/\rho^2$, with $\rho$ denoting the mean radius of the estimated sphere. Note, that the horizontal axis is not linear. It ranges from $10^{-4} \times 3.77$ to $10^{4} \times 7.053$ with about half the values $\leq 1$.

...mation problem are that observed points are degraded by zero mean isotropic Gaussian noise whose variance is known (and therefore does not have to be estimated) as well as uniform distribution of the observed points around the hypersphere to be estimated. We have shown that it is these assumptions — rather than the nature of the problem or the type of estimator — that are responsible for the observed bias of the estimator, if points are not uniformly distributed around the hypersphere.

These assumptions were made because either they are justifiable in many practical applications — e.g. Gaussian noise as a consequence of the central limit theorem — or because they greatly simplify the derivation of the formulas — e.g. uniform distribution and isotropic noise simplify the involved integrals.

Our empirical results indicate that, given the same amount of data, a maximum likelihood estimator has much lower uncertainty about the estimated hypersphere parameters than a non-linear least squares estimator. Among other factors, this is due to the fact that our estimator treats the amount of observation noise as given, whereas a least squares estimator tries to minimize that as part of the estimation process. Alternatively, a maximum likelihood estimator can reach the same level of uncertainty as a least squares estimator with a lower amount of data, i.e. it is more efficient.

In order to harvest the benefits of the full Bayesian approach in a specific application some modifications may have to be made. Firstly, it should be carefully determined which parameters are to be estimated and which parameters are incidental or should be regarded as nuisance parameters. Secondly, all prior knowledge (or belief) about the distribution of the parameters should be incorporated in the (likelihood) formulas. Note, that in some settings this can change “online” as more data is observed.

Finally, in this chapter we analyzed the behavior of a maximum likelihood estimator for hyperspheres, which results in a single output. However, if further decisions or estimates should be based on this output it is better not to home in on a single estimate prematurely.
Preferably one should retain the full likelihood function and propagate it properly through the estimation process.
Discussion and Conclusion

This thesis is concerned with geometric data processing. In this field practical problems are posed by the observation of geometrical data to which a geometrical model is fit. The observed data generally consists of various geometric entities, such as points, lines, planes etc. Usually, the data is afflicted with uncertainty that is introduced, for example, by the inaccuracy of physical sensors or previous calculations. The model is described by a number of simultaneous equations that have to be fulfilled by the entirety of the observed data. In the context of this thesis the model affords a geometric interpretation as another geometric entity — for example, a hypersphere — or a geometric transformation of the data.

In the formalism of classical linear algebra, two of the three problems treated in this thesis (namely in chapters 3 and 4) are of the following form: Given \( N \) vector correspondences \( v_i \leftrightarrow v'_i \), solve the equation

\[
Mv_i = v'_i, \quad i = 1, \ldots, N
\]  

(6.1)

for the matrix \( M \) with the additional constraint that \( M \) is orthogonal. We pointed out that standard linear algebra techniques indiscriminately use vectors to represent geometric objects such as points, planes or directions. In principle, the matrix \( M \) can change the type of object, but in practice correspondences are mostly given between objects of the same type.

In geometric algebra, where orthogonal transformations are represented by versors, the problem can be represented geometrically more faithfully, but that representation is algebraically more complicated. There, the correspondence data consists of \( N \) multivectors \( X_i \leftrightarrow X'_i \) of
grade $m_i$ and we have to solve the equation

$(-1)^m_i S X_i \hat{S}^{-1} = X'_i, \ i = 1, \ldots, N$

(6.2)

for the versor $S$. There are compelling reasons to turn to this formalism. Multivectors are better suited to represent geometric data because the type of the represented object is faithfully reflected in its algebraic properties. Moreover, versors act on the data in a structure (and grade) preserving manner and parametrize orthogonal transformations minimally. However, the position of the argument of the transformation is more awkward in equation (6.2). It is often more difficult to isolate the versor and solve for it.

We solved this difficulty in this thesis in particular instances of geometric data processing either by reducing the specific problem from CGA to a classical linear algebra problem (in chapter 3) or by translating the problem formulation from CGA to a classical linear algebra formulation (in chapter 4). We see the difference between the two approaches as follows. In the former case a crucial step is finding a linear transformation between frames of Euclidean direction vectors. This transformation can be found completely in the established terms of classical linear algebra. Only then is the representation of that transformation translated into geometric algebra to yield a closed form solution to the total problem. By contrast, in the latter case the solution is a transformation completely represented in geometric algebra as a set of versors. The constraints on these versors are an intricate set of linear equations which are solved using classical linear algebra techniques. But the result of these calculations is a set of versors. All questions of uncertainty or optimality are dealt with in the realm of geometric algebra and are not influenced by a conversion of representations from classical linear algebra.

Geometric data processing can be based on exact data and yield exact solutions. We believe the exact cases to be invaluable in order to prove or disprove the validity of an approach, explore what is theoretically possible, and provide ground truth for exact data. In this thesis we have developed exact solutions to particular problems. In practice, geometric data processing usually involves mechanisms for integrating ambiguous or contradictory data that arises from observation noise or uncertainty. We addressed this issue only tentatively in chapters 3 and 4. In chapter 5, where we estimated a hypersphere from noisy point data, we approached this matter directly and found that it poses a great difficulty. There, we used neither full CGA nor classical linear algebra to tackle the problem of observation noise. Instead we employed a Bayesian approach to obtain a geometrically motivated hypersphere estimator in an — albeit practically relevant — special case.

6.1. Conformal Transformations and Extended Vahlen Matrices

In chapter 3 we explained how to use exact correspondence data to determine a conformal transformation in $n$-dimensional Euclidean space. By counting the number of degrees of freedom we showed that a minimal set of correspondence data for determining such a transformation
may be made up by a set of Euclidean direction vectors anchored at a common point and an additional point in Euclidean space.

We represented conformal transformations by Vahlen matrices, a known representation of CGA versors by $2 \times 2$-matrices with entries from a Euclidean geometric algebra. In order to fulfill the properties of versors, which are the geometric product of a number of invertible vectors, Vahlen matrices have to fulfill a number of algebraic constraints. By relaxing these constraints one obtains extended Vahlen matrices which can be used to represent not only versors but all the elements of CGA including, in particular, conformal points, circles and tangents.

Employing extended Vahlen matrices, we solved the problem of determining conformal transformations of Euclidean space from correspondence data in closed form. We were able to reduce the problem to finding an orthogonal transformation between sets of Euclidean direction vectors and generalized the solution to arbitrary dimensions. Moreover, the required correspondence data consists only of geometrically meaningful quantities. The “bookkeeping” dimensions of CGA are treated as unknowable. In particular, as opposed to other methods that try to solve the same problem, the weight of conformal points does not have to be known.

There is a potential general advantage to using extended Vahlen matrices that does not just apply to our particular problem or choice of correspondence data, namely the reduction of CGA problems to problems in the purely Euclidean geometric (sub-)algebra. Extended Vahlen matrices represent the full CGA, but because their entries are purely Euclidean, during calculations the null elements never appear explicitly. This facilitates the role of geometric algebra as an interface to classical linear algebra. There, often a Euclidean vector basis is implicitly assumed and the Euclidean metric is implemented by default. As a result, software implementations of Euclidean geometric algebra can be reused. Theorems that have been proven for Euclidean geometric algebra apply to the entries of extended Vahlen matrices. No special attention needs to be paid to the Minkowski metric of the underlying vector space of CGA and its dimension is reduced by two: instead of $\mathcal{G}(\mathbb{R}^{n+1,1})$ one only needs to consider $\mathcal{G}(\mathbb{R}^n)$. On the other hand, potentially four entries of the Vahlen matrix have to be considered and calculated.

At first glance, one might interject that by using extended Vahlen matrices we give up the advantages of geometric algebra, such as the possibility to represent geometric objects and transformations in a coordinate free manner. But this is not so. Extended Vahlen matrices only structure the representation of CGA elements. Unlike classical linear algebra matrices, extended Vahlen matrices do not act upon vectors (which would result in the same typification issues that classical linear algebra is struggling with) but on other extended Vahlen matrices. They do not require the introduction of a particular basis on which their components are to be interpreted. The fact that there is an isomorphism between the representation by extended Vahlen matrices and the representation using an arbitrary vector basis makes that clear.

When it comes to the significance of conformal transformations, one may take the standpoint that they only come into play if one uses CGA, that their appearance is a “side effect” of using the conformal model. But we have presented convincing reasons to use CGA, namely
its all-purpose applicability, coordinate freeness and minimal parametrization of orthogonal transformations. From that standpoint conformal transformations would be interesting because they include rigid body motions and Euclidean similarities as special cases of unquestionable practical value. Moreover, conformal transformations warrant further study as the boundary cases of a subset of CGA restricted, for example, to rigid body motions. But conformal transformations are also interesting in their own right, for example for applications in physics. Functions that are governed by a potential are still governed by a potential after a conformal transformation. This makes CGA applicable in fluid dynamics, electromagnetic field theory and other similar fields.

6.2. Decomposing Composite Motions

In chapter 4 we introduced a method to decompose a composite 3D rigid body motion into its constituent elemental rigid body motions. Such composite motions occur, for example, at the end of an open kinematic chain where each of the links performs an elemental rigid body motion. In order to solve this problem we made some constraining assumptions. The most restrictive of those is that of constant speed of the constituent elemental motions during the whole period of observation.

We gave a formalized problem description on which we based our considerations. This formalization was based on the Lie group structure of the group of 3D rigid body motions, $SE(3)$, and was independent of the particular representation of that group. We obtained a solution using classical linear algebra methods, such as the determination of the null space of a matrix or an approximation of it using singular value decomposition (SVD). The important innovation that made a solution possible is that we first represented motions by CGA versors and only then found a classical linear algebra analogue for this representation. This intermediate step allowed us to parametrize 3D rigid body motions minimally. Minimal parametrization meant a minimal number of linear correspondence equations that we had to solve for the components of the CGA versors.

By contrast, using classical linear algebra right away comes at a great disadvantage. The classical matrix representation of rigid body motions is redundant and requires a number of algebraic constraints which may be non-linear or require iterative procedures to enforce. Depending on the concrete formulation of these constraints they may involve the use of (non-linear) sines and cosines or they require the matrix columns to be mutually orthogonal vectors implying quadratic constraints.

Previous attempts to recover articulated structure from the observation of features in motion usually use one of two main lines of approach. One is to set up a number of correspondence equations that are driven by geometric considerations, constraints and heuristics. Such constraints may be motivated by physics and may include the rigidity of the observed moving bodies or the connectedness of the individual links of the articulated structure. Sometimes these constraints are not linear and have to be obtained by methods other than linear algebra.

The other, more modern, approach is called factorization and is based on algebraic consid-
6.3 Bayesian Hypersphere Estimation

In chapter 5 we presented a Bayesian approach to the estimation of \( n \)-dimensional hyperspheres from noisy points in \( n \)-dimensional Euclidean space. We derived a closed form formula for the likelihood of the hypersphere parameters “radius” and “location of center”. In order to obtain this formula we made a number of assumptions about the distribution of the points and the noise characteristics. More precisely we assumed the points to be uniformly distributed over the surface of the hypersphere and the noise to be isotropic Gaussian noise.

Both are simplifications that designers of common circle and sphere estimators make, explicitly or implicitly. For example, easy to implement least squares estimators try to minimize the sum of squared distances between the observed points and the estimated sphere. Incidentally, they also minimize the (implicitly assumed isotropic Gaussian) observation noise. In the case of isotropic Gaussian noise, least squares estimators are equivalent to maximum likelihood estimators that treat the amount of observation noise as a parameter to be estimated. By contrast our approach treats the amount of observation noise as an input parameter. We believe that this better reflects practical situations where one often has some idea of the observation noise.
level. The assumption of uniform distribution is often even less warranted. It only becomes evident in the fact that maximum likelihood estimators are not consistent if this assumption is violated. In cases where the assumption of uniform distribution is made explicitly, the intention to avoid this inconsistency is given as a justification. The Bayesian approach forces the designer of an estimator to make these assumptions explicit in the form of prior knowledge and give justifications for them which are motivated by the problem rather than by the behavior of the solution. Both the Bayesian approach to maximum likelihood circle and sphere estimation from point data as well as the generalization to arbitrary dimensions are novel results.

Even though we did not use geometric algebra to describe the geometric data, the research presented in chapter 5 was inspired by open questions related to CGA. A proper, geometrically motivated model for noise or uncertainty in CGA is still missing and the classical linear algebra (implicitly Euclidean) noise models do not apply. For example, the vector representation of an uncertain conformal point should allow for uncertainty in the Euclidean components which represent the position of the point in Euclidean space. The additional ”bookkeeping” dimensions, however, should be constrained, because otherwise the vector may come to represent a conformal hypersphere with non-zero radius. When reasoning about uncertain points this constraint constitutes a form of prior knowledge. A good way to implement this and other kinds of prior knowledge is offered by the Bayesian approach.

In CGA, the outer product of conformal vectors represents higher grade geometrical objects such as circles or the outer product null space (OPNS, see section 2.4) representation of hyperspheres. Given uncertainty in the conformal vectors, representing the resulting uncertainty in these higher grade objects is difficult. The reason for that lies not only in the unusual behavior of the bilinear, anti-symmetric outer product or the unusual metric employed by CGA, but in the inherent geometric circumstances of the problem. Finding a way to formalize this problem in CGA warrants further research. But by describing it in purely Euclidean geometric terms we have shown how difficult this formalization is, algebraically. Moreover, we have obtained a ”gold standard” by which any future attempt to incorporate (isotropic Gaussian) noise into CGA can be evaluated.

6.4. Future Research

Using the example of some representative problems of practical relevance, we have demonstrated that geometric algebra is a useful mediator between geometric intuition and efficient and well-tried, linear algebra based implementation. However, geometric algebra is a relatively young research field and much of its territory remains uncharted.

Regarding the focus of this thesis an important question is how to represent and process noisy or uncertain data. In chapter 3 uncertain correspondence data would determine an uncertain conformal transformation. In chapter 4 uncertainty in the input instantaneous transformations leads to an uncertain estimate of the decomposition of a composite rigid body motion. The methods presented there offer work-arounds and ways to fit geometric models to noisy observations. For example, in chapter 3 uncertainty in the frame of Euclidean direction vectors is
handled by classical linear algebra methods on which we fall back for solving this sub-problem. In chapter 4 uncertainty in a composite rigid body motion is dealt with by employing the singular value decomposition to obtain a least squares estimate of the composite elemental rigid body motions. However, a framework that incorporates noise in a geometrically motivated as well as algebraically sound way is still lacking. For instance, in chapter 3 the rationale and geometric interpretation of the resulting fit of the Euclidean sub-problem can be transferred from the classical linear algebra methods used to solve it, but they do not extend in an obvious way to the full problem. In chapter 4 the solution is approximated in a least squares sense on the Euclidean vector representation of versors. This representation does not account for some of the finer points of versor estimation (see appendix A). Since in this thesis geometric algebra was used as an interface to classical linear algebra techniques, a first step may be to apply well-tested linear algebra noise models at the implementation level and analyze the geometric implications for the higher level geometric algebra descriptions.

Another interesting question is, how well geometric algebra interfaces with fields other than classical linear algebra. We have seen that geometric algebra offers a novel way of parameterizing geometric objects and transformations. Classical linear algebra offers one way of estimating these parameters from observation data, and we have analyzed its advantages and disadvantages. A promising alternative is the Bayesian approach to parameter estimation, which allows — even forces — to incorporate prior knowledge into the estimation process. The Bayesian approach is sometimes more complicated, mathematically, and can lead to mathematical models whose solution is considered infeasible. With advancements in the development of more and more powerful computer hardware and increasing understanding of probability theory the solution to many such problems is coming within reach. The mutual benefits of geometric algebra and Bayesian probability theory could be explored in greater detail.
Appendices
Simulation Experiments on Motion Decomposition

Here, we present some experiments on the theoretical results concerning the decomposition of composite rigid body motions obtained in chapter 4. Before we discuss the experimental results some general remarks are appropriate.

A.1. Issues with Uncertainty

The method introduced in chapter 4 decomposes a composite rigid body motion into its constituent elemental rigid body motions. However, it is difficult to determine how similar (respectively, different) two rigid body motions are. Much consideration has been given to this topic (e.g. [Par95, Pen98, Dub11]), often from a statistical point of view. We raise three major concerns here and present the choices we make in regard to them in section A.2.

- Non-Existence of a Bi-Invariant Distance Metric:

  Park [Par95] proves that there is no positive definite bi-invariant distance metric (i.e. a distance metric that is invariant under arbitrary coordinate transformations) on rigid body motions. In practice this has the consequence that one has to choose between a left-invariant and a right-invariant metric. A left-invariant metric is preserved under an arbitrary change of the inertial (i.e. “world”) coordinate frame, while a right-invariant
metric is preserved under a change of the body-fixed (i.e. “object”) coordinate frame. Park shows that the choice of a suitable metric also involves striking an arbitrary (or application dependent) trade-off between the difference in rotation and the difference in translation. In CGA, this choice is avoided by employing the bi-invariant inner product in a difference measure instead of a distance metric. But parts of the versors representing the rigid body motion are null blades. This means that — even though they do change the represented rigid body motions — they do not impact the difference measure based on the inner product.

- **Intrinsic Difference Measure:**

The group of rigid body motions forms a differentiable manifold that is not flat. A proper difference measure between rigid body motions would have to take into account the shape of that manifold by measuring differences along geodesics. This measure would require a whole backdrop of differential geometry involving differentiation and integration of curves in the rigid body motion manifold (see also [Dub11]).

- **Ambiguity of Geometric Configurations / Data Dependence of Difference Measure:**

Even an exact decomposition of a composite rigid body motion does not lead to a unique geometric configuration of a physical kinematic chain that created the observed motion. One way of handling the arising ambiguities is the well-known Denavit-Hartenberg convention [DH55]. But it is a convention for describing coordinate-based motion parameters of a *known* kinematic chain rather than for inferring a kinematic chain from coordinate-based motion parameters. And even then it leaves room for freedom of choice in parameters, for example in the case of parallel rotation axes of consecutive joints. Therefore, given only the estimate of coordinate-based motion parameters, it is not uniquely possible to reconstruct the physical kinematic chain that created the motion. Consequently, it is problematic to compare a kinematic chain estimated from those parameters with the true kinematic chain.

Related to this is the fact that the difference of rigid body motions depends on the objects that are transformed by them. Imagine a rotation through a fixed angle about a fixed axis in 3D passing through the origin. Consider a copy of this rotation with an axis that is rotated slightly about the origin. How different are these rotations from each other? A set of points close to the origin that is transformed by the original rotation and its copy will yield images that are not very different from each other. On the other hand, a set of points close to the original rotation axis but far away from the origin will yield images that are very far apart. Some rigid body motion estimation techniques (such as the well-known Procrustes analysis [GD04] or motor estimation using CGA [VD11]) therefore measure the quality of the obtained transformation by measuring the distance between the transformed and the observed data.
A.2. Experimental Setup

In view of these issues we have conducted experiments with simulated data in MATLAB as follows, hoping to approximate situations that may arise in practice. We generate \( N \) elemental motions represented by versors \( E_j \) which, in turn, are generated as exponentials of bivectors \( B_j \),

\[
E_j(t) = \exp(B_j(t)).
\]  

Then we construct the composite true motion for a number \( K + 1 \) of time instants

\[
0 R(i) = E_N(i) \ldots E_1(i), \quad i = 0, \ldots, K.
\]  

In the case of constant speed elemental motions, the parametrization of \( B_j \) by \( t \) in (A.1) is linear and the true motion (A.2) evaluates to

\[
0 R(i) = E_N^i \ldots E_1^i, \quad i = 0, \ldots, K.
\]  

Next, we generate a constellation \( X_0 \) consisting of a number \( P \) of points normally distributed with zero mean and a standard deviation of 1 around some assumed “end effector” at position \( \sum_{j=1}^{N} \omega E_j \). This position is somewhat arbitrary (see Ambiguity of Geometric Configurations above), but is supposed to provide some offset from the origin which depends on the true motion parameters. We transform this initial constellation by the true motion to obtain snapshots of the constellation at all time instants \( i = 0, \ldots, K \) as

\[
X_i = 0 R(i) X_0 \left(0 R(i)\right)^{-1}.
\]

We degrade each of the \( P \) points in each of the constellations by independent isotropic zero mean Gaussian noise with a variable standard deviation of \( \sigma \). This mimics isotropic observation noise on point features moving with the end frame of a kinematic chain. Then we use the MATLAB built-in function \texttt{procrustes()} to estimate the rigid body motion between constellations \( X_{i-1} \) and \( X_i \), precluding reflections and scaling. This yields an estimate \( \tilde{1} R_i \) for the instantaneous motion \( 1 R_i \) that transforms one constellation into the next by

\[
X_i = 1 R_i X_{i-1} \left(1 R_i\right)^{-1}.
\]

These estimates are used as input to the decomposition method introduced in chapter 4.

A number of parameters influences the quality of the decomposition results. Not all of them can be neatly plotted in a graph. Firstly, the configuration of the true elemental motions plays a big role. In section 4.4 we discussed geometric degeneracies. In a practical setting as well as in our experiments, we expect that configurations close to degeneracies lead to bad estimates. We have run our experiments with two types of configurations. On one hand we have generated random bivector parameters (i.e. screw axis direction, rotation angle, screw axis offset and pitch). We generated the parameters as follows. Rotation angle and pitch are
drawn from a uniform distribution on the interval between 0 and 1. The screw axis offset is a 3D vector whose three scalar components are drawn from a uniform distribution on the interval between 0 and 1. The screw axis direction is created in the same way and subsequently normalized to have unit length. The exponentials of the resulting bivectors $B_j$ made up our true elemental motions. We believe that this procedure creates geometric configurations that are random enough to include a fair proportion of near degeneracies, even though we are not able to give the distribution of created rigid body motions in closed form. On the other hand we have generated a configuration where the screw axes were parallel to the edges of a unit square (respectively, unit cube) with random rotation angle and pitch uniformly distributed on the interval between 0 and 1. This configuration we deemed particularly stable by intuition.

Secondly, the configuration of the constellation used to estimate the instantaneous motions has consequences for the quality of those estimates [Dor05]. Geometric degeneracies can arise here as well. For example, if all points happen to be collinear, no unique 3D rigid body motion can be determined.

Thirdly, the resilience of the classical linear algebra techniques against noise has to be taken into account. This pertains to both the Procrustean estimate of the instantaneous motions as well as to the decomposition into elemental motions. Both use least squares techniques based on the singular value decomposition (SVD), which are vulnerable to outliers and unstable under minimal data. For the Procrustes method in 3D at least 3 point correspondences are needed to yield a unique rigid body motion. But if only 3 points are used, even small noise (relative to the separation of the points from each other) can greatly influence the result. The same goes for the decomposition. To recover three elemental motions at least 4 instantaneous motions have to be observed. But if only 4 instantaneous motions are used, then even small errors in the estimates $\hat{1}R_i$ can have a big influence on the resulting estimates for the elemental motions.

Moreover, in our particular application it is important to note that under non-exact (respectively non-minimal) data, in the case of $N = 3$ links, as an intermediate step we find a solution for the meta-motion $\hat{2}R_1$ in the null space of the solution matrix (4.83). The versor nature of $\hat{2}R_1$ is encoded in this matrix. Using SVD to approximate the null space will yield a vector representation that is only approximately a versor, i.e. it might not fulfill the property (2.36) that $\hat{2}R_1\hat{2}R_1 \in \mathbb{R} \setminus \{0\}$. When we solve the equation (4.9) with $k = 2$,

$$
\hat{2}R_1 = E_3E_2,
$$

(A.6)

for $E_2$ using the estimated elemental versor $E_3$ the non-versor-ness of $\hat{2}R_1$ “carries over” to $E_2$ and from there to $E_1$. In our preliminary experiments we disregard the fact that the resulting estimates for $E_2$ and $E_1$ may not be versors.

In our experiments the initial constellation is created randomly following a standard normal distribution about a given point. We varied the observation noise level on the points in the constellation by varying the standard deviation $\sigma$ of the zero mean normal distribution of the noise from $\sigma = 0.01$ to $\sigma = 0.02$ to $\sigma = 0.05$. Additionally, we varied the number $P$ of points
A.3 Experimental Results

constituting the constellation from \( P = 4 \) to \( P = 10 \) to \( P = 30 \) and the number \( K \) of observed instantaneous motions from \( K = 4 \) to \( K = 6 \) to \( K = 8 \) to \( K = 12 \) to \( K = 20 \). These variations simulate varying degrees of geometric and numerical stability against noise.

Finally, the assumption of constant speed motions is essential to our approach. In order to evaluate what happens when this assumption is violated we simulated experiments where the \( N \)-th link of the kinematic chain undergoes an accelerated motion \( E_N \). For this purpose we introduce an acceleration parameter \( a \) which is used for a quadratic parametrization of the bivector \( B_N(t) \) by \( t \). Specifically, we let the \( N \)-th elemental motion

\[
E_N(t) = \exp(B_N(t)) = \exp(tB_N + \frac{1}{2}at^2B_N).
\]

(A.7)

We vary the acceleration parameter from \( a = 0 \) for constant speed motion to \( a = 0.01 \) to \( a = 0.05 \) to \( a = 0.1 \) to \( a = 0.5 \). For the accelerated motion experiments we only used configurations where the screw axes were aligned with the unit square (respectively, unit cube) and we do not degrade the points of the constellation by noise.

### A.3. Experimental Results

For each of the experimental setups we have performed 200 runs. We obtained estimates \( \widehat{1R_i} \) for the instantaneous motions at instants \( i = 1, \ldots, K \) as described above. The results of the decomposition method described in chapter 4 are estimates \( \widehat{E_j}, j = 1, \ldots, N \), for the elemental motions of which the true motion is composed. From these estimates we can recover an estimate for the true motion at discrete time instants by

\[
\widehat{0R(i)} = \widehat{E_N^i \ldots E_1^1}, \quad i = 1, \ldots, K.
\]

(A.8)

The effect of the estimated true motion on the original constellation is denoted by

\[
\widehat{X_i} = \widehat{0R(i)}X_0 \left(\widehat{0R(i)}\right)^{-1}.
\]

(A.9)

When comparing the estimates of versors with their true counterparts directly, we report chord lengths, i.e. the Euclidean length of the differences between their vector representations. Restricted to the rotational part, this is equivalent to the well-known quaternion metric (see, for instance, [HTD10]). This value disregards the proper intrinsic distance measure, but it is easier to calculate and we believe that it gives a good approximation to manifold arc length for small differences.

From now on, we use the 8-dimensional vector representations of versors only. The dot-product is the classical, Euclidean dot-product between those vector representations. For each run we calculate the mean difference between the actual (i.e. simulated) instantaneous motion and the one estimated by the Procrustes method from noisy points and report the median
Simulation Experiments
on Motion Decomposition

( denounced $\mu_{1/2}$ ) over all 200 runs

$$\mu_{1/2} \left( \sum_{i=1}^{K} \frac{1}{K} \sqrt{\left( \hat{R}_{i} - \hat{R}_{i} \right) \cdot \left( \hat{R}_{i} - \hat{R}_{i} \right)} \right). \quad (A.10)$$

In the same way we report the median of the mean difference between the actual (i.e. simulated) true motion and the true motion reconstructed by formula (A.8),

$$\mu_{1/2} \left( \sum_{i=1}^{K} \frac{1}{K} \sqrt{\left( \hat{R}(i) - \hat{R}(i) \right) \cdot \left( \hat{R}(i) - \hat{R}(i) \right)} \right). \quad (A.11)$$

Note that the value (A.11) is subject to an error build-up that derives from the error we make in obtaining the elemental versors $\hat{E}_{j}$ (compare equation (A.8)). We report the median of the sum of differences between the actual (i.e. simulated) and the estimated elemental versors

$$\mu_{1/2} \left( \sum_{j=1}^{N} \sqrt{\left( E_{j} - \hat{E}_{j} \right) \cdot \left( E_{j} - \hat{E}_{j} \right)} \right). \quad (A.12)$$

The problem that, in the case of $N = 3$ links, $\hat{E}_{2}$ and $\hat{E}_{1}$ may not be versors (see section A.2) has very little influence on the absolute error measure (A.12), especially since its impact on the data is not yet very well understood.

Finally, we report the median of the mean error that we make when applying the reconstructed true motion to the original constellation. In our experiments the constellation consists of $P$ labeled 3D points. Let the 3D vector representation of each point of the constellation at discrete time instant $i$ be given by $[X_{i}]_{p}, \, p = 1, \ldots, P$. Then we calculate

$$\mu_{1/2} \left( \sum_{i=1}^{K} \sum_{p=1}^{P} \frac{1}{K} \frac{1}{P} \sqrt{\left( [X_{i}]_{p} - [\hat{X}_{i}]_{p} \right) \cdot \left( [X_{i}]_{p} - [\hat{X}_{i}]_{p} \right)} \right). \quad (A.13)$$

Note that, because this calculation uses the reconstructed true motion (see equations (A.9) and (A.8)), it is subject to the error build-up derived from the error we make in obtaining the elemental versors $\hat{E}_{j}$. Here, the fact that, in the case of $N = 3$ links, $\hat{E}_{2}$ and $\hat{E}_{1}$ may not be versors does play a role. If these elemental motion estimates are not versors, the points from the original constellation will not be transformed into points, but rather into (small) spheres. The versor product applied with non-versors may even introduce higher-grade terms in the result. By disregarding this fact and considering only the location of those spheres (and ignoring possible higher-grade terms) we distort the reported quantity. This accounts in part for the difference between the mean point distances in the cases with $N = 2$ links and $N = 3$ links shown in the following.
A.3 Experimental Results

A.3.1. Random Configuration, 3 Links

First, we report the experimental results for the configuration of three randomly generated screw motions.

![Graph](image1)

![Graph](image2)

![Graph](image3)

Figure A.1: Median of the mean difference between the simulated instantaneous motions and the Procrustes estimates for the instantaneous motions for point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$. The original configuration used three randomly generated screw motions.

In Figure A.1 and Figure A.2 we can see that even though the absolute error made by the Procrustes method in estimating the instantaneous motions may be small, the absolute error of the elemental motion estimates can be quite large. We see that the mean error made by the Procrustes method is increasing slightly with more observed instantaneous motions while the error of the elemental motion estimates drastically decreases with more observed instantaneous motions. Moreover, the stability of both the Procrustes method as well as the rigid body motion decomposition increases with more points being used in the observed constellation.

Figure A.3 shows that the mean absolute error made in reconstructing the true motion from elemental motion estimates is quite small, especially compared to the absolute error of the individual elemental motion estimates. Partly due to the considerations in section A.1 it is difficult to judge how detrimental these absolute errors really are. The expected error build-up that results from using powers of already slightly erroneous elemental motion estimates can be
Figure A.2: Median of sum of chord errors of the vector representations of the elemental motions. Three randomly generated elemental motions were used. The plots show experiments with point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$, respectively.

observed as an increase of the error with increasing observation time in Figure A.3. In Figure A.4 we see the results of using the reconstructed true motion for transforming the original constellation. Recall that the original constellation was generated using a standard normal distribution (i.e. the separation of the points should be in the order of 3 units). With this in mind a mean point error of 0.5 seems reasonable, while a mean point error of 10 or even 100 (as observed in case of large noise and few points per constellation) can be considered catastrophic. The conclusion must be that this setup as described in section A.2 is only suitable for low noise levels of up to one percent on point observations.

In this and the following setups we note that the mean distance between true points and points transported by the reconstructed true motion can be rather small, even when the sum of chord errors between the true and estimated elemental motions are not (for example, see Figures A.2 and A.4).
Figure A.3.: Median of mean chord errors of the vector representations of the reconstructed true motion that is the product of powers of three elemental motion estimates. Three randomly generated elemental motions were used. The plots show experiments with point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$, respectively.
Simulation Experiments
on Motion Decomposition

Figure A.4: Median of mean distance between a true point at time instant $i$ and original point moved by reconstructed true motion $\hat{R}(i)$. Three randomly generated elemental motions were used. The plots show experiments with point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$, respectively.
A.3 Experimental Results

A.3.2. Cube Configuration, 3 Links

Here, we show the experimental results for a configuration where directions of the screw axes of three elemental screw motions are aligned with the axes of the Cartesian coordinate system.

![Graphical representation of experimental results]

Figure A.5: Median of the mean difference between the simulated instantaneous motions and the Procrustes estimates for the instantaneous motions for point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$. The original configuration used three screw motions whose screw axes are aligned with axes of the Cartesian coordinate system.

This configuration precludes geometric degeneracies as briefly considered in section 4.4 and in that situation the results of our decomposition method improve dramatically. Figure A.5 shows that the absolute error of the Procrustean estimates for the instantaneous motions is largely unaffected by the configuration, a result that is not surprising. However, in Figure A.6 we see that the absolute error made by the elemental motion estimates is overall lower and decreases much faster with a growing number of instantaneous motion observations (compare Figure A.2).

Comparing Figure A.7 with Figure A.3 shows the stabilizing influence of the cube configuration on the reconstructed true motion. The error build-up is less dramatic, and we conjecture that the reason for that is the independence of the three elemental motions. It leads to a smaller interaction between individual elemental motions; no component of one elemental mo-
Simulation Experiments on Motion Decomposition

Figure A.6.: Median of sum of chord errors of the vector representations of the elemental motions. Three elemental motions were used whose screw axes were aligned with the axes of the Cartesian coordinate system. The plots show experiments with point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$, respectively.

...tion is subsumed in another elemental motion. Also, the mean distance between true points and original points transformed by the reconstructed true motion behaves more nicely. In Figure A.8 we observe a slower increase of that distance with increasing number of instantaneous motion observations (note the different scales of the graphs). A catastrophic “explosion” of the mean point error can mainly be attributed to large observation noise or a low number of points per constellation.

In this setup the method now performs reasonably for noise levels of one and two percent on the point observations, given that enough (i.e. ten or more) points are used.
Figure A.7: Median of mean chord errors of the vector representations of the reconstructed true motion that is the product of powers of three elemental motion estimates. Three elemental motions were used whose screw axes were aligned with the axes of the Cartesian coordinate system. The plots show experiments with point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$, respectively.
Figure A.8.: Median of mean distance between a true point at time instant $i$ and original point moved by reconstructed true motion $\overrightarrow{R}(i)$. Three elemental motions were used whose screw axes were aligned with the axes of the Cartesian coordinate system. The plots show experiments with point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$, respectively.
A.3.3. Random Configuration, 2 Links

Here, we show the experimental results for a configuration of two randomly generated 3D screw motions. One big difference to the case of $N = 3$ links is that the resulting elemental motion estimates are guaranteed to be versors. In the case of $N = 3$ links the versor-ness of the result is an explicit constraint (see (4.81)) which is subject to the approximation using the SVD and therefore can be violated. In the case of $N = 2$ links the results are versors in any case. Only their behavior is subject to the SVD approximation, not their versor nature.

Figure A.9.: Median of the mean difference between the simulated instantaneous motions and the Procrustes estimates for the instantaneous motions for point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$. The original configuration used two randomly generated screw motions.

We observe in Figures A.9 and A.10 that the absolute noise in the Procrustean estimates of the instantaneous motions does not depend on the configuration of the elemental motions and that the absolute error made in the elemental motion estimates is reasonably small and decreases quickly with more instantaneous motion observations. However, even a slight increase of point observation noise leads to a large error here which can only be compensated for by increasing the number of point observations in a constellation or by increasing the number of instantaneous motions used.

The behavior exhibited by the mean chord error between the actual and reconstructed true
Simulation Experiments
on Motion Decomposition

Figure A.10.: Median of sum of chord errors of the vector representations of the elemental motions. Two randomly generated elemental motions were used. The plots show experiments with point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$, respectively.

motion for increasing number of time (depicted in Figure A.11) instances is very similar to the case of $N = 3$ links (compare Figure A.3), but the overall magnitude of this error is much smaller. The mean distance between true points and points of the original configuration transformed by the reconstructed true motion (shown in Figure A.12) is smaller than for the case of $N = 3$ links (compare Figure A.4) in absolute terms, but shows the same tendencies to "explode" for unstable constellations with few points or for large point observation noise.

In this setup the method produces reasonable results for noise levels of up to five percent on the point observations, if a large number of points (namely 10 or 30) are used and only a few (up to 12 for small noise, up to 6 for large noise) instantaneous motions are observed.
Figure A.11.: Median of mean chord errors of the vector representations of the reconstructed true motion that is the product of powers of two elemental motion estimates. Two randomly generated elemental motions were used. The plots show experiments with point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$, respectively.
Simulation Experiments
on Motion Decomposition

Figure A.12: Median of mean distance between a true point at time instant $i$ and original point moved by reconstructed true motion $\hat{0}R(i)$. Two randomly generated elemental motions were used. The plots show experiments with point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$, respectively.
A.3.4. Cube Configuration, 2 Links

In this section we show the experimental results for a configuration where the axes of two elemental 3D screw motions are aligned with the $x-$ and $y-$axes of the Cartesian coordinate system.

Figure A.13: Median of the mean difference between the simulated instantaneous motions and the Procrustes estimates for the instantaneous motions for point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$. The original configuration used two screw motions whose screw axes are aligned with axes of the Cartesian coordinate system.

The differences between this case and the random configuration with two links (i.e. between Figure A.14 and Figure A.10) are similar to the differences between the cube and random configurations in the case with $N = 3$ links (i.e. between Figure A.6 and Figure A.2). The observation that the sum of chord errors of the elemental motion estimates is absolutely much smaller than in the cases with $N = 3$ links cannot be fully accounted for by the fact that we add the chords of only two instead of three elemental motions. We believe additional reasons for that observation to be a reduced chance of geometric degeneracies (for random configurations), the guarantee of the versor nature of the elemental motion estimates and the fact that the solution depends more immediately on the elemental motions (i.e. meta-motions $^2R_i$ do not have to be considered).
Simulation Experiments
on Motion Decomposition

Figure A.14.: Median of sum of chord errors of the vector representations of the elemental motions. Two elemental motions were used whose screw axes were aligned with the axes of the Cartesian coordinate system. The plots show experiments with point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$, respectively.

When it comes to the absolute mean chord error of the reconstructed true motion and the mean distance between true points and points transformed by the reconstructed true motion we see that both are generally lower both than in the case of a random configuration of $N = 2$ links as well as a cube configuration of $N = 3$ links. The overall magnitude of this mean point error is well within the order of the observation noise if a sufficient number of points is used for the Procrustean estimate of the instantaneous motions.
A.3 Experimental Results

Figure A.15: Median of mean chord errors of the vector representations of the reconstructed true motion that is the product of powers of two elemental motion estimates. Two elemental motions were used whose screw axes were aligned with the axes of the Cartesian coordinate system. The plots show experiments with point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$, respectively.
Simulation Experiments on Motion Decomposition

Figure A.16: Median of mean distance between a true point at time instant $i$ and original point moved by reconstructed true motion $^0R(i)$. Two elemental motions were used whose screw axes were aligned with the axes of the Cartesian coordinate system. The plots show experiments with point observation noise levels of $\sigma = 0.01$, $\sigma = 0.02$ and $\sigma = 0.05$, respectively.
A.3.5. Cube Configuration, 3 Links, Accelerated Third Link

In this section we report the experimental results for a configuration where the directions of the screw axes of three elemental 3D screw motions are aligned with the axes of the Cartesian coordinate system. The last link undergoes an accelerated screw motion with constant acceleration parameter $a$ as described in section A.2. The point observations are not degraded by noise, which is why we only use 4 points per constellation.

![Figure A.17.](image)

**Figure A.17.** Median of the mean difference between the simulated instantaneous motions and the Procrustes estimates for the instantaneous motions for constant acceleration parameters $a = 0.01$, $a = 0.05$, $a = 0.1$ and $a = 0.5$ on a linear and on a logarithmic scale. The original configuration used three screw motions whose screw axes are aligned with axes of the Cartesian coordinate system.

The error in the instantaneous motions made by the Procrustean estimates is insignificant in all cases. Note the scale of $10^{-14}$ in the plots in Figure A.17. In all other reported error measures (see Figures A.18 through A.20) there is a tendency to grow seemingly exponentially fast with a growing number of observed instantaneous motions. The massive increase of the error with increasing number of observed instantaneous motions is expected since the acceleration parameter ensures that the observed instantaneous motions deviate more and more from the assumed constant speed rigid body motions that our decomposition method requires. Reasonable results (meaning results with error measures that are of a similar order of magnitude as in the previous experiments) can be obtained for low acceleration parameters and a small number of observed instantaneous rigid body motions.
Simulation Experiments on Motion Decomposition

Figure A.18: Median of sum of chord errors of the vector representations of the elemental motions. Three elemental motions were used whose screw axes were aligned with the axes of the Cartesian coordinate system. The plots show experiments with constant acceleration parameter \(a = 0.01, a = 0.05, a = 0.1\) and \(a = 0.5\), respectively, on a linear and on a logarithmic scale.

Figure A.19: Median of mean chord errors of the vector representations of the reconstructed true motion that is the product of powers of three elemental motion estimates. Three elemental motions were used whose screw axes were aligned with the axes of the Cartesian coordinate system. The plots show experiments with constant acceleration parameter \(a = 0.01, a = 0.05, a = 0.1\) and \(a = 0.5\), respectively, on a linear and on a logarithmic scale.
Figure A.20.: Median of mean distance between a true point at time instant $i$ and original point moved by reconstructed true motion $\theta_i R(i)$. Three elemental motions were used whose screw axes were aligned with the axes of the Cartesian coordinate system. The plots show experiments with constant acceleration parameter $a = 0.01$, $a = 0.05$, $a = 0.1$ and $a = 0.5$, respectively, on a linear and on a logarithmic scale.
A.3.6. Cube Configuration, 2 Links, Accelerated Second Link

Here we present the experimental results for a configuration where the directions of the screw axes of two elemental 3D screw motions are aligned with the $x -$ and $y -$axes of the Cartesian coordinate system. The last link undergoes an accelerated screw motion with constant acceleration parameter $a$ as described in section A.2. The point observations are not degraded by noise, which is why we only use 4 points per constellation.

![Graph of mean chord length vs. number of instantaneous motions for different acceleration parameters.](image)

**Figure A.21**: Median of the mean difference between the simulated instantaneous motions and the Procrustes estimates for the instantaneous motions for constant acceleration parameters $a = 0.01$, $a = 0.05$, $a = 0.1$ and $a = 0.5$ on a linear and on a logarithmic scale. The original configuration used two screw motions whose screw axes are aligned with axes of the Cartesian coordinate system.

The error measures for this case exhibit the same seemingly exponential increase as in the case with $N = 3$ links. For small acceleration parameters and few observed instantaneous motions the results are encouraging. For example, Figure A.24 shows that the median of the mean distance between true points and points moved by the reconstructed true motion are sometimes smaller than in all previous experiments. We conclude that an acceleration of $a = 0.01$ still produces acceptable results for up to 8 observed instantaneous motions. Recall, however, that in this section our point observations were not degraded by noise. The instantaneous motions used for the decomposition method were virtually error free. If point observation noise would come into play, the results are expected to degrade further.
A.3 Experimental Results

Figure A.22: Median of sum of chord errors of the vector representations of the elemental motions. Two elemental motions were used whose screw axes were aligned with the axes of the Cartesian coordinate system. The plots show experiments with constant acceleration parameter $a = 0.01$, $a = 0.05$, $a = 0.1$ and $a = 0.5$, respectively, on a linear and on a logarithmic scale.

Figure A.23: Median of mean chord errors of the vector representations of the reconstructed true motion that is the product of powers of two elemental motion estimates. Two elemental motions were used whose screw axes were aligned with the axes of the Cartesian coordinate system. The plots show experiments with constant acceleration parameter $a = 0.01$, $a = 0.05$, $a = 0.1$ and $a = 0.5$, respectively, on a linear and on a logarithmic scale.
Simulation Experiments
on Motion Decomposition

Figure A.24.: Median of mean distance between a true point at time instant $i$ and original point moved by reconstructed true motion $\hat{\mathbf{R}}(i)$. Two elemental motions were used whose screw axes were aligned with the axes of the Cartesian coordinate system. The plots show experiments with constant acceleration parameter $a = 0.01$, $a = 0.05$, $a = 0.1$ and $a = 0.5$, respectively, on a linear and on a logarithmic scale.
In this section we present numerical values for the results of our experiments. For compactness we omit the cases where $N = 10$ and the cases where $\sigma = 2$, because they represent intermediate cases which do not add significantly to the interpretation of our results.
## Quantitative Results

of Bayesian Hypersphere Estimation

<table>
<thead>
<tr>
<th>4 points</th>
<th>20 points</th>
<th>100 points</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b_{NLS} = 12.920$</td>
<td>$b_{NLS} = 93.784$</td>
<td>$b_{NLS} = 12.216$</td>
</tr>
<tr>
<td>$b_{MLE} = 12.825$</td>
<td>$b_{MLE} = 9.092$</td>
<td>$b_{MLE} = 7.568$</td>
</tr>
<tr>
<td>$tr_{NLS} = 6.631$</td>
<td>$tr_{NLS} = 10^6 \times 1.362$</td>
<td>$tr_{NLS} = 10^5 \times 2.053$</td>
</tr>
<tr>
<td>$tr_{MLE} = 3.916$</td>
<td>$tr_{MLE} = 7.155$</td>
<td>$tr_{MLE} = 0.901$</td>
</tr>
<tr>
<td>$det_{NLS} = 1.721$</td>
<td>$det_{NLS} = 10^{18} \times 4.542$</td>
<td>$det_{NLS} = 10^{14} \times 4.043$</td>
</tr>
<tr>
<td>$det_{MLE} = 0.380$</td>
<td>$det_{MLE} = 0.065$</td>
<td>$det_{MLE} = 10^{-5} \times 1.546$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4 points</th>
<th>20 points</th>
<th>100 points</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b_{NLS} = 12.960$</td>
<td>$b_{NLS} = 11.643$</td>
<td>$b_{NLS} = 66.729$</td>
</tr>
<tr>
<td>$b_{MLE} = 13.315$</td>
<td>$b_{MLE} = 13.229$</td>
<td>$b_{MLE} = 13.258$</td>
</tr>
<tr>
<td>$tr_{NLS} = 9.005$</td>
<td>$tr_{NLS} = 10^3 \times 4.357$</td>
<td>$tr_{NLS} = 10^6 \times 1.738$</td>
</tr>
<tr>
<td>$tr_{MLE} = 0.792$</td>
<td>$tr_{MLE} = 0.251$</td>
<td>$tr_{MLE} = 0.041$</td>
</tr>
<tr>
<td>$det_{NLS} = 3.015$</td>
<td>$det_{NLS} = 10^9 \times 1.637$</td>
<td>$det_{NLS} = 10^{19} \times 2.233$</td>
</tr>
<tr>
<td>$det_{MLE} = 0.001$</td>
<td>$det_{MLE} = 10^{-5} \times 1.054$</td>
<td>$det_{MLE} = 10^{-9} \times 5.346$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4 points</th>
<th>20 points</th>
<th>100 points</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b_{NLS} = 12.549$</td>
<td>$b_{NLS} = 12.700$</td>
<td>$b_{NLS} = 12.833$</td>
</tr>
<tr>
<td>$b_{MLE} = 12.936$</td>
<td>$b_{MLE} = 12.868$</td>
<td>$b_{MLE} = 12.857$</td>
</tr>
<tr>
<td>$tr_{NLS} = 14.462$</td>
<td>$tr_{NLS} = 9.331$</td>
<td>$tr_{NLS} = 0.478$</td>
</tr>
<tr>
<td>$tr_{MLE} = 1.274$</td>
<td>$tr_{MLE} = 0.266$</td>
<td>$tr_{MLE} = 0.053$</td>
</tr>
<tr>
<td>$det_{NLS} = 24.504$</td>
<td>$det_{NLS} = 4.120$</td>
<td>$det_{NLS} = 10^{-5} \times 9.468$</td>
</tr>
<tr>
<td>$det_{MLE} = 0.008$</td>
<td>$det_{MLE} = 10^{-5} \times 1.3317$</td>
<td>$det_{MLE} = 10^{-9} \times 2.083$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4 points</th>
<th>20 points</th>
<th>100 points</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b_{NLS} = 9.794$</td>
<td>$b_{NLS} = 13.665$</td>
<td>$b_{NLS} = 10.124$</td>
</tr>
<tr>
<td>$b_{MLE} = 10.257$</td>
<td>$b_{MLE} = 10.110$</td>
<td>$b_{MLE} = 10.100$</td>
</tr>
<tr>
<td>$tr_{NLS} = 116.091$</td>
<td>$tr_{NLS} = 10^6 \times 1.113$</td>
<td>$tr_{NLS} = 7.127$</td>
</tr>
<tr>
<td>$tr_{MLE} = 21.821$</td>
<td>$tr_{MLE} = 4.457$</td>
<td>$tr_{MLE} = 0.888$</td>
</tr>
<tr>
<td>$det_{NLS} = 10^3 \times 1.161$</td>
<td>$det_{NLS} = 10^9 \times 3.841$</td>
<td>$det_{NLS} = 3.619$</td>
</tr>
<tr>
<td>$det_{MLE} = 557.803$</td>
<td>$det_{MLE} = 1.008$</td>
<td>$det_{MLE} = 10^{-3} \times 1.612$</td>
</tr>
</tbody>
</table>

Table B.1.: sample points with spread $q = 0.1$
<table>
<thead>
<tr>
<th></th>
<th>4 points</th>
<th>20 points</th>
<th>100 points</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma = 0.1$</td>
<td>b$_{NLS}$ = 11.326</td>
<td>b$_{NLS}$ = 4.651</td>
<td>b$_{NLS}$ = 0.030</td>
</tr>
<tr>
<td></td>
<td>b$_{MLE}$ = 9.707</td>
<td>b$_{MLE}$ = 2.735</td>
<td>b$_{MLE}$ = 1.696</td>
</tr>
<tr>
<td></td>
<td>tr$_{NLS}$ = 12.152</td>
<td>tr$_{NLS}$ = $10^5 \times 1.004$</td>
<td>tr$_{NLS}$ = 0.504</td>
</tr>
<tr>
<td></td>
<td>tr$_{MLE}$ = 18.615</td>
<td>tr$_{MLE}$ = 3.279</td>
<td>tr$_{MLE}$ = $10^{-9} \times 3.465$</td>
</tr>
<tr>
<td></td>
<td>det$_{NLS}$ = 16.755</td>
<td>det$_{NLS}$ = $10^9 \times 1.410$</td>
<td>det$_{NLS}$ = 0.381</td>
</tr>
<tr>
<td></td>
<td>det$_{MLE}$ = 37.989</td>
<td>det$_{MLE}$ = $10^{-3} \times 1.061$</td>
<td>det$_{MLE}$ = $10^{-9} \times 1.086$</td>
</tr>
<tr>
<td>$\sigma = 0.5$</td>
<td>b$_{NLS}$ = 11.895</td>
<td>b$_{NLS}$ = 126.044</td>
<td>b$_{NLS}$ = 266.951</td>
</tr>
<tr>
<td></td>
<td>b$_{MLE}$ = 12.400</td>
<td>b$_{MLE}$ = 10.475</td>
<td>b$_{MLE}$ = 9.367</td>
</tr>
<tr>
<td></td>
<td>tr$_{NLS}$ = 9.971</td>
<td>tr$_{NLS}$ = $10^7 \times 1.414$</td>
<td>tr$_{NLS}$ = $10^7 \times 1.829$</td>
</tr>
<tr>
<td></td>
<td>tr$_{MLE}$ = 4.316</td>
<td>tr$_{MLE}$ = 5.042</td>
<td>tr$_{MLE}$ = 1.107</td>
</tr>
<tr>
<td></td>
<td>det$_{NLS}$ = 30.331</td>
<td>det$_{NLS}$ = $10^{22} \times 2.759$</td>
<td>det$_{NLS}$ = $10^{21} \times 5.586$</td>
</tr>
<tr>
<td></td>
<td>det$_{MLE}$ = 1.011</td>
<td>det$_{MLE}$ = 0.261</td>
<td>det$_{MLE}$ = $10^{-5} \times 3.544$</td>
</tr>
<tr>
<td>$\sigma = 1$</td>
<td>b$_{NLS}$ = 11.844</td>
<td>b$_{NLS}$ = 16.924</td>
<td>b$_{NLS}$ = 100.013</td>
</tr>
<tr>
<td></td>
<td>b$_{MLE}$ = 12.441</td>
<td>b$_{MLE}$ = 12.237</td>
<td>b$_{MLE}$ = 12.286</td>
</tr>
<tr>
<td></td>
<td>tr$_{NLS}$ = 12.652</td>
<td>tr$_{NLS}$ = $10^5 \times 4.597$</td>
<td>tr$_{NLS}$ = $10^6 \times 3.674$</td>
</tr>
<tr>
<td></td>
<td>tr$_{MLE}$ = 3.058</td>
<td>tr$_{MLE}$ = 0.941</td>
<td>tr$_{MLE}$ = 0.168</td>
</tr>
<tr>
<td></td>
<td>det$_{NLS}$ = 76.936</td>
<td>det$_{NLS}$ = $10^{16} \times 1.923$</td>
<td>det$_{NLS}$ = $10^{21} \times 2.951$</td>
</tr>
<tr>
<td></td>
<td>det$_{MLE}$ = 0.242</td>
<td>det$_{MLE}$ = $10^{-3} \times 1.689$</td>
<td>det$_{MLE}$ = $10^{-6} \times 1.425$</td>
</tr>
<tr>
<td>$\sigma = 5$</td>
<td>b$_{NLS}$ = 9.548</td>
<td>b$_{NLS}$ = 9.801</td>
<td>b$_{NLS}$ = 9.996</td>
</tr>
<tr>
<td></td>
<td>b$_{MLE}$ = 9.987</td>
<td>b$_{MLE}$ = 9.951</td>
<td>b$_{MLE}$ = 9.938</td>
</tr>
<tr>
<td></td>
<td>tr$_{NLS}$ = 115.693</td>
<td>tr$_{NLS}$ = 54.279</td>
<td>tr$_{NLS}$ = 7.460</td>
</tr>
<tr>
<td></td>
<td>tr$_{MLE}$ = 22.754</td>
<td>tr$_{MLE}$ = 4.730</td>
<td>tr$_{MLE}$ = 0.949</td>
</tr>
<tr>
<td></td>
<td>det$_{NLS}$ = $10^5 \times 6.536$</td>
<td>det$_{NLS}$ = $10^4 \times 2.018$</td>
<td>det$_{NLS}$ = 4.253</td>
</tr>
<tr>
<td></td>
<td>det$_{MLE}$ = 686.906</td>
<td>det$_{MLE}$ = 1.326</td>
<td>det$_{MLE}$ = $10^{-3} \times 2.075$</td>
</tr>
</tbody>
</table>

Table B.2.: sample points with spread $q = 0.2$
## Quantitative Results of Bayesian Hypersphere Estimation

<table>
<thead>
<tr>
<th></th>
<th>4 points</th>
<th>20 points</th>
<th>100 points</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma = 0.1$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$b_{NLS}$ = 6.858</td>
<td>$b_{NLS}$ = 0.018</td>
<td>$b_{NLS}$ = $10^{-3} \times 2.338$</td>
<td></td>
</tr>
<tr>
<td>$b_{MLE}$ = 3.452</td>
<td>$b_{MLE}$ = 0.111</td>
<td>$b_{MLE}$ = 0.056</td>
<td></td>
</tr>
<tr>
<td>$\text{tr}_{NLS}$ = 25.185</td>
<td>$\text{tr}_{NLS}$ = 0.473</td>
<td>$\text{tr}_{NLS}$ = 0.017</td>
<td></td>
</tr>
<tr>
<td>$\text{tr}_{MLE}$ = 20.948</td>
<td>$\text{tr}_{MLE}$ = 0.120</td>
<td>$\text{tr}_{MLE}$ = 0.012</td>
<td></td>
</tr>
<tr>
<td>$\text{det}_{NLS}$ = 73.759</td>
<td>$\text{det}_{NLS}$ = $10^{-8} \times 4.165$</td>
<td>$\text{det}_{NLS}$ = $10^{-12} \times 3.520$</td>
<td></td>
</tr>
<tr>
<td>$\text{det}_{MLE}$ = 31.223</td>
<td>$\text{det}_{MLE}$ = $10^{-9} \times 2.460$</td>
<td>$\text{det}_{MLE}$ = $10^{-13} \times 6.374$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>0.5</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$b_{NLS}$ = 7.538</td>
<td>$b_{NLS}$ = 0.333</td>
<td>$b_{NLS}$ = 0.047</td>
<td></td>
</tr>
<tr>
<td>$b_{MLE}$ = 7.810</td>
<td>$b_{MLE}$ = 1.930</td>
<td>$b_{MLE}$ = 1.201</td>
<td></td>
</tr>
<tr>
<td>$\text{tr}_{NLS}$ = 30.384</td>
<td>$\text{tr}_{NLS}$ = 336.660</td>
<td>$\text{tr}_{NLS}$ = 0.436</td>
<td></td>
</tr>
<tr>
<td>$\text{tr}_{MLE}$ = 24.096</td>
<td>$\text{tr}_{MLE}$ = 2.645</td>
<td>$\text{tr}_{MLE}$ = 0.328</td>
<td></td>
</tr>
<tr>
<td>$\text{det}_{NLS}$ = 637.044</td>
<td>$\text{det}_{NLS}$ = 0.961</td>
<td>$\text{det}_{NLS}$ = $10^{-6} \times 1.436$</td>
<td></td>
</tr>
<tr>
<td>$\text{det}_{MLE}$ = 301.184</td>
<td>$\text{det}_{MLE}$ = $10^{-3} \times 2.630$</td>
<td>$\text{det}_{MLE}$ = $10^{-7} \times 4.067$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$b_{NLS}$ = 7.954</td>
<td>$b_{NLS}$ = 1.219</td>
<td>$b_{NLS}$ = 0.113</td>
<td></td>
</tr>
<tr>
<td>$b_{MLE}$ = 9.145</td>
<td>$b_{MLE}$ = 4.796</td>
<td>$b_{MLE}$ = 3.619</td>
<td></td>
</tr>
<tr>
<td>$\text{tr}_{NLS}$ = 56.732</td>
<td>$\text{tr}_{NLS}$ = $10^3 \times 2.251$</td>
<td>$\text{tr}_{NLS}$ = 1.931</td>
<td></td>
</tr>
<tr>
<td>$\text{tr}_{MLE}$ = 19.756</td>
<td>$\text{tr}_{MLE}$ = 6.850</td>
<td>$\text{tr}_{MLE}$ = 1.155</td>
<td></td>
</tr>
<tr>
<td>$\text{det}_{NLS}$ = $10^4 \times 1.052$</td>
<td>$\text{det}_{NLS}$ = $10^7 \times 2.316$</td>
<td>$\text{det}_{NLS}$ = $10^{-4} \times 5.018$</td>
<td></td>
</tr>
<tr>
<td>$\text{det}_{MLE}$ = 224.848</td>
<td>$\text{det}_{MLE}$ = 0.247</td>
<td>$\text{det}_{MLE}$ = $10^{-5} \times 9.641$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$b_{NLS}$ = 7.680</td>
<td>$b_{NLS}$ = 8.330</td>
<td>$b_{NLS}$ = 8.237</td>
<td></td>
</tr>
<tr>
<td>$b_{MLE}$ = 8.861</td>
<td>$b_{MLE}$ = 8.750</td>
<td>$b_{MLE}$ = 8.730</td>
<td></td>
</tr>
<tr>
<td>$\text{tr}_{NLS}$ = $10^3 \times 1.492$</td>
<td>$\text{tr}_{NLS}$ = 51.285</td>
<td>$\text{tr}_{NLS}$ = 10.505</td>
<td></td>
</tr>
<tr>
<td>$\text{tr}_{MLE}$ = 30.318</td>
<td>$\text{tr}_{MLE}$ = 6.355</td>
<td>$\text{tr}_{MLE}$ = 1.262</td>
<td></td>
</tr>
<tr>
<td>$\text{det}_{NLS}$ = $10^8 \times 6.196$</td>
<td>$\text{det}_{NLS}$ = $10^4 \times 2.222$</td>
<td>$\text{det}_{NLS}$ = 16.406</td>
<td></td>
</tr>
<tr>
<td>$\text{det}_{MLE}$ = $10^5 \times 2.372$</td>
<td>$\text{det}_{MLE}$ = 4.531</td>
<td>$\text{det}_{MLE}$ = $10^{-3} \times 6.638$</td>
<td></td>
</tr>
</tbody>
</table>

Table B.3.: sample points with spread $q = 0.5$
<table>
<thead>
<tr>
<th></th>
<th>4 points</th>
<th>20 points</th>
<th>100 points</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma = 0.1$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$b_{NLS} = 2.885$</td>
<td>$b_{NLS} = 10^{-3} \times 2.712$</td>
<td>$b_{NLS} = 10^{-3} \times 1.588$</td>
</tr>
<tr>
<td></td>
<td>$b_{MLE} = 0.934$</td>
<td>$b_{MLE} = 10^{-3} \times 4.529$</td>
<td>$b_{MLE} = 10^{-3} \times 3.057$</td>
</tr>
<tr>
<td></td>
<td>$\text{tr}_{NLS} = 18.307$</td>
<td>$\text{tr}_{NLS} = 0.019$</td>
<td>$\text{tr}_{NLS} = 10^{-3} \times 2.807$</td>
</tr>
<tr>
<td></td>
<td>$\text{tr}_{MLE} = 7.897$</td>
<td>$\text{tr}_{MLE} = 0.013$</td>
<td>$\text{tr}_{MLE} = 10^{-3} \times 1.804$</td>
</tr>
<tr>
<td></td>
<td>$\text{det}_{NLS} = 30.589$</td>
<td>$\text{det}_{NLS} = 10^{-10} \times 1.195$</td>
<td>$\text{det}_{NLS} = 10^{-14} \times 7.618$</td>
</tr>
<tr>
<td></td>
<td>$\text{det}_{MLE} = 1.863$</td>
<td>$\text{det}_{MLE} = 10^{-11} \times 2.768$</td>
<td>$\text{det}_{MLE} = 10^{-14} \times 1.288$</td>
</tr>
<tr>
<td>$\sigma = 0.5$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$b_{NLS} = 3.041$</td>
<td>$b_{NLS} = 0.039$</td>
<td>$b_{NLS} = 0.027$</td>
</tr>
<tr>
<td></td>
<td>$b_{MLE} = 2.858$</td>
<td>$b_{MLE} = 0.153$</td>
<td>$b_{MLE} = 0.088$</td>
</tr>
<tr>
<td></td>
<td>$\text{tr}_{NLS} = 23.529$</td>
<td>$\text{tr}_{NLS} = 0.490$</td>
<td>$\text{tr}_{NLS} = 0.072$</td>
</tr>
<tr>
<td></td>
<td>$\text{tr}_{MLE} = 20.969$</td>
<td>$\text{tr}_{MLE} = 0.340$</td>
<td>$\text{tr}_{MLE} = 0.047$</td>
</tr>
<tr>
<td></td>
<td>$\text{det}_{NLS} = 148.680$</td>
<td>$\text{det}_{NLS} = 10^{-5} \times 4.922$</td>
<td>$\text{det}_{NLS} = 10^{-8} \times 3.169$</td>
</tr>
<tr>
<td></td>
<td>$\text{det}_{MLE} = 126.865$</td>
<td>$\text{det}_{MLE} = 10^{-5} \times 1.075$</td>
<td>$\text{det}_{MLE} = 10^{-9} \times 5.547$</td>
</tr>
<tr>
<td>$\sigma = 1$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$b_{NLS} = 3.190$</td>
<td>$b_{NLS} = 0.137$</td>
<td>$b_{NLS} = 0.106$</td>
</tr>
<tr>
<td></td>
<td>$b_{MLE} = 3.933$</td>
<td>$b_{MLE} = 0.576$</td>
<td>$b_{MLE} = 0.354$</td>
</tr>
<tr>
<td></td>
<td>$\text{tr}_{NLS} = 36.463$</td>
<td>$\text{tr}_{NLS} = 2.130$</td>
<td>$\text{tr}_{NLS} = 0.300$</td>
</tr>
<tr>
<td></td>
<td>$\text{tr}_{MLE} = 27.024$</td>
<td>$\text{tr}_{MLE} = 1.438$</td>
<td>$\text{tr}_{MLE} = 0.197$</td>
</tr>
<tr>
<td></td>
<td>$\text{det}_{NLS} = 10^3 \times 2.290$</td>
<td>$\text{det}_{NLS} = 0.017$</td>
<td>$\text{det}_{NLS} = 10^{-6} \times 9.255$</td>
</tr>
<tr>
<td></td>
<td>$\text{det}_{MLE} = 529.410$</td>
<td>$\text{det}_{MLE} = 10^{-3} \times 3.313$</td>
<td>$\text{det}_{MLE} = 10^{-6} \times 1.620$</td>
</tr>
<tr>
<td>$\sigma = 5$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$b_{NLS} = 5.862$</td>
<td>$b_{NLS} = 4.276$</td>
<td>$b_{NLS} = 3.028$</td>
</tr>
<tr>
<td></td>
<td>$b_{MLE} = 5.650$</td>
<td>$b_{MLE} = 5.290$</td>
<td>$b_{MLE} = 5.183$</td>
</tr>
<tr>
<td></td>
<td>$\text{tr}_{NLS} = 180.102$</td>
<td>$\text{tr}_{NLS} = 73.665$</td>
<td>$\text{tr}_{NLS} = 13.280$</td>
</tr>
<tr>
<td></td>
<td>$\text{tr}_{MLE} = 46.224$</td>
<td>$\text{tr}_{MLE} = 10.465$</td>
<td>$\text{tr}_{MLE} = 2.108$</td>
</tr>
<tr>
<td></td>
<td>$\text{det}_{NLS} = 10^6 \times 3.896$</td>
<td>$\text{det}_{NLS} = 10^4 \times 9.112$</td>
<td>$\text{det}_{NLS} = 26.098$</td>
</tr>
<tr>
<td></td>
<td>$\text{det}_{MLE} = 10^4 \times 1.270$</td>
<td>$\text{det}_{MLE} = 27.684$</td>
<td>$\text{det}_{MLE} = 0.042$</td>
</tr>
</tbody>
</table>

Table B.4.: sample points with spread $q = 1.0$
Quantitative Results
of Bayesian Hypersphere Estimation

<table>
<thead>
<tr>
<th>$\sigma$</th>
<th>4 points</th>
<th>20 points</th>
<th>100 points</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma = 0.1$</td>
<td>$b_{NLS} = 0.447$</td>
<td>$b_{NLS} = 10^{-3} \times 1.961$</td>
<td>$b_{NLS} = 10^{-3} \times 1.199$</td>
</tr>
<tr>
<td></td>
<td>$b_{MLE} = 0.060$</td>
<td>$b_{MLE} = 10^{-3} \times 1.432$</td>
<td>$b_{MLE} = 10^{-3} \times 1.103$</td>
</tr>
<tr>
<td></td>
<td>$tr_{NLS} = 6.396$</td>
<td>$tr_{NLS} = 10^{-3} \times 9.829$</td>
<td>$tr_{NLS} = 10^{-3} \times 1.775$</td>
</tr>
<tr>
<td></td>
<td>$tr_{MLE} = 2.281$</td>
<td>$tr_{MLE} = 10^{-3} \times 6.769$</td>
<td>$tr_{MLE} = 10^{-3} \times 1.155$</td>
</tr>
<tr>
<td></td>
<td>$det_{NLS} = 5.487$</td>
<td>$det_{NLS} = 10^{-11} \times 2.015$</td>
<td>$det_{NLS} = 10^{-14} \times 2.135$</td>
</tr>
<tr>
<td></td>
<td>$det_{MLE} = 0.092$</td>
<td>$det_{MLE} = 10^{-12} \times 4.544$</td>
<td>$det_{MLE} = 10^{-15} \times 3.730$</td>
</tr>
<tr>
<td>$\sigma = 0.5$</td>
<td>$b_{NLS} = 0.358$</td>
<td>$b_{NLS} = 0.029$</td>
<td>$b_{NLS} = 0.025$</td>
</tr>
<tr>
<td></td>
<td>$b_{MLE} = 0.234$</td>
<td>$b_{MLE} = 0.024$</td>
<td>$b_{MLE} = 0.024$</td>
</tr>
<tr>
<td></td>
<td>$tr_{NLS} = 9.845$</td>
<td>$tr_{NLS} = 0.248$</td>
<td>$tr_{NLS} = 0.044$</td>
</tr>
<tr>
<td></td>
<td>$tr_{MLE} = 8.914$</td>
<td>$tr_{MLE} = 0.167$</td>
<td>$tr_{MLE} = 0.029$</td>
</tr>
<tr>
<td></td>
<td>$det_{NLS} = 30.066$</td>
<td>$det_{NLS} = 10^{-6} \times 7.827$</td>
<td>$det_{NLS} = 10^{-9} \times 8.128$</td>
</tr>
<tr>
<td></td>
<td>$det_{MLE} = 19.447$</td>
<td>$det_{MLE} = 10^{-6} \times 1.555$</td>
<td>$det_{MLE} = 10^{-9} \times 1.443$</td>
</tr>
<tr>
<td>$\sigma = 1$</td>
<td>$b_{NLS} = 0.133$</td>
<td>$b_{NLS} = 0.130$</td>
<td>$b_{NLS} = 0.104$</td>
</tr>
<tr>
<td></td>
<td>$b_{MLE} = 0.280$</td>
<td>$b_{MLE} = 0.106$</td>
<td>$b_{MLE} = 0.100$</td>
</tr>
<tr>
<td></td>
<td>$tr_{NLS} = 17.214$</td>
<td>$tr_{NLS} = 1.004$</td>
<td>$tr_{NLS} = 0.181$</td>
</tr>
<tr>
<td></td>
<td>$tr_{MLE} = 15.507$</td>
<td>$tr_{MLE} = 0.659$</td>
<td>$tr_{MLE} = 0.112$</td>
</tr>
<tr>
<td></td>
<td>$det_{NLS} = 269.339$</td>
<td>$det_{NLS} = 10^{-3} \times 2.091$</td>
<td>$det_{NLS} = 10^{-6} \times 2.272$</td>
</tr>
<tr>
<td></td>
<td>$det_{MLE} = 170.300$</td>
<td>$det_{MLE} = 10^{-4} \times 0.022$</td>
<td>$det_{MLE} = 10^{-7} \times 3.552$</td>
</tr>
<tr>
<td>$\sigma = 5$</td>
<td>$b_{NLS} = 4.614$</td>
<td>$b_{NLS} = 3.247$</td>
<td>$b_{NLS} = 10^{-3} \times 1.199$</td>
</tr>
<tr>
<td></td>
<td>$b_{MLE} = 1.392$</td>
<td>$b_{MLE} = 2.379$</td>
<td>$b_{MLE} = 10^{-3} \times 1.103$</td>
</tr>
<tr>
<td></td>
<td>$tr_{NLS} = 222.533$</td>
<td>$tr_{NLS} = 46.437$</td>
<td>$tr_{NLS} = 10^{-3} \times 1.775$</td>
</tr>
<tr>
<td></td>
<td>$tr_{MLE} = 52.700$</td>
<td>$tr_{MLE} = 10.288$</td>
<td>$tr_{MLE} = 10^{-3} \times 1.155$</td>
</tr>
<tr>
<td></td>
<td>$det_{NLS} = 10^{6} \times 7.734$</td>
<td>$det_{NLS} = 10^{4} \times 1.300$</td>
<td>$det_{NLS} = 10^{-14} \times 2.135$</td>
</tr>
<tr>
<td></td>
<td>$det_{MLE} = 10^{4} \times 2.378$</td>
<td>$det_{MLE} = 30.885$</td>
<td>$det_{MLE} = 10^{-15} \times 3.731$</td>
</tr>
</tbody>
</table>

Table B.5.: sample points uniformly distributed
Table B.6.: sample points cosine distributed over the half-sphere

<table>
<thead>
<tr>
<th></th>
<th>4 points</th>
<th>20 points</th>
<th>100 points</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sigma = 0.1 )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( b_{NLS} )</td>
<td>4.709</td>
<td>( b_{NLS} ) = 10^{-3} \times 2.471</td>
<td>( b_{NLS} ) = 10^{-3} \times 1.314</td>
</tr>
<tr>
<td>( b_{MLE} )</td>
<td>1.728</td>
<td>( b_{MLE} ) = 10^{-2} \times 3.458</td>
<td>( b_{MLE} ) = 10^{-2} \times 2.156</td>
</tr>
<tr>
<td>( \text{tr}_{NLS} )</td>
<td>22.794</td>
<td>( \text{tr}_{NLS} ) = 10^{-2} \times 5.387</td>
<td>( \text{tr}_{NLS} ) = 10^{-3} \times 7.571</td>
</tr>
<tr>
<td>( \text{tr}_{MLE} )</td>
<td>12.314</td>
<td>( \text{tr}_{MLE} ) = 10^{-2} \times 3.709</td>
<td>( \text{tr}_{MLE} ) = 10^{-3} \times 4.931</td>
</tr>
<tr>
<td>( \det_{NLS} )</td>
<td>63.827</td>
<td>( \det_{NLS} ) = 10^{-10} \times 8.558</td>
<td>( \det_{NLS} ) = 10^{-13} \times 5.651</td>
</tr>
<tr>
<td>( \det_{MLE} )</td>
<td>6.698</td>
<td>( \det_{MLE} ) = 10^{-10} \times 1.848</td>
<td>( \det_{MLE} ) = 10^{-14} \times 9.712</td>
</tr>
<tr>
<td>( \sigma = 0.5 )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( b_{NLS} )</td>
<td>5.148</td>
<td>( b_{NLS} ) = 0.062</td>
<td>( b_{NLS} ) = 0.033</td>
</tr>
<tr>
<td>( b_{MLE} )</td>
<td>5.003</td>
<td>( b_{MLE} ) = 0.722</td>
<td>( b_{MLE} ) = 0.491</td>
</tr>
<tr>
<td>( \text{tr}_{NLS} )</td>
<td>30.275</td>
<td>( \text{tr}_{NLS} ) = 1.374</td>
<td>( \text{tr}_{NLS} ) = 0.191</td>
</tr>
<tr>
<td>( \text{tr}_{MLE} )</td>
<td>25.237</td>
<td>( \text{tr}_{MLE} ) = 0.891</td>
<td>( \text{tr}_{MLE} ) = 0.119</td>
</tr>
<tr>
<td>( \det_{NLS} )</td>
<td>668.231</td>
<td>( \det_{NLS} ) = 10^{-4} \times 3.673</td>
<td>( \det_{NLS} ) = 10^{-7} \times 2.156</td>
</tr>
<tr>
<td>( \det_{MLE} )</td>
<td>332.646</td>
<td>( \det_{MLE} ) = 10^{-5} \times 7.978</td>
<td>( \det_{MLE} ) = 10^{-8} \times 3.789</td>
</tr>
<tr>
<td>( \sigma = 1 )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( b_{NLS} )</td>
<td>5.614</td>
<td>( b_{NLS} ) = 0.188</td>
<td>( b_{NLS} ) = 0.116</td>
</tr>
<tr>
<td>( b_{MLE} )</td>
<td>6.703</td>
<td>( b_{MLE} ) = 2.228</td>
<td>( b_{MLE} ) = 1.689</td>
</tr>
<tr>
<td>( \text{tr}_{NLS} )</td>
<td>44.724</td>
<td>( \text{tr}_{NLS} ) = 10.410</td>
<td>( \text{tr}_{NLS} ) = 0.814</td>
</tr>
<tr>
<td>( \text{tr}_{MLE} )</td>
<td>25.119</td>
<td>( \text{tr}_{MLE} ) = 3.130</td>
<td>( \text{tr}_{MLE} ) = 0.439</td>
</tr>
<tr>
<td>( \det_{NLS} )</td>
<td>10^3 \times 6.735</td>
<td>( \det_{NLS} ) = 0.486</td>
<td>( \det_{NLS} ) = 10^{-5} \times 6.278</td>
</tr>
<tr>
<td>( \det_{MLE} )</td>
<td>502.512</td>
<td>( \det_{MLE} ) = 10^{-2} \times 1.985</td>
<td>( \det_{MLE} ) = 10^{-6} \times 9.289</td>
</tr>
<tr>
<td>( \sigma = 5 )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( b_{NLS} )</td>
<td>7.694</td>
<td>( b_{NLS} ) = 7.104</td>
<td>( b_{NLS} ) = 6.249</td>
</tr>
<tr>
<td>( b_{MLE} )</td>
<td>7.765</td>
<td>( b_{MLE} ) = 7.537</td>
<td>( b_{MLE} ) = 7.548</td>
</tr>
<tr>
<td>( \text{tr}_{NLS} )</td>
<td>171.614</td>
<td>( \text{tr}_{NLS} ) = 10^3 \times 1.415</td>
<td>( \text{tr}_{NLS} ) = 13.536</td>
</tr>
<tr>
<td>( \text{tr}_{MLE} )</td>
<td>36.754</td>
<td>( \text{tr}_{MLE} ) = 7.956</td>
<td>( \text{tr}_{MLE} ) = 1.543</td>
</tr>
<tr>
<td>( \det_{NLS} )</td>
<td>10^6 \times 3.108</td>
<td>( \det_{NLS} ) = 10^6 \times 8.432</td>
<td>( \det_{NLS} ) = 34.226</td>
</tr>
<tr>
<td>( \det_{MLE} )</td>
<td>10^3 \times 5.352</td>
<td>( \det_{MLE} ) = 10.865</td>
<td>( \det_{MLE} ) = 10^{-2} \times 1.427</td>
</tr>
</tbody>
</table>
Background on the Hypergeometric Function

In this section we provide some brief background discussion about concepts needed to express the results in section 5.4.

Given any function, one can approximate its value at the point $x = 0$ by its power series expansion

$$f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \ldots \quad (C.1)$$

Conversely, given a sequence of coefficients $\{a_0, a_1, a_2, \ldots\}$, one can attempt to find a function that generates this sequence as coefficients of the respective power in its power series expansion. This function is called a *generating function* of the sequence. For example, the sequence $\{1, 1, 1, \ldots\}$ is generated by the function $f(x) = \frac{1}{1-x}$.

A sequence which fulfills the property that the ratio of consecutive terms can be written as a rational function of polynomials in the index, is called a *hypergeometric sequence*, i.e.

$$\frac{c_{k+1}}{c_k} = \frac{P(k)}{Q(k)} = \frac{(k + a_1)(k + a_2)\ldots(k + a_p)}{(k + b_1)(k + b_2)\ldots(k + b_q)(k + 1)} \quad (C.2)$$

where the factor $(k + 1)$ in the denominator is present for historical reasons of notation. A *hypergeometric function* is a function that has a hypergeometric series (i.e. one whose coefficients
Background on the Hypergeometric Function

A hypergeometric sequence) as its power series expansion, i.e.

\[ \begin{align*}
_\text{p}F_\text{q}(x) &= _\text{p}F_\text{q}(a_1, \ldots, a_\text{p}, b_1, \ldots, b_\text{q}, x) \\
&= \sum_k c_k x^k \\
&= \sum_k \frac{(a_1)_k (a_2)_k \ldots (a_\text{p})_k x^k}{(b_1)_k (b_2)_k \ldots (b_\text{q})_k k!},
\end{align*} \]

where consecutive coefficients \(c_k\) and \(c_{k+1}\) fulfill (C.2) and \((a)_k\) is the Pochhammer symbol or rising factorial \((a)_k = \frac{\Gamma(a+k)}{\Gamma(a)} = a(a+1)\ldots(a+k-1)\).

Some hypergeometric functions have specific names. For example, \(_0F_1\) is called the confluent hypergeometric limit function and often arises in statistical physics as well as in section 5.4. For more details about hypergeometric functions see for example [LBC10].

Note that some well-known functions can be expressed as hypergeometric functions, e.g.

\[ \begin{align*}
_0F_1 \left( \frac{1}{2}, \frac{x^2}{4} \right) &= \cosh(x), \\
_0F_1 \left( 1, \frac{x^2}{4} \right) &= I_0(x), \\
_0F_1 \left( \frac{3}{2}, \frac{x^2}{4} \right) &= \frac{\sinh(x)}{x},
\end{align*} \]

where \(I_0(x)\) denotes the modified Bessel function of the first kind.

Among many other representations, the hypergeometric limit function \(_0F_1\) has the integral representation

\[ _0F_1(b, x) = \frac{\Gamma(b)}{\sqrt{\pi} \Gamma \left( b - \frac{1}{2} \right)} \int_0^\pi \exp \left( -2\sqrt{x} \cos t \right) (\sin t)^{2b-2} \, dt, \]

if the real part of the argument \(b, \text{Re}(b)\), is larger than 1/2. If we substitute \(b = n/2\) with \(n > 1\), we get

\[ _0F_1 \left( \frac{n}{2}, x \right) = \frac{\Gamma \left( \frac{n}{2} \right)}{\sqrt{\pi} \Gamma \left( \frac{n-1}{2} \right)} \int_0^\pi \exp \left( -2\sqrt{x} \cos t \right) (\sin t)^{n-2} \, dt. \]

We invoked this result (C.5) in section 5.4.


