Statistics on Manifolds with Applications to Modeling Shape Deformations

by

Oren Freifeld

B.Sc., Tel-Aviv University; Tel-Aviv, Israel, 2005

M.Sc., Tel-Aviv University; Tel-Aviv, Israel, 2007

Sc.M., Brown University; Providence, RI, 2009

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in The Division of Applied Mathematics at Brown University

PROVIDENCE, RHODE ISLAND

May 2014

This dissertation by Oren Freifeld is accepted in its present form
by The Division of Applied Mathematics as satisfying the
dissertation requirement for the degree of Doctor of Philosophy.

Date_____ _____

Michael J. Black, Ph.D., Advisor

Recommended to the Graduate Council

Date_____ _____

Élie Lucien Bienenstock, Ph.D., Reader

Date_____ _____

Erik B. Sudderth, Ph.D., Reader

Date_____ _____

John W. Fisher III, Ph.D., Reader

Approved by the Graduate Council

Date_____ _____

Peter Weber, Dean of the Graduate School

Vitae

Oren Freifeld was born in Israel during the 20th century. He grew up[1] in Givatayim, and, following three years in the Israel Defense Forces, enrolled as a student at Tel-Aviv University. He received his B.Sc. and M.Sc. degrees, both in Biomedical Engineering, in 2005 and 2007, respectively. In September 2007, he joined the Applied Mathematics graduate program at Brown University. In 2009, he received his Sc.M. degree. During the course of his doctoral studies, he was also an exchange scholar as well as a regular visitor at the department of Electrical Engineering, Stanford University. In June 2013, he successfully defended his Ph.D. thesis, following which this dissertation has been accepted in August 2013. In Fall 2013 he will start his next position as postdoctoral associate at the Massachusetts Institute of Technology.

[1] Note this claim is often disputed [40].

Acknowledgements

First and foremost, I thank my advisor, Prof. Michael Black, for his guidance, mentorship, and help throughout my PhD. Michael's genuine enthusiasm and passion for research are contagious. His deep insights and original way of thinking had a tremendous impact on me and reshaped the way I think. I am fortunate for having had the opportunity to work with Michael and the many lessons he taught me will remain with me in years to come. I am grateful for his invaluable support during the recent years, as well as for arranging my visit at Stanford.

I thank the members of my PhD committee – Prof. Élie Bienenstock, Prof. Erik Sudderth and Prof. John Fisher – for their feedback and useful comments that helped me improve this dissertation.

I thank Prof. Krishna Shenoy, my host at Stanford, for the unique opportunity to be involved in an exciting research project in the field of neuroscience and neural prosthetics. The work on this project (led by Krishna's students, Justin Foster and Paul Nuyujukian) is not covered in this dissertation – but see Justin's upcoming dissertation and its associated publications. The amazing science done in Krishna's lab is inspiring, and his kindness and hospitality contributed greatly to my positive experience at Stanford.

During my PhD I enjoyed interacting with many colleagues and friends at Michael's Vision lab at Brown and later at his Perceiving Systems Department at the Max Planck Institute for Intelligent Systems. They helped me in many ways. Especially, I thank Matt Loper for his great work on registering datasets of human meshes, and my co-authors – Alex Weiss, Silvia Zuffi, Peng Guan and Søren Hauberg – for all their priceless contri-

butions. I also thank Loretta Reiss for her help with the biometric measurements, Alex Bălan for making his SCAPE implementation available to me, and Melanie Feldhofer for helping me numerous times. I also benefited from fruitful discussions with my colleagues, in particular Søren Hauberg, Silvia Zuffi, Alex Weiss, Matt Loper, Eric Rachlin, David Hirshberg, Deqing Sun, Peng Guan, Javier Romero, Jonas Wulff and Aggeliki Tsoli.

Being a PhD student in Applied Mathematics at Brown was a splendid experience for me, during which I have learned more than I had ever hoped to. Coming from an engineering background to graduate-level classes in mathematics was a challenging experience for me. Fortunately, I had great math teachers who helped me along the way. In particular, I thank Prof. Stu Geman and Prof. Basilis Gidas for teaching me advanced statistics, Prof. Boris Rozovsky for teaching me measure-theoretic probability, and Prof. Constantine Dafermos and Prof. Jill Pipher for teaching me analysis.

I thank my friends and fellow Applied Math students for their help and support, especially during my first years at Brown. The list is long, so I will mention only a few of them here: Christian Pfrang, Mario Micheli, Mahir Hadzic, Sergey Kushnarev, Andreas Kloeckner, Lo-Bin Chang, Luan Lin, Kenny Chowdhary, Chia Ying Lee, Ravi Srinivasan, Akil Narayan, Yi Cai, Lauren Alpert, and Seshu Tirupathi.

Finally, I want to thank my family for all their love and support. Words cannot express how much I am grateful to my parents, Hadassah and Igal, for everything they have done for me; I will always cherish their endless giving. I am grateful to my grandparents. In particular, without their inspiring struggles to survive through the hardest times – all of whom are holocaust survivors – I, literally speaking, would not have been here today. I also thank my sister, Noa, and my brother, Tamir. Last but not least, I thank Limor, my wife. I could not have accomplished this journey without her, and I can attest that Miracle Max had it almost[2] right: true love is the greatest thing in the world.

[2]MLT sandwiches are highly overrated.

This dissertation is dedicated with my admiration, respect, and affection, to the memory of Eyal Weiss, 1967-2002.

Abstract of "Statistics on Manifolds with Applications to Modeling Shape Deformations" by Oren Freifeld, Ph.D., Brown University, May 2014

Statistical models of non-rigid deformable shape have wide application in many fields, including computer vision, computer graphics, and biometry. We show that shape deformations are well represented through nonlinear manifolds that are also matrix Lie groups. These pattern-theoretic representations lead to several advantages over other alternatives, including a principled measure of shape dissimilarity and a natural way to compose deformations. Moreover, they enable building models using statistics on manifolds. Consequently, such models are superior to those based on Euclidean representations. We demonstrate this by modeling 2D and 3D human body shape. Shape deformations are only one example of manifold-valued data. More generally, in many computer-vision and machine-learning problems, nonlinear manifold representations arise naturally and provide a powerful alternative to Euclidean representations. Statistics is traditionally concerned with data in a Euclidean space, relying on the linear structure and the distances associated with such a space; this renders it inappropriate for nonlinear spaces. Statistics can, however, be generalized to nonlinear manifolds. Moreover, by respecting the underlying geometry, the statistical models result in not only more effective analysis but also consistent synthesis. We go beyond previous work on statistics on manifolds by showing how, even on these curved spaces, problems related to modeling a class from scarce data can be dealt with by leveraging information from related classes residing in different regions of the space. We show the usefulness of our approach with 3D shape deformations. To summarize our main contributions: 1) We define a new 2D articulated model – more expressive than traditional ones – of deformable human shape that factors body-shape, pose, and camera variations. Its high realism is obtained from training data generated from a detailed 3D model. 2) We define a new manifold-based representation of 3D shape deformations that yields statistical deformable-template models that are better than the current state-of-the-art. 3) We generalize a transfer learning idea from Euclidean spaces to Riemannian manifolds. This work demonstrates the value of modeling manifold-valued data and their statistics explicitly on the manifold. Specifically, the methods here provide new tools for shape analysis.

Contents

List of Tables

List of Figures

I am in shape. Round is a shape.

Garfield

Chapter 1

Introduction

1.1 Deformable Shape Models

Statistical generative models of non-rigid deformable shape have numerous applications in computer vision, biometry, computer graphics, medical imaging, robotics, and other domains. The choice of *shape representation* has a great influence on the effectiveness of the statistical models. While shape representations come in different flavors, in the context of statistical modeling of deformable shape there are several properties we may desire the representation to possess. In particular, it is often desirable that the representation should:

1. support *shape synthesis*, not just *shape analysis*;
2. lend itself to composition of more than one type of shape deformations (*e.g.*, two human shapes may differ from each other due to differences not only in physique but also in pose);
3. be differentiable;
4. allow for a meaningful notion of shape (dis)similarity.

While the vast literature on shape analysis provides a variety of different shape representations, many of which exhibit the first property, the second property suggests focusing on *transformations* of shapes, rather than on the shapes themselves. In particular, this brings to mind the notion of a *group of transformations*. The algebraic structure of a group implies certain favorable *closure* properties, which in turn are translated into a *consistent* representation. Of course, a group structure alone is not enough – to build meaningful statistical models one needs to be able to do calculus (*e.g.*, take derivatives - as mentioned in the third property) and, more importantly, to measure distances. Thus, the groups of interest here belong to a special class called (finite-dimensional) *Lie groups*. These mathematical objects are, among other things, *smooth* manifolds – usually nonlinear – upon which, once endowed with a *Riemannian structure*, *geodesic distances* can be measured; this provides a principled way to accommodate for (dis)similarity – the last property in the above list.

To be more concrete, later on we will describe how the shape deformations of 2D polygonal curves can be represented through a particular nonlinear Lie group. While this group was previously considered applicable only for open curves [53] we will show that in fact there is a way to employ this group for closed curves as well. We will also define a novel nonlinear Lie group for the representation of shape deformations of 3D triangular surfaces.

Remark 1.1.1 (Statistical generative shape models). The way we use the term *statistical generative model* in this work involves a slight abuse of terminology. Our meaning here is a statistical model that can be used for not only assigning (non-normalized) probabilities to *features* of a shape but also generating a new shape. In other words, we mean a statistical model of shape that supports shape synthesis; implicitly, this also assumes that the shape representation too supports shape synthesis[1]. We do not necessarily require, however, that *every* possible new shape can be generated from the model. Thus, for example, for our purposes we are content to include Principal Component Analysis under the category of statistical generative models of shape, although strictly speaking, PCA is not a generative

[1]The notions of synthesis and analysis will be explained in more detail later on.

model [125].

At first, switching to manifold-based representation seems to introduce a major obstacle. After all, *classical statistics* is concerned with $\mathbb{R}^n$-valued data, and relies on both the linear structure and the distances associated with $\mathbb{R}^n$; consequently, it is not suitable for such nonlinear spaces. It turns out, however, that many statistical tools can be generalized from $\mathbb{R}^n$ to manifold-valued data. The branch of statistics that addresses these generalizations is often called *statistics on manifolds*. Moreover, statistical models on manifolds are usually superior to their $\mathbb{R}^n$-counterparts. In fact, when the underlying geometry of the space of interest is taken into account, the statistical modeling results in not only more effective analysis but also consistent synthesis; namely, a sample from the model is guaranteed to lie on the manifold.

Another possible concern is whether these benefits come at a dire computational cost. Fortunately, this is not the case as usually the computations are only slightly more complicated than the in case of $\mathbb{R}^n$. In fact, sometimes the use of manifolds *lowers* the computational burden due to a (usually) lower number of degrees of freedom.

In Chapter 3 we will focus on deformations of 2D contours, while 3D deformable surfaces are the topic of Chapter 4. In both cases, the building blocks creating a global non-rigid deformation are local transformations acting on basic units – also known as *primitives* – that comprise the template. As we will see, in the 2D case the primitives are the small directed line segments that form the polygon while in the 3D case these are the triangles that the mesh consists of, expressed through pairs of directed (3D) line segments.

1.1.1 Modeling Transformations Acting on a Template

Modeling transformations, rather than modeling the transformed objects themselves, is pivotal in Ulf Grenander's *Pattern Theory* [51, 52, 55]; see also [107]. One motivation for such an approach is that usually it simplifies the resulting statistical models. A second

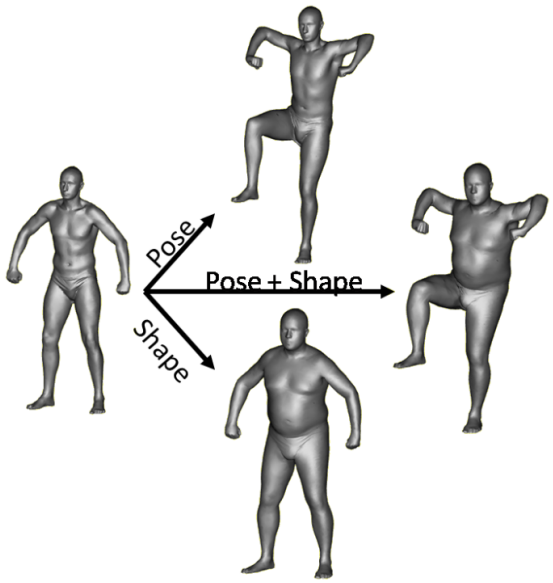
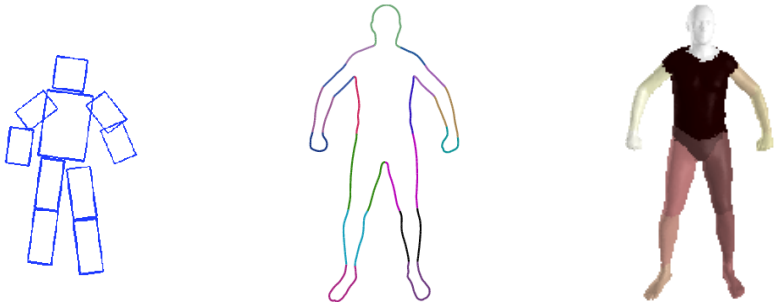


Figure 1.1: SCAPE [6]: a deformable template model that enables the composition of shape and pose deformations.

motivation is related to the fact that transformations lend themselves more easily to *composition* of different sources of variation. In practice this means that a deformation can be factored into its different types, and each type can be modeled separately (perhaps with statistical dependence between those types). Then, upon sampling from the model, different types of deformations are composed to produce the resulting total deformation. For example, SCAPE [6] is a factored model of 3D shape and pose that enables the composition of two types of deformations; see Fig. 1.1. In Chapter 3 we will introduce a 2D model designed for composition of three types of deformations: pose, body-shape, and camera variation.

Remark 1.1.2 (Composition *vs*. compositionality). Note that the notion of *composition*, as used here, is different from the notion of *compositionality* in computer vision as advocated by Bienenstock, S. Geman, and their colleagues [11,47] (see also theses by Potter and Zhang [117, 153]). There, the compositional aspect refers to a multi-layer hierarchical structure and re-usability of parts. Likewise, and for similar reasons, our notion of composition is also different from the notion of composition as used in the *inference-by-composition* approach of Irani *et al*. [29, 74].



(a) Pictorial Structures (b) Contour Person (c) 3D model

Figure 1.2: The Contour Person model. Most 2D body models are based on a simple shape representation that consists of a simple articulated collection of geometric primitives (a) while 3D models have become increasingly detailed and realistic (c). The Contour Person model is based on a more detailed representation of 2D shape (b), leading to realism akin to modern 3D models but with computational benefits not far from those of traditional 2D models.

1.1.2 2D Deformable Shapes

The detection of people and the analysis of their pose in images or video has many applications and has drawn significant attention. In the case of uncalibrated monocular images and video, 2D models dominate while in calibrated or multi-camera settings, 3D models are popular. In recent years, 3D articulated models of the human body have become sophisticated and highly detailed, with the ability to accurately model human shape and pose [6,16,63,72] Fig. 1.2c). In contrast, 2D articulated models typically treat the body as a collection of polygonal regions that only crudely capture body shape Fig. 1.2a) [18,32,71,76,119]. Two-dimensional models are popular because they are relatively low dimensional, do not require camera calibration, and admit computationally attractive inference methods (e.g. with belief propagation [4, 30, 86, 133]). For many problems such as pedestrian detection, full 3D reasoning many not be needed, and 2D models suffice. While such 2D models predominate, they have changed little in 20 or more years [71, 76].

In Chapter 3 we describe a new 2D model of the human body that has many of the benefits of the more sophisticated 3D models while retaining the computational advantages of 2D. This *Contour Person* (CP) model (Fig. 1.2b) provides a detailed 2D representation of natural body shape and captures how it varies across a population. It retains, however,

the part-based representation of current 2D models as illustrated by the different colors in Fig. 1.2b. An articulated, part-based, model is required for pose estimation using inference methods such as belief propagation. Importantly, the CP model also captures the non-rigid deformation of the body that occurs with articulation. This allows the contour model to accurately represent a wide range of human shapes and poses. Like other 2D body models, the approach is inherently view-based with 2D models constructed for a range of viewing directions.

1.1.3 3D Deformable Shapes

Three dimensional mesh models of objects play a central role in many computer vision algorithms that perform *analysis-by-synthesis* [16, 62, 91, 128, 149]. Capturing the variability of 3D meshes for an object class is critical and the increasing availability of 3D mesh data enables the employment of statistical learning methods for building such models. In particular, for capturing human shape variation, *deformable template* models are popular for representing non-rigid deformations and articulations [6, 16, 48, 58, 62, 72, 149]. Such models have wide application in computer vision, computer graphics, virtual reality, shape compression, biometrics, and the fashion industry. With some notable exceptions [63], current methods typically use a Euclidean representation of deformations and measure distance in a Euclidean space, ignoring the geometry of the space of deformations. These methods model triangle deformations as elements of $\mathbb{R}^{3\times 3}$ while deformations live, in fact, in a 6 dimensional nonlinear manifold. Despite the use of heuristics to remove excessive degrees of freedom (DoF) their deformations might still be noisy or have negative determinant (especially during synthesis). The latter is physically impossible (e.g. reflections). In contrast, in Chapter 4 we propose a novel manifold representation of shape deformations that eliminates the above problems and has many other benefits, both practical and theoretical. In particular, respecting the underlying geometry enables better statistical learning methods, distance computation, shape interpolation, and the valid composition of multiple causes of shape deformation.

1.2 Manifold Representations

Shape deformations are only one example for manifold-valued data. More generally, in many computer-vision and machine-learning problems, nonlinear manifold representations arise naturally, while in others such representations provide a powerful alternative to their $\mathbb{R}^n$ counterparts. In this work we show how these ideas lead to better statistical deformable-template shape models and that, more generally than the spaces of shape deformations, certain *transfer learning*[2] problems – traditionally limited to $\mathbb{R}^n$-valued data – can be solved even on such curved metric spaces. Thus, while a large part of this thesis is dedicated to the topic of statistical modeling of shape deformations, many of the ideas we will see later on have a broader scope. In particular, see Chapter 5.

As stated before, classical data analysis is concerned with $\mathbb{R}^n$-valued data. Namely, it assumes that the data points, usually denoted by $\{x_1, x_2, \ldots, x_N\}$, satisfy

$$\{x_1, x_2, \ldots, x_N\} \subset \mathbb{R}^n \,, \tag{1.1}$$

N being the number of data points. In fact, this assumption is often made automatically without stopping to reflect about its implications; the main ones pertain to the linear structure of $\mathbb{R}^n$ and to its associated distance functions. While the simplicity of $\mathbb{R}^n$ conveys an understandable appeal, there are numerous situations where a different point of view is in order.

A finite-dimensional manifold, to be defined in Chapter 2, is a certain generalization of a Euclidean space[3], and is usually nonlinear. In what follows, we describe several of the motivations for manifold representations.

[2] Transfer learning means, in our context, an approach that utilizes knowledge obtained from learning a model of one class in order to improve the modeling of another related class.

[3] Analogously, infinite-dimensional manifolds are a generalization of infinite-dimensional Hilbert spaces. Infinite-dimensional are beyond the scope of this work.

1.2.1 Nonlinearity

Frequently, it is known that the data, while being $\mathbb{R}^n$-valued, must satisfy one or more constraints that render the linear structure assumption invalid. In other words,

$$\{x_1, x_2, \ldots, x_N\} \subset S \subset \mathbb{R}^n \,, \tag{1.2}$$

where S is the set of all points that satisfy the constraints, but S is not linear. We now give several important examples.

1.2.1.1 Spherical Data

There are numerous cases in computer vision where the data are known to be living on a sphere. To give a few examples, spherical data appear in the context of panoramic mosaics or omni-directional images [100], normalized SIFT features [79], Kendall's pre-shape spaces [80], and 3D surface normals.

As we will see in Chapter 2 (see Examples 2.1.1 and 2.1.2), a sphere is not a linear space. In fact, it is not even convex[4]. One *statistical implication* of the lack of convexity is that $\hat{\mu} \stackrel{\text{def}}{=} \frac{1}{N}\sum_{i=1}^{N} x_i$, the sample mean[5] of the data, usually falls outside the space of interest: $\hat{\mu} \notin S$. In section 1.2.2 we will see that even when the sample mean happens to fall inside the space, the result typically leaves much to be desired.

[4] A convex set S is a set that is closed under convex combinations: $x_1, x_2 \in S$ implies $(1-c)x_1 + cx_2 \in S$ for every $c \in [0, 1]$; this is a weaker requirement than closure under linear combinations.

[5] As is common in statistics, we reserve the symbol μ, without the "hat", for the "true" unknown value of the mean, while the "hat" stands for its estimator.

1.2.1.2 Matrix-Valued Data

The space of $m \times n$ matrices, denoted by $\mathbb{R}^{m\times n}$, is a linear space. Matrix-valued data, namely

$$\{x_1, x_2, \ldots, x_N\} \subset \mathbb{R}^{m\times n}\,, \tag{1.3}$$

can be seen as $\mathbb{R}^{mn}$-valued data via any natural correspondence between $\mathbb{R}^{m\times n}$ and $\mathbb{R}^{mn}$ and appear in many applications. While $\mathbb{R}^{m\times n}$ is a linear space, there are often nonlinear constraints on matrix-valued data. For example, when $m = n$ (*i.e.*, square matrices), determinant-related constraints frequently occur. Several examples for such constraints will be described in Section 2.1.2. Another important class of square matrices is the set of Symmetric Positive Definite (SPD) matrices. SPD-matrix-valued data have many applications in computer vision. For example, non-degenerate covariance matrices of pixel-wise image features provide a stable image region descriptor with certain additional attractive properties [116,145]. We will make use of this region descriptor in some of our experiments in Chapter 5. Another example is given by certain modalities in medical imaging, such as Diffusion Tensor Images (DTI) [7], that provide SPD-matrix-valued images; namely, the measurement at each pixel (or voxel) corresponds to an SPD matrix. As we will see in Section 2.1.2, SPD matrices do not form a linear space. For additional applications of matrix-valued images, including the registration of two depth-images through a matrix-valued image that captures the rigid transformations between corresponding pixels, see works by Rosman *et al.* [121–123].

1.2.1.3 Shape Deformations

Two particular kinds of matrix-valued data, with a nonlinear structure, pertain to the representation of shape deformations. The first is related to deformable 2D or 3D polygonal curves [43, 51–53]. In Chapter 3 we will elaborate about this representation. In Chapter 4 we will define a second kind, based on a novel nonlinear space of shape deformations of 3D meshes [41].

1.2.1.4 Additional Nonlinear Spaces

There exist many additional nonlinear spaces of interest. Two closely related examples are the Stiefel and Grassmannian manifolds; namely, the spaces of $n \times k$ "tall-skinny" ($n > k$) orthogonal matrices and k-dimensional subspaces of $\mathbb{R}^n$, respectively [1]. For several computer vision and machine learning applications of data in these spaces, see [9, 17, 68, 127, 134, 144]. Another example is the space of probability density functions which can also be related to the spaces of re-parameterizations of 2D curves and time-warping used in shape analysis and activity analysis respectively [138]. In the interest of space, we avoid going into further details.

1.2.2 Distances

Let S denote some space. A distance function on S is a function, $d : S \times S \to \mathbb{R}^+$, that measures how (dis)similar two elements in S are. The lower the value of $d(x, y)$ is, the more x and y are similar to each other. For $\mathbb{R}^n$, the typical distance function is given by the ℓ^2 distance; namely the ℓ^2 norm of the Euclidean difference between x and y:

$$d(x, y)_{\ell^2} \stackrel{\text{def}}{=} \|x - y\|_{\ell^2} = \left(\sum_{i=1}^{n} |x_i - y_i|^2 \right)^{1/2} . \tag{1.4}$$

Remark 1.2.1. Note that while at the moment we do not require d to be a *metric* (see Definition A.3.1, page 204), all of the distance functions considered in this work are in fact metrics.

In many cases, especially but not limited to nonlinear spaces, the ℓ^2 distance is inap-

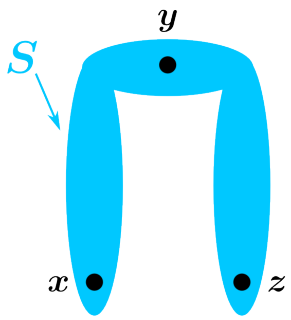


Figure 1.3: Consider the three points $x, y, z \in S$. When paths are restricted to lie in S (light blue), which one is closest to z: x or y?

propriate. This is also true for the other usual suspects:

$$d(x,y)_{\ell^p} \stackrel{\text{def}}{=} \|x-y\|_{\ell^p} = \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p} \quad , \quad 1 \leq p < \infty \; ; \tag{1.5}$$

$$d(x,y)_{\infty} \stackrel{\text{def}}{=} \|x-y\|_{\ell^\infty} = \max_{i \in \{1,\ldots,n\}} |x_i - y_i| \quad , \quad p = \infty \; ; \tag{1.6}$$

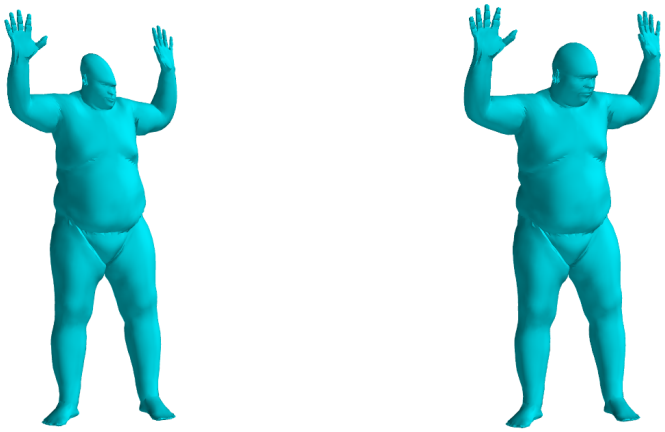
$$d(x,y)_{\Sigma} \stackrel{\text{def}}{=} (x-y)^T \Sigma^{-1} (x-y) \quad , \quad \Sigma^{-1} \in \text{SPD} \; , \tag{1.7}$$

where in the particular cases of taking $p = 2$ in Eqn. (1.5) or $\Sigma = I$ in Eqn. (1.7) we recover the ℓ^2 distance.

Remark 1.2.2. For a situation where the space is linear but still none of the usual distance functions will do, consider *multi-metric learning*. In this branch of machine learning, single-metric learning in $\mathbb{R}^n$ (as done in, *e.g.*, [130, 131, 151, 152]) is generalized to learning multiple *local* metrics in $\mathbb{R}^n$ [44, 45, 64, 101, 118, 148]. Of note, these methods are unable to yield a metric space, compromising their applicability to certain statistical applications such as regression or PCA. As it turns out, however, multi-metric learning can still give rise to a *global* metric; this is achieved through adopting a Riemannian manifold perspective; see our recent work in Hauberg *et al.* [65].

Our first example addresses a simple issue, illustrated in Fig. 1.3 with three points in the 2D plane. Which point is closer to z: x or y? In $\mathbb{R}^2$ (with the ℓ^2 distance) the answer is trivial: x is closer to z than y is. But what if our space is S (see Fig. 1.3)? In which case, it would seem odd to claim that x is still the right answer.

(a) $\mathcal{T}$ (b) Applying Q_1 to $\mathcal{T}$ (c) Applying Q_2 to $\mathcal{T}$



(d) Applying $\frac{Q_1+Q_2}{2}$ to $\mathcal{T}$ (e) Applying $f(\mathcal{Q}_1, \mathcal{Q}_2)$ to $\mathcal{T}$

Figure 1.4: Linear interpolation of shape deformations produces an unsatisfying result, suggesting *a problem with the distance function.* (a) $\mathcal{T}$, a template shape (b) Applying $\mathcal{Q}_1$ to $\mathcal{T}$ (c) Applying $\mathcal{Q}_2$ to $\mathcal{T}$ (d) Applying $(\mathcal{Q}_1 + \mathcal{Q}_2)/2$ to $\mathcal{T}$ produces an unsatisfying result (e) A better result is obtained from applying $f(\mathcal{Q}_1, \mathcal{Q}_2)$ to $\mathcal{T}$. See text for more details.

Our second example is related to deformable-template models, a topic to be described in detail later on. For now, Fig. 1.4 will suffice to illustrate the problem. Fig. 1.4a shows a template 3D shape, denoted by $\mathcal{T}$, while figures 1.4b and 1.4c show two other shapes. We think of the last two as deformations from $\mathcal{T}$, and we denote the associated deformations by $\mathcal{Q}_1$ and $\mathcal{Q}_2$ respectively. As we have yet to define deformations properly, our discussion will remain abstract at the moment. Suppose now that we would like to find the average deformation of the shapes shown in Figures 1.4b and 1.4c. It turns out that linear interpolation, namely $(\mathcal{Q}_1 + \mathcal{Q}_2)/2$, provides a rather poor result: as can be seen in Fig. 1.4d, while the outcome may be considered more or less reasonable in the lower part of the body (where $\mathcal{Q}_1$ and $\mathcal{Q}_2$ have similar effects), in the upper part of the body (where their effects differ more significantly) it is clearly disappointing; it seems that some unnatural "squashing" effect has taken place.

In broad sweeps, let us try to explain what is going on; any omitted details will be given when we return to this example in Chapter 4. We start with a well known fact.

Fact 1.2.1. The sample mean, in $\mathbb{R}^n$, coincides with

$$\arg\inf_{x\in\mathbb{R}^n} \sum_{i=1}^{N} d(x_i, x)^2_{\ell^2} \; ; \tag{1.8}$$

namely, the minimizer of the sum of squared ℓ^2 distances from the data[6,7].

Consequently, and as a particular case, when there are only two data points ($N = 2$) the sample mean linearly interpolates between them, and is equidistant from both:

$$\hat{\mu} \overset{\text{by def.}}{=} \frac{x_1 + x_2}{2} \; , \; d(x_1, \hat{\mu}) = d(x_2, \hat{\mu}) \; . \tag{1.9}$$

Thus, what Fig. 1.4e suggests is that there is something wrong with the distance. While

[6]The proof is easy. Equate the gradient (with respect to x) of the convex cost function from Equation (1.8) to $\mathbf{0}_n \in \mathbb{R}^n$ and solve for x to get the global minimizer, $\frac{1}{N}\sum_{i=1}^{N} x_i \in \mathbb{R}^n$.

[7]Analogously, the multi-variate generalization of the sample median coincides with $\arg\inf_{x\in\mathbb{R}^n} \sum_{i=1}^{N} d(x_i, x)_{\ell^1}$.

the Euclidean averaging did produce *some* shape deformation, it is not one we want. In other words, we successfully minimized the *wrong cost function*. In Chapter 4 we will see that shape deformations can be restricted to lie on a manifold, and that this manifold is equipped with a better notion of distance. Once this distance is substituted in Eqn. (1.8) instead of the ℓ^2 distance, we get a different minimizer. To stress this minimizer is still a function of the data points $\mathcal{Q}_1$ and $\mathcal{Q}_2$, we denote it by $f(\mathcal{Q}_1, \mathcal{Q}_2)$. Note that $f(\mathcal{Q}_1, \mathcal{Q}_2)$ produces a more successful result than $\frac{\mathcal{Q}_1+\mathcal{Q}_2}{2}$; see Fig. 1.4e.

The purpose of the above discussion was to show that there might be a problem with not only linear interpolation but also, and more importantly, the familiar distances from $\mathbb{R}^n$. As discussed in the next section, the latter point should alarm any one who wants to do statistics on manifold-valued data.

1.2.3 Probability and Statistics Depend on Distances and the Structure of the Space

In the previous sections we discussed how both the linear structure and its implied distances are not necessarily appropriate for various applications. Both classical theories of probability[8] and statistics have a strong dependency on these concepts[9]. To fix ideas, let us start with an example from probability; the corresponding statistical example will follow naturally.

Example 1.2.1 (Gaussian random variables and distances)**.** Let the n-dimensional random variable X be normally distributed with some mean, μ, and a unit covariance:

$$X \sim \mathcal{N}(\mu, I_{n\times n}) \ , \ \mu \in \mathbb{R}^n \ . \tag{1.10}$$

[8] Let us pay our measure-theoretic taxes at the outset: while measure-theoretic probability supplies the mathematical rigor justifying many of the probabilistic and statistical results mentioned in this work, it is at best tangential to our main discussion; therefore, we shall not worry ourselves with any measure-theoretic issues.

[9] A purist may argue that the dependency on distances can happen only when the discussion is restricted to random variables taking values in metric spaces – as opposed to, say, abstract topological spaces; however, this is indeed the case in most real-world applications.

The probability density function (PDF) of X is a *function of the distance* between the point of interest, x, and μ:

$$p(x|\mu) \propto e^{-\frac{1}{2}d(x,\mu)^2_{\ell^2}} = \text{func}(d(x,\mu)_{\ell^2}) \ , \tag{1.11}$$

Thus, $\Pr(X \in B)$, where $B \subset \mathbb{R}^n$, depends on

$$\{d(x,\mu)_{\ell^2} : x \in B\} \ ;$$

namely, the set of ℓ^2 distances between μ and each one of the points that B consists of. More generally, if $X \sim \mathcal{N}(\mu, \Sigma)$, then

$$p(x|\mu,\Sigma) = \text{func}(d(x,\mu)_\Sigma, \det\Sigma) \ . \tag{1.12}$$

and $\Pr(X \in B)$ depends on

$$\{d(x,\mu)_\Sigma : x \in B\}$$

(as well as on $\det\Sigma$).

Example 1.2.1 tells us that the Gaussian, which is perhaps the most commonly used PDF, has a strong dependency on the concept of a distance. If the ℓ^2 distance (or $d(\cdot,\cdot)_\Sigma$) is inappropriate, then we cannot use the Gaussian PDF as it is. We can now turn to the complementary side of the token: statistics.

Example 1.2.2 (Sample mean for normally distributed data)**.** Let $\{x_i\}_{i=1}^N$ be an Independent and Identically Distributed (*i.i.d.*) sequence of samples[10]. $x_i \sim \mathcal{N}(\mu, I_{n\times n})$ where $\mu \in \mathbb{R}^n$. Let $L(\mu) \overset{\text{def}}{=} \prod_{i=1}^N p(x_i|\mu)$ denote the likelihood of the observed sequence. The log-likelihood, $\log L(\mu)$, is proportional to the cost function in Equation (1.8), and thus

[10]Usually it is a good practice to reserve capital letters for random variables, *e.g.*, X, and use lowercase letters for their realized values, e.g, x; this creates the required distinction between the function (*i.e.*, the random variable) and the values it takes. Thus, one can write, say, $\Pr(X \leq x)$. In this work, however, this practice will be shamelessly ignored more often than not; we will later need notations from manifold theory, and then this practice would have led to a notational clash. Therefore we downgrade it from a rule to a guideline. This annoyance arises often in texts that cover both statistics and manifolds, and it seems that an agreed-upon solution has yet to be found.

the sample mean is the maximum-likelihood estimator of μ:

$$\hat{\mu} \overset{\text{by def.}}{=} \frac{1}{N}\sum_{i=1}^{N} x_i = \arg\sup_{\mu\in\mathbb{R}^n} L(\mu) = \arg\inf_{\mu\in\mathbb{R}^n} \sum_{i=1}^{N} d(x_i,\mu)_{\ell^2}^2 \ . \tag{1.13}$$

It is easy to verify that changing the distribution from $\mathcal{N}(\mu, I_{n\times n})$ to the more general $\mathcal{N}(\mu,\Sigma)$ would not have changed this result.

In Example 1.2.2 we see that the likelihood being maximized (equivalently, the standard loss function being minimized) is based on distances, and that the optimizer is a linear combination of the data. In cases where the ℓ^2 distance (or $d(\cdot,\cdot)_\Sigma$) and/or the linear structure are inappropriate, we see that even simple statistical operations such as computing the sample mean for normally distributed data can fail.

The Gaussian is of course only one example. The take-home message in the context of probability or statistics is simple: when the structure of the space is nonlinear, or when the standard distances of $\mathbb{R}^n$ will not do – typically, but not always, these two phenomena happen at the same time – it is unwise or even impossible to keep doing the statistical analysis as if we were still working in $\mathbb{R}^n$.

This naturally raises the question: how can we do *statistics on manifolds*? Some answers will be provided in Section 2.5, when we touch upon several basic techniques for doing statistics on manifolds. Later on, in Chapter 5, we will introduce a novel technique that goes beyond previous work to provide the first generalization of *transfer learning*, from $\mathbb{R}^n$ to a manifold setting. This will enable us to addresses data scarcity of one manifold-valued class by leveraging data from another manifold-valued class.

1.2.4 Consistency

Earlier we saw a situation where a natural operation, such an averaging, is applied to data points, and yet the result falls outside the space. In other words, we saw a situation where

a natural operation is applied to "good" elements, yet it produces a "bad" one. In which case we can say that the representation (or the operation - depending on the point of view) is *inconsistent*. Phrasing it a bit differently, lack of consistency can be thought of as lack of *closure*. Manifolds provide us with consistent representations in the sense that as long as we make sure to use the right operations, we are guaranteed to stay on the manifold, and so the constraints that define the manifold remain satisfied. Often we will be interested in algebraic closure under specific operations, such as inversion of an element and composition of two elements (*i.e.*, *a group structure*). As discussed earlier, this will lead us the concept of a particular type of manifolds, known as *matrix Lie groups*; see Section 2.3.

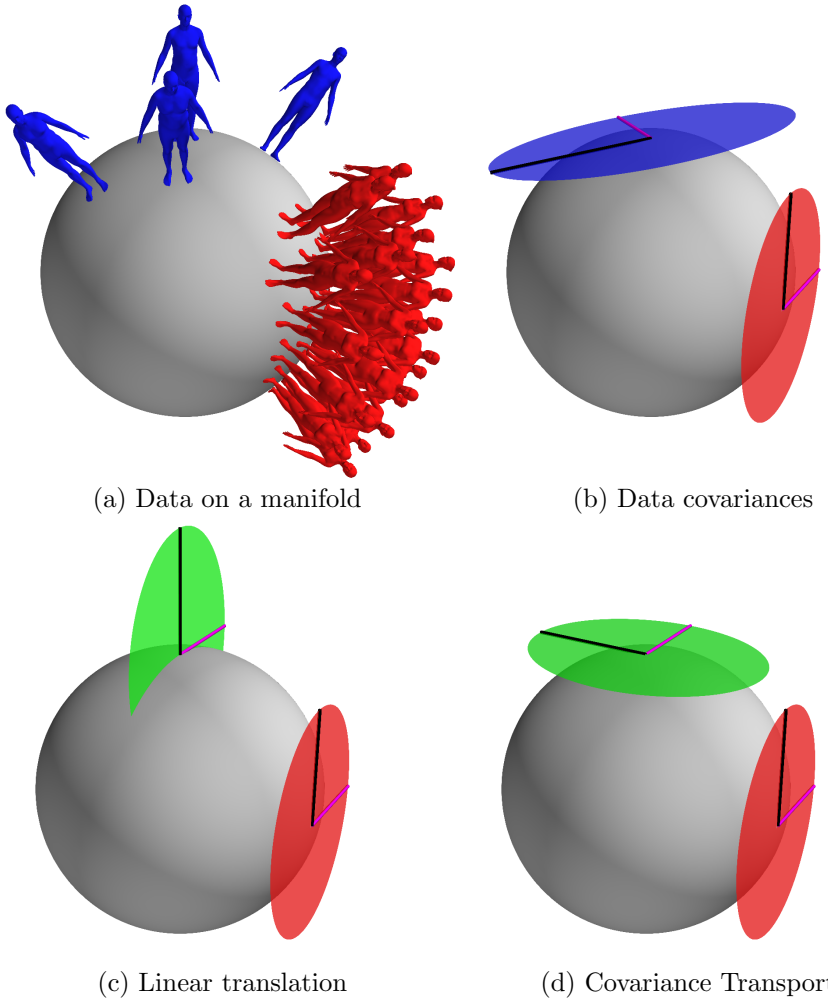
Remark 1.2.3. (Consistent representation *vs*. consistent estimator) Note that our use of the term *consistency* is different from the notion of consistency in statistical inference[11].

1.3 Transfer Learning on Manifolds

A common problem in many scientific disciplines involves learning a statistical model from a small number of observations. Such small-sample scenarios occur when data collection is time consuming, prohibitively expensive, or when the phenomenon of interest rarely occurs. For $\mathbb{R}^n$-valued data, transfer learning can utilize a second, closely related, class (if it exists) for which more data are available to improve statistical learning of the small-sample class. This works well when both classes have similar models or when the large-sample class contains variations that are related to those of the small-sample one. Unfortunately, despite their ubiquity, this transfer learning approach has yet to be generalized to nonlinear manifolds.

In Chapter 5 we consider the more general case where the data lie on a *known Riemannian manifold* instead of $\mathbb{R}^n$. In particular, we address the following problem. Fix a manifold M. Given two M-valued datasets, one small ($\mathcal{D}_{\mathrm{S}}$) and one large ($\mathcal{D}_{\mathrm{L}}$), we wish to

[11] Loosely speaking, there consistency means that as the number of examples tends to infinity, a consistent estimator is one that converges (in some sense) to the true value.



(a) Data on a manifold (b) Data covariances

(c) Linear translation (d) Covariance Transport

Figure 1.5: Covariance Transport. On nonlinear manifolds, statistics of the Many (red) are transported to improve a statistical model of the Few (blue). See text for details.

learn a model from the smaller dataset, while leveraging statistical information from the larger one. To fix ideas we use a manifold of 3D body shape deformations from Chapter 4 as our running example; as illustrated in Fig. 1.5a, every point on M represents an entire human shape[12]. For illustration purposes, assume that examples of female body shapes (in red) are much more plentiful than those of males (in blue). We stress that this particular manifold is just an example of a nonlinear space; *our approach is applicable to a very broad class of manifolds*, including those that are not matrix Lie groups (to be defined in section 2.3). Later on we will see that the approach applies to both additional manifolds and different classes on this particular manifold.

[12]For visualization purposes, the manifold here is shown as a 2D manifold embedded in $\mathbb{R}^3$; its true dimension is much higher. See Chapter 4.

Imagine we want to model the body shape variation of men while leveraging the model of the shape variation of women. Models learned from scarce data are prone to over-fitting and poor generalization. We suspect, however, that some aspect of shape variation among women may apply to men as well and that we can use this to learn a better model for male shape variation.

The key contribution in Chapter 5 is generalizing transfer learning from $\mathbb{R}^n$ to known nonlinear manifolds by formulating a solution based on any *metric parallel transport*; namely, parallel transport that respects the Riemannian metric of the manifold (all these notions will be defined in Chapter 2). This allows us to move a statistical model learned in one region of M to another where it can be used to improve the model of interest. In practice this means we can estimate the covariance of one class and move it to the region of M where the data of interest reside.

Referring again to Fig. 1.5, we first compute, for $\mathcal{D}_{\mathrm{L}}$, the (intrinsic) mean and a covariance expressed in a tangent space[13] whose point of tangency is this mean (Fig. 1.5b, red). We would like to move this covariance to the mean of $\mathcal{D}_{\mathrm{S}}$ such that it can be used to improve the covariance estimation of $\mathcal{D}_{\mathrm{S}}$ (Fig. 1.5b, blue). Clearly we cannot simply translate the covariance as it would no longer be tangent to M (Fig. 1.5c, green) yielding an undefined result. Instead we take the geometry of M into account and move the covariance between the tangent spaces in a way that *preserves the structure of the covariance while adapting to the structure of* M (Fig. 1.5d, green).

When moved along a smooth curve on a nonlinear manifold, a geometric object, such as a covariance matrix, is deformed to adapt to the local metric and curvature. We prove that when *a covariance is moved via metric parallel transport, its statistical meaning is preserved*. Not only is this theoretically sound, in practice it can work remarkably well; *e.g.*, Fig. 5.10c illustrates how well the shape of one gender is transported to another. Once the covariance has been moved, it can be used to improve the covariance estimation of $\mathcal{D}_{\mathrm{S}}$.

[13] These notions of mean and covariance will be explained in Chapter 2.

This can be done by "fusing" the covariances or using the transported covariance of $\mathcal{D}_{\mathrm{L}}$ to regularize that of $\mathcal{D}_{\mathrm{S}}$. We call this framework *Covariance Transport* (CT).

The framework allows us to solve seemingly difficult problems with surprising results; *e.g.*, we find that observations of *normal-weight* people help in modeling people with *high body mass index* (BMI). This is important because body scans of high-BMI people tend to be scarce. Also, image covariance descriptors of many faces in many poses can be similarly adapted to model the face of a new person captured in only a few poses. Our method can form the foundation for many problems in computer vision, pattern recognition and statistical modeling.

1.4 A Sketch of this Thesis

To summarize, the main contributions of this thesis are as follows:

1. The *Contour Person* model: a new learned 2D articulated model – more expressive than traditional pictorial structures models – of deformable human shape that factors body-shape, pose, and camera variations (Chapter 3; see also Freifeld *et al.* [43]).

2. *Lie Shapes*: a new representation of 3D shape deformations which in turn gives rise to statistical deformable-template models that are better than the current state-of-the-art (Chapter 4; see also Freifeld and Black [41]).

3. *Covariance Transport*: generalizing certain transfer learning techniques – traditionally limited to $\mathbb{R}^n$-valued data – from $\mathbb{R}^n$ to a Riemannian setting (Chapter 5; see also Freifeld *et al.* [42]).

But first, some preliminaries are needed. Chapter 2 covers the mathematical background that is required for the understanding of its subsequent chapters.

Chapter 2

Mathematical Background

This chapter contains the mathematical background required for understanding its subsequent chapters, especially Chapter 4 and Chapter 5.

Note that to understand most of Chapter 3, it is not required to read the current chapter. Small portions of Chapter 3, however, may be better understood after having read the first three sections of the current chapter.

In Section 2.1 we expand the discussion regarding several nonlinear spaces mentioned earlier. In Section 2.2 we provide intuition for the concept of manifolds as well as an illustrated tutorial for basic manifold-related notions using the sphere as a running example. In Section 2.3 we present a class of manifolds, called matrix Lie groups, that is of special importance in this work.

In Section 2.4 we provide a gentle introduction to manifold theory. For reasons to be elaborated on later, our treatment follows the abstract (and more general) approach to manifold theory rather than presenting manifolds as special subsets of $\mathbb{R}^n$. For now, we briefly state that the primary reason for doing so is that, while the abstract approach requires more mathematical sophistication, it makes working with manifolds *easier*. Having

said that, Section 2.4.2.3 is more abstract than the rest of the chapter and may be skipped.

In Section 2.5 we touch upon statistics on manifolds. Along the way, we briefly cover how several standard statistical concepts from $\mathbb{R}^n$, such as the sample mean and PCA, are generalized to a manifold setting.

In Section 2.6 we discuss statistics on matrix Lie groups. As a particular class of manifolds, the discussion from Section 2.5 applies transparently to these groups too, with many of the associated equations given in simple closed form. There are, however, certain additional techniques that can be used on matrix Lie groups due to their special structure. We will refer to these techniques as *Lie-algebraic*. Importantly, there are practical differences between the resulting models. We illustrate this using an example with models of image deformations (such as those used for optical flow in [93, 94, 147]) where we compare a Riemannian model with a Lie-algebraic one; to the best of our knowledge this is the first time such a direct comparison is made in computer vision applications. Thus, even in the presence of a matrix Lie group, one may want to consider the pure Riemannian approach. In particular, for the new tool for statistics on manifolds we will introduce in Chapter 5, the Riemannian approach will be more natural as well as easier to use.

2.1 Examples for Nonlinear Spaces

In Section 1.2.1 we mentioned several nonlinear spaces that often arise in applications. The current section provides additional details.

2.1.1 Spheres

As a warm-up, let us consider two points on the unit circle.

Example 2.1.1 (The unit circle is not linear nor convex)**.** Let $x_1, x_2 \in S^1$, where

$$S^1 \stackrel{\text{def}}{=} \left\{(x, y) : (x, y) \in \mathbb{R}^2, x^2 + y^2 = 1\right\} . \tag{2.1}$$

It is easy to see that $x_1 + x_2$ is not in S^1 and thus S^1 cannot be a linear space. Alternatively, the nonlinearity can be shown by noting that if $c \in \mathbb{R} \setminus \{-1, 1\}$, then $cx_1 \notin S^1$. Moreover, not only is S^1 nonlinear, it is not even convex: if $x_1 \neq x_2$, then $(x_1 + x_2)/2 \notin S^1$.

The generalization to higher dimensions is straightforward.

Example 2.1.2 (The unit sphere is neither linear nor convex)**.** The unit sphere in $\mathbb{R}^n$,

$$S^{n-1} \stackrel{\text{def}}{=} \left\{(x_1, x_2, \ldots, x_n) : (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n, \sum_{i=1}^{n} x_i^2 = 1\right\} , \tag{2.2}$$

is not a linear space; it is not even a convex set. The arguments are identical to the ones from Example 2.1.1.

Remark 2.1.1 (*n vs. N*; subscript notation). In this work, the number of data points is typically denoted by N while the standard Euclidean space is denoted by $\mathbb{R}^n$ (and not by, say, $\mathbb{R}^N$). As both n and N are usually greater than one, we need a notation for not only the components of a given data point but also its index with respect to the entire set of data points. In the context of statistics, using superscripts for either of these cases is cumbersome, especially when powers are involved[1]. Consequently, *we use subscripts for both*. Usually this should lead to no confusion; whenever the particular case of use is not immediately evident from the text, it will be stated explicitly.

2.1.2 Constraints on Matrix-Valued Data

Several ubiquitous constraints on matrix-valued data are determinant-related.

[1] In the context of differential geometry, Einstein summation, a notational tool that does employ superscripts, provides an elegant way to avoid lengthy expressions; however, the likely readership of this work might not be familiar with such notation.

Example 2.1.3 (Determinant constraints)**.** Let n be a fixed positive integer. The following three sets of matrices are nonlinear spaces:

$$\text{(invertible matrices)} \qquad \left\{Q : Q \in \mathbb{R}^{n\times n},\ \det Q \neq 0\right\}; \tag{2.3}$$

$$\text{(positive determinant)} \qquad \left\{Q : Q \in \mathbb{R}^{n\times n},\ \det Q > 0\right\}; \tag{2.4}$$

$$\text{(volume-preserving)} \qquad \left\{Q : Q \in \mathbb{R}^{n\times n},\ \det Q = 1\right\}. \tag{2.5}$$

Another important case is related to orthogonality.

Example 2.1.4. The space of orthogonal matrices,

$$\left\{Q : Q \in \mathbb{R}^{n\times n},\ Q^TQ = QQ^T = I_{n\times n}\right\}, \tag{2.6}$$

is nonlinear.

When the orthogonality constraint is added to the volume-preserving constraint, we get another ubiquitous space; the space of rotation matrices.

Example 2.1.5 (Rotation matrices)**.** The space of $n \times n$ rotation matrices,

$$\left\{Q : Q \in \mathbb{R}^{n\times n},\ Q^TQ = QQ^T = I_{n\times n},\ \det Q = 1\right\}, \tag{2.7}$$

is nonlinear.

Remark 2.1.2. The determinant constraint in Eqn. (2.7) is not superfluous: while $Q^TQ = QQ^T = I_{n\times n}$ implies that $|\det Q|$ is equal to 1, it does not rule out the case of $\det Q = -1$.

Example 2.1.6 (Symmetric positive-definite matrices)**.** The set of SPD matrices,

$$\left\{Q : Q \in \mathbb{R}^{n\times n},\ Q = Q^T,\ a^TQa > 0\ \forall \text{ non-zero } a \in \mathbb{R}^n\right\}, \tag{2.8}$$

is nonlinear.

To see that all of the aforementioned spaces are nonlinear, take Q to be the $n \times n$ identity matrix and observe that $Q - Q$ is not in any of these sets.

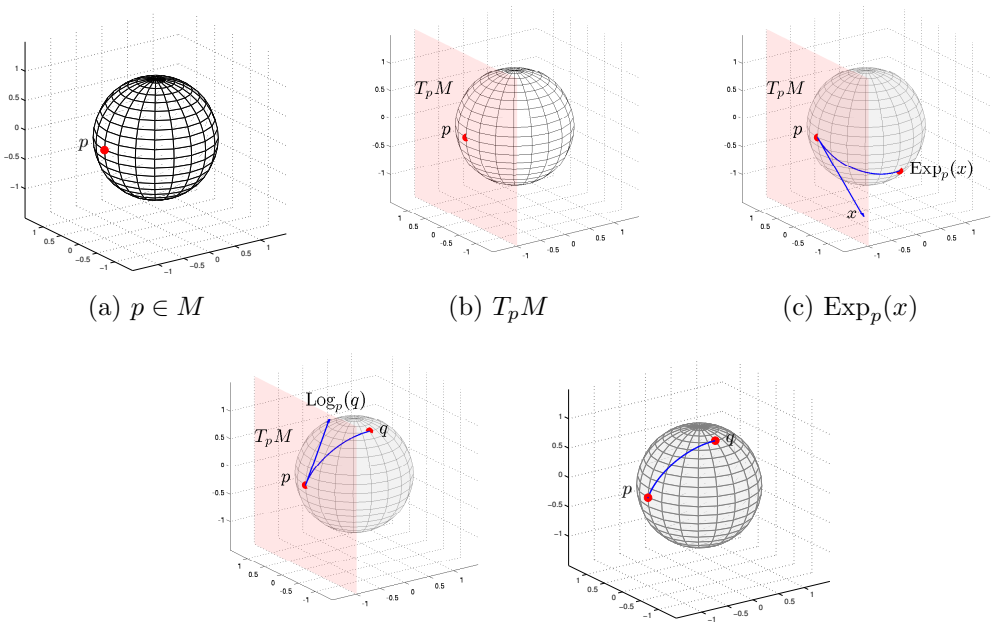
2.2 Manifolds: Intuition

In Section 2.1 we saw several examples for nonlinear spaces and in Chapters 3 and 4 we will see that the spaces of 2D and 3D shape deformations are nonlinear. All these spaces, however, are not merely nonlinear; they also have a *manifold* structure. The purpose of the current section is to provide the reader with some intuition in regard to manifolds. Thus, our treatment here will be quite informal. While the discussion will be general, we will use S^{n-1}, the unit sphere in $\mathbb{R}^n$, as a running example. Most of the formulas in this section that are related to the sphere are based on [1].

A *finite-dimensional (topological) manifold* M of dimension n is a nice[2] space that in some local sense resembles the Euclidean space $\mathbb{R}^n$. Usually M is nonlinear, but this need not be the case; *e.g.*, $\mathbb{R}^n$ itself is a finite-dimensional manifold. Formally, and as we will see later on, an ambient space is not required; however, for gaining intuition, readers unfamiliar with manifold theory may regard M as a curved subset of a Euclidean space. For example, the sets in the examples from the previous section are nonlinear subsets of $\mathbb{R}^{n \times n}$.

A *finite-dimensional smooth manifold* is a finite-dimensional manifold that is smooth in some sense. Admittedly, even as an informal definition this tautology is not very satisfying. But for now, we ask the reader to be content with knowing that the unspecified sense above refers to our ability to "do calculus" on the manifold. Henceforth, all manifolds in this work are assumed to be smooth and of finite dimension.

[2] When mathematicians say that some mathematical object is nice, they mean that it satisfies several properties that are useful in the given context. This shorthand frees them from having to specify or explain the actual properties – which might be too long, distracting, or even impossible to do. Another similar term used in that context is "well-behaved". We prefer the term "nice" as it is shorter. And nicer.



(a) $p \in M$ (b) T_pM (c) $\mathrm{Exp}_p(x)$

(d) $\mathrm{Log}_p(q)$ (e) The geodesic from p to q

Figure 2.1: An illustration of Riemannian geometry.

Example 2.2.1 (The sphere: characterization of elements in M)**.** For the sphere, the characterization of elements in M is given by

$$M = S^{n-1} \stackrel{\text{def}}{=} \{p : p \in \mathbb{R}^n,\ \|p\|_{\ell^2} = 1\}\ . \tag{2.9}$$

The dimension of M is $n-1$; e.g., S^2, the unit sphere in $\mathbb{R}^3$, is two-dimensional. See Fig. 2.1a.

While M is usually nonlinear, we can associate a tangent space, denoted by T_pM, to every point $p \in M$. T_pM is a vector space whose dimension is the same as that of M. The origin of T_pM is at p. If M is embedded in some Euclidean space, we may think of T_pM as an affine subspace such that: 1) it touches M at p; 2) at least locally, M lies completely on one of side of it. See Fig. 2.2. Elements of T_pM are called *tangent vectors*.

Example 2.2.2 (The sphere: characterization of elements in T_pM)**.** Let p be a point on

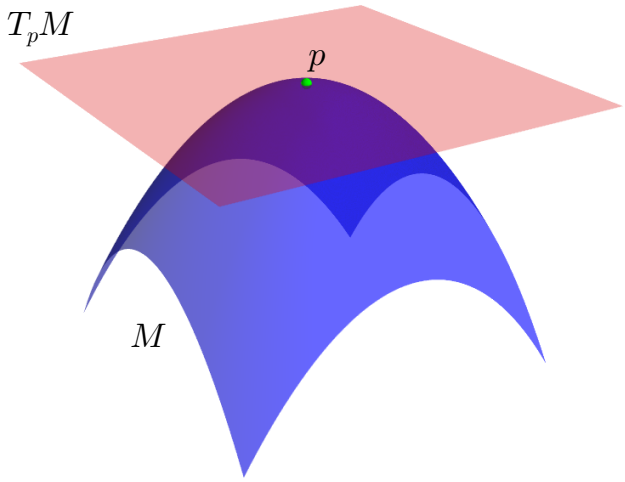


Figure 2.2: A manifold M and a tangent space, T_pM, whose point of tangency is p.

the sphere. The tangent space T_pM is fully characterized by

$$T_pM = \left\{x : x^T p = 0 \ , \ x \in \mathbb{R}^n\right\} \ . \tag{2.10}$$

See Fig. 2.1b.

The tangent space can be equipped with an *inner-product* (see Definition A.3.4, page 205):

$$\langle \cdot, \cdot \rangle_p : T_pM \times T_pM \to \mathbb{R} \ . \tag{2.11}$$

Two tangent vectors are called *orthogonal* if their inner-product is 0. We use the inner-product to induce a *norm* on T_pM, rendering T_pM a *normed vector space* (see Definition A.3.2, page 204):

$$\|\cdot\|_p : T_pM \to \mathbb{R}^+, \ \|\cdot\|_p : x \mapsto \sqrt{\langle x, x \rangle} \ . \tag{2.12}$$

The norm of a vector can be interpreted as its length. Together, the inner-product and the norm enable us to define an angle (see Definition A.3.5, page 205) between a pair of tangent vectors.

Note the p subscript in $\langle \cdot, \cdot \rangle_p$ and $\|\cdot\|_p$. In general, these maps depend on p because 1) they may behave differently as p varies; 2) their domains ($T_pM \times T_pM$ or T_pM) depend on p.

If the collection of inner-products, $\left\{\langle \cdot, \cdot \rangle_p\right\}_{p \in M}$, is sufficiently nice, then it is called a *Riemannian metric* and M is said to be a *Riemannian manifold* [27, 88]. A Riemannian metric enables us, among other things, to define lengths of curves in M. Consider two manifold points p and q and all of the nice curves (in M) that start at p and end at q. The curves that are local minimizers of the function that maps a curve to its length are called *geodesic curves* or simply *geodesics*. Let $d(p, q)$ denote the length of the shortest geodesic between p and q. We think of $d(p, q)$ as the distance between these two points. This distance is called the *geodesic distance*. By a slight abuse of terminology, we will often write "*the* geodesic" between p and q, and by this will mean the shortest geodesic between them. Henceforth, whenever we refer to a distance between a pair of points in M, as opposed to a distance between two tangent vectors, we mean geodesic distance.

The term *metric*, as used in the terms *metric space* (see Definition A.3.1, page 204) and *Riemannian metric*, is unfortunately overloaded. There is, however, a relation between the two meanings. A Riemannian metric can *define* a metric on M in the sense of a metric space: if M is nice enough, then the pair (M, d) is a metric space. As usual, in such cases we usually omit d and just say that M is a metric space. As M is (usually) nonlinear, there is no point in talking about M as a normed space since by definition, normed spaces are always linear. In contrast, it does make sense to talk about T_pM as a normed space.

If M is embedded in some Euclidean space, then one possible Riemannian metric on M is the one induced by the inner-product of the ambient space. This is arguably the simplest choice of a Riemannian metric. Being a Riemannian metric, it induces geodesic distances between points in M. Recall that the Euclidean ambient space has its own distance function, denoted by $d(\cdot, \cdot)_{\text{Ext}}$. Usually, $d(p, q) \neq d(p, q)_{\text{Ext}}$. In other words, even with this simple Riemannian metric, the distance between points on a manifold is different

from the distance between them where they are regarded as points in the ambient space.

Example 2.2.3 (The sphere: the Riemannian metric induced by the inner-product of the ambient Space)**.** In this case, the inner-product of the ambient space, $\langle \cdot, \cdot \rangle_{Ext}$, is the dot product in $\mathbb{R}^n$:

$$\langle \cdot, \cdot \rangle_{Ext} : (x, y) \mapsto x \cdot y = x^T y \ . \tag{2.13}$$

In other words, if x and y are in T_pM, then their inner-product is given by

$$\begin{aligned} \langle x, y \rangle_p &= \langle (x + p) - p, (y + p) - p \rangle_{Ext} \\ &= \langle x, y \rangle_{Ext} \\ &= x^T y \ , \end{aligned} \tag{2.14}$$

where in the first equality we have taken into account the fact that the origin of T_pM is $p \in \mathbb{R}^n$.

Since manifolds are usually nonlinear, they are more complicated objects than $\mathbb{R}^n$. A standard trick that often saves the day is to make use of the existence of tangent spaces, the latter being linear (hence simpler). We already alluded to that: the geodesic distance was implied by a collection of inner-products, each one of which is defined in a tangent space. In particular, the existence of tangent spaces is what makes Riemannian manifolds simple enough to allow for tractable statistics – as we will see later on. To utilize the tangent spaces, we need mappings back and forth between between T_pM and M. Note that these mappings depend on p. There are two kinds of such pairs of mappings of interest to us. The first kind is the pair of *Riemannian exponential map* and *Riemannian logarithm*:

$$\text{(Riemannian exponential map)} \qquad \text{Exp}_p : T_pM \to M \ ; \tag{2.15}$$

$$\text{(Riemannian logarithm)} \qquad \text{Log}_p : M \to T_pM \ . \tag{2.16}$$

These maps are tied to the concepts of geodesic distances and geodesic paths. The second type of interest to us is related to Lie groups. We will return to it later this chapter.

Example 2.2.4 (The sphere: the Riemannian exponential and logarithm maps)**.** The

exponential map, $\mathrm{Exp}_p : T_pM \to M$ is given by

$$\mathrm{Exp}_p(x) = p\cos\left(\|x\|\right) + \frac{x}{\|x\|}\sin(\|x\|)\ . \tag{2.17}$$

See Fig. 2.1c. Let $x \in T_pM$, $q = \mathrm{Exp}_p(x)$, $u = \frac{x}{\|x\|}$, and $m = \|x\|$. A simple calculation, using the fact that $u \in M \cap T_pM$, shows that

$$q^Tq = \sin(2m)p^Tu + \cos^2(m)\|p\|^2 + \sin^2(m)\|u\|^2 = \cos^2(m) + \sin^2(m) = 1 \tag{2.18}$$

and so q is indeed in M. Similarly, let p and q be in M. The logarithm map $\mathrm{Log}_p : M \to T_pM$ is given by

$$\mathrm{Log}_p(q) = (q - p\cos\theta)\frac{\theta}{\sin\theta} \tag{2.19}$$

where $\theta = \arccos(p^Tq)$. See Fig. 2.1d.

Example 2.2.5 (The sphere: geodesic curves and geodesic distances)**.** Let $p, q \in M$. Set $x = \mathrm{Log}_p(q)$, $u = x/\|x\|$ and $m = \|x\|$. The geodesic curve between p and q is given by

$$c(t) = p\cos(mt) + u\sin(mt)\ . \tag{2.20}$$

A similar calculation to the one before shows that $c(t)^Tc(t) = 1$ for all t and thus $c(t) \in M$. See Fig. 2.1e. Finally, the geodesic distance between p and q is given by

$$d(p,q) = \arccos(p^Tq) \neq d(p,q)_{\mathrm{Ext}}\ . \tag{2.21}$$

2.3 Matrix Lie groups

2.3.1 Definition and Basic Properties

Definition 2.3.1 (Matrix Lie groups)**.** Let n be a fixed positive integer. A *matrix Lie group* is a set G of $n \times n$ matrices together with the binary operation of matrix product

on G (that is, the domain is $G \times G$) such that:

(G1) $I_{n\times n} \in G$;

(G2) $A, B \in G \Rightarrow AB \in G$;

(G3) $A \in G \Rightarrow A$ is an invertible matrix, $A^{-1} \in G$;

Matrix Lie groups are also called *matrix groups*, the terms being identical. It is possible to use a similar definition for matrix Lie groups whose elements take complex values; in this work, however, the discussion is restricted to real-valued matrix Lie groups.

Let G satisfy the conditions in Definition 2.3.1. Basic linear algebra shows that:

(G4) $A \in G \Rightarrow AI_{n\times n} = I_{n\times n}A = A$

(G5) $A, B, C \in G \Rightarrow (AB)C = A(BC) = ABC$

(G6) $A \in G \Rightarrow AA^{-1} = A^{-1}A = I_{n\times n}$.

What (G1) through (G6) imply is that the set G, together with the binary operation of matrix product, is a group and that $I_{n\times n}$ is the identity element of the group. So unsurprisingly, every matrix group is a group. When n is understood from the context, we will sometimes denote the identity matrix by I instead of $I_{n\times n}$.

If G is a set of $n \times n$ matrices, which may or may not form a matrix group once taken together with matrix product, it may be the case that there is some *other* binary operation, denoted by, say, $\diamond : G \times G \to G$, such that the pair $(G, \diamond)$ forms a group. This group, however, is not a matrix group, although it is a "group of matrices". For example, if G is $\mathbb{R}^{n\times n}$ and $\diamond$ is matrix addition, than the group $(G, \diamond)$ is not a matrix group[3]. Henceforth, by a slight abuse of notation, we will write expressions such as "G is a matrix Lie group",

[3]Likewise, the groups of matrices defined by Alexa in [2] are not matrix Lie groups according to the definition we use here; they are, however, Lie groups.

with the convention that we will mean that the *set* G, together with matrix product, is a matrix Lie group. Similarly, we will write "the matrix Lie group G is given by the following set", meaning that the set, together with matrix product, defines the matrix Lie group of interest.

Definition 2.3.2 (The difference between two elements in a matrix Lie group)**.** Let G be a matrix Lie group, and let A and B be in G. The *group difference* of A and B is given by $A^{-1}B$; it is not symmetric.

Note that (G2) and (G3) imply the following:

(G7) $A, B \in G \Rightarrow A^{-1}B \in G$.

Properties (G2), (G3), and (G7), are referred to as the group (algebraic) *closure* under its operations of *composition*, *inversion*, and *difference*, respectively. Suppose our data can be represented as elements of a matrix Lie group. These closure properties imply that the representation is consistent.

Definition 2.3.3 (Abelian matrix Lie group)**.** If $AB = BA$ for any A and B in a matrix Lie group G, then G is called *Abelian*.

Finally, the class of matrix Lie groups is contained in the class of (finite-dimensional) *Lie groups* [89]: every matrix Lie group is a Lie group while the converse is false. Matrix Lie groups are simpler to work with (and define) than the more general case of Lie groups. All Lie groups in this work are matrix Lie groups, which considerably simplifies the discussion. We remark, however, that the tool we will present in Chapter 5 is applicable to a large class of Riemannian manifolds, regardless of whether they are Lie groups or not.

2.3.2 Several Important Matrix Lie Groups

We start with the most general matrix Lie group.

Definition 2.3.4 (The general linear group of order n)**.** The *general linear group of order* n, denoted by GL(n), is given by Eqn. (2.3).

We call it the most general as, by (G3), we see that *every* matrix Lie group of $n \times n$ matrices is a *matrix Lie subgroup*[4] of GL(n),

Note that GL(n) is not a connected space (see Definition A.1.9, page 203). To see that, pick a matrix A in GL(n) with a positive determinant and a matrix B in GL(n) with a negative determinant. Then, set $p = A$ and $q = B$. As $\det : \mathbb{R}^{n\times n} \to \mathbb{R}$ is continuous (see Example A.1.4, page 201), any continuous curve between p and q must pass through a matrix of zero determinant; *i.e.*, the curve must make an excursion outside of GL(n). In fact, it can be shown that GL(n) has exactly two connected components.

Definition 2.3.5 (The identity component)**.** The *identity component* of a matrix Lie group is one that contains $I_{n\times n}$. The identity component of a matrix Lie group is always a matrix Lie group by itself.

Definition 2.3.6 ($\mathrm{GL}^+(\mathrm{n})$)**.** The *identity component of* GL(n), denoted by $\mathrm{GL}^+(\mathrm{n})$, is given by Eqn. (2.4).

The following matrix Lie groups are all of high importance for this thesis.

Definition 2.3.7 (US(n))**.** The *uniform scale group*, denoted by US(n), is given by

$$\left\{Q : Q = SI_{n\times n},\ S \in \mathbb{R}^+\right\}. \tag{2.22}$$

Specifically, we will use our own notation for the scalar case (where $\mathrm{GL}^+(\mathrm{n})$ coincides with US(n)):

Definition 2.3.8 (G_S)**.** The group of scales, denoted by G_S, is given by $\mathbb{R}^+$ together with the operation of scalar multiplication.

[4]The relation between a matrix Lie subgroup and a matrix Lie group is analogous to the relation between a subgroup and a group.

Definition 2.3.9 (O(n))**.** The *orthogonal group of degree* n, denoted by O(n), is given by Eqn. (2.6).

Definition 2.3.10 (SO(n))**.** The *special orthogonal group of degree* n, also known as the rotation group and denoted by SO(n), is given by Eqn. (2.7).

In particular, we single out the case of $n = 3$.

Definition 2.3.11 (SO(3))**.** The *special orthogonal group of degree* 3, also known as the rotation group (of order 3), is denoted by SO(3).

Definition 2.3.12 (SPD)**.** The SPD group, is given by Eqn. (2.8).

2.3.3 Direct Products of Matrix Lie Groups

Let G_i be a matrix Lie subgroup of $\mathrm{GL(n_i)}$, $i = 1, 2, \ldots, k$. The standard group direct product of $\{G_i\}_{i=1}^{k}$, denoted by $G_1 \times G_2 \times \ldots \times G_k$, is not technically a matrix Lie group, as its elements are k-tuples of matrices (such as $(g_1, g_2, \ldots, g_k)$) rather than matrices. It is, however, easy to identify such a direct product with a matrix Lie subgroup of $\mathrm{GL}(n_1 + n_2 + \ldots + n_k)$ using the correspondence between a $k-$tuple $(g_1, g_2, \ldots, g_k)$ and a block-diagonal matrix:

$$(g_1, g_2, \ldots, g_k) \leftrightarrow \begin{pmatrix} g_1 & & & 0 \\ & g_2 & & \\ & & \ddots & \\ 0 & & & g_k \end{pmatrix} \in \mathrm{GL}(n_1 + n_2 + \ldots + n_k)\ . \tag{2.23}$$

Lastly, if the G_i's are k copies of the same matrix Lie group G, then we denote their direct product by G^k.

2.3.4 Matrix Lie Groups as Smooth Manifolds

Note that no matrix Lie group is a linear space[5]: Merely observe that if A is any invertible matrix (in fact, in GL(n)), then $A - A = 0_{n\times n}$ is not invertible. Thus, while a matrix Lie group is closed under the operations mentioned earlier, it is not closed under linear combinations. In particular, it does not make sense to talk about linear subspaces of such a group. While nonlinear, *every matrix Lie group is a finite-dimensional smooth manifold.* Thus, a matrix Lie group is a group on which one can "do calculus". Alternatively, a matrix Lie group is a smooth manifold with a group structure. Groups are usually denoted by G, while manifolds are usually denoted by M. Since a matrix Lie group is both a group and a manifold, both notations may be used.

2.3.4.1 Dimensionality

Every matrix Lie group of $n \times n$ matrices is a nonlinear subset of $\mathbb{R}^{n\times n}$, the latter being an n^2-dimensional space. Since such a group is also a finite-dimensional manifold, it has its own dimension. To avoid a notational clash, we will use D to denote the dimension of the group; note $D \leq n^2$.

The dimension of both GL(n) and GL^+(n), for example, is $D = n^2$. Note this is the same dimension as that of $\mathbb{R}^{n\times n}$, in spite of the fact that GL^+(n) is a proper subset of GL(n), which in turn is a proper subset of $\mathbb{R}^{n\times n}$. Our point here is that once we deal with manifolds, even if two spaces have the same dimension, one can still be strictly larger than the other. Another example is SO(3), whose dimension is 3.

[5]It may seem odd that an object called the general *linear* group is not linear, but the linearity in the name refers to the fact that every element A of the group is affiliated with a linear map $\mathbb{R}^n \to \mathbb{R}^n$, $x \mapsto Ax$.

2.3.4.2 Smoothness and Tangent Spaces

One important implication of the fact that a matrix Lie group G is a smooth manifold is that it is possible to define smooth curves on G. Note that if $c : J \to G$ is a smooth curve (to be defined later) from some open interval J in $\mathbb{R}$ into G, then $c(t)$ is itself a matrix that belongs to G. If we differentiate c with respect to t (*i.e.*, differentiate each entry of the matrix $c(t)$ with respect to t), then we will get a new curve, $\dot{c}$. However, this curve is a map $\dot{c} : J \to \mathbb{R}^{n\times n}$, not $\dot{c} : J \to G$. Regarding $\mathbb{R}^{n\times n}$ as $\mathbb{R}^{n^2}$ (which is hard to visualize: even for 2-by-2 matrices this is already a 4D space), we can imagine $\dot{c}(t)$ as an n^2-dimensional vector attached to the curve at the n^2-dimensional point $c(t)$. In terms of standard differential geometry of curves and surfaces in Euclidean spaces, $\dot{c}(t)$ is a *tangent vector to the curve*. In fact, it is also a tangent vector to the manifold (again, in those terms, where we refer to a manifold as a surface in a Euclidean space) at the point p.

For a given point p, and a given open interval J that contains 0, consider all smooth curves $c : J \to G$ that pass though p and satisfy $c(0) = p$. Let us denote this class of curves by $C_{J,p}$. The tangent vectors, at p, are given by the set of *distinct*[6] elements

$$\{\dot{c}(0) : \ c \in C_{J,p}\} \ . \tag{2.24}$$

It can be shown that the particular choice of J is immaterial. It turns out that the tangent vectors form a subspace of $\mathbb{R}^{n\times n}$. This subspace is denoted by T_pG (had we used M instead of G we would have written T_pM) and its dimension is identical to that of the matrix Lie group G:

$$\dim G = \dim T_pG \ . \tag{2.25}$$

For example, for $G = \mathrm{GL(n)}$, the dimension of the tangent space is always n^2 (so T_pG is a perfect copy of $\mathbb{R}^{n\times n}$, both being Euclidean spaces of the same dimension) while for $G = \mathrm{SO(3)}$ the dimension of the tangent space is always 3. Let G be a D-dimensional

[6] Many different curves in $C_{J,p}$ can have the same derivative at t equals zero. Thus, their derivatives form equivalent classes. The set-theoretic notation means we pick one representative from each class.

matrix Lie group of $n \times n$ matrices. Since at every point p in G we can attach a tangent space which is a copy of $\mathbb{R}^D$, all these tangent spaces are the same in some sense. Having said that, the tangent space at the identify (*i.e.*, $p = I$) is rather special as we shall see in Section 2.3.6.

Finally, in spite of the fact that an element of T_pG is always an $n \times n$ matrix, it is sometimes useful to regard it a vector in $\mathbb{R}^{n \times n}$. For this we use the following notation:

$$\text{vec}\,(\cdot) : T_pG \to \mathbb{R}^{n^2} \qquad ; \qquad \text{mat}\,(\cdot) : \mathbb{R}^{n^2} \to T_pG\,. \tag{2.26}$$

In Eqn. (2.26), vec stands for concatenating the matrix columns in a single long column, while mat is the inverse of *vec*.

2.3.4.3 Characterization of Elements of a Tangent Space

Let $c \in C_{J,p}$. Define another smooth curve by $c_I : J \to G$, $c_I : t \mapsto p^{-1}c(t)$. We think of this operation as *left-translation* (of a curve) by p^{-1}. It follows that $c_I(0) = I$ (as $c(0) = p$) and that $c_I \in C_{J,I}$. Likewise, for every c in $C_{J,I}$ there is an element in $C_{J,p}$, denoted by c_p, such that $c_p : J \to G$, $c_p : t \mapsto pc(t)$. Naturally, we think of this as left-translation (of a curve) by p. These relations establish an obvious bijection between $C_{J,I}$ and $C_{J,p}$.

Let $c \in C_{J,I}$. Since

$$\frac{d}{dt}(pc(t))\bigg|_{t=0} = p\frac{d}{dt}c(t)\bigg|_{t=0}\,, \tag{2.27}$$

it follows that the tangent vectors that comprise T_pG are given by

$$T_pG = \{px :\ x \in T_IG\}\,. \tag{2.28}$$

Consequently, if we know how to characterize T_IG, then we know how to characterize T_pG. Similarly, $T_IG = \{p^{-1}x :\ x \in T_pG\}$.

In Eqn. (2.28), p is in G, $x \in T_I G$, and their matrix product px is in $T_p G$. We think of the map $T_I G \to T_p G$, $x \mapsto px$ as left-translation (of a tangent vector) by p. Right-translation by p is defined similarly by $T_I G \to T_p G$, $x \mapsto xp$.

More generally, if p and g are in G then *one way* to map the tangent vectors in $T_p G$ onto $T_q G$ is by left-multiplication of a tangent vector (which is in fact a matrix!) from one tangent space by qp^{-1} to produce another tangent vector (again, a matrix) in the second. Again, this is called left-translation, with right-translation defined in a similar way.

Back to characterization. Sometimes it is easy to characterize $T_I G$.

Example 2.3.1. Let $G = \mathrm{O(n)}$. We first characterize the elements of $T_I G$. Let $c \in C_{J,I}$. Then,

$$c(t)^T c(t) = I \qquad \text{(by Eqn. (2.7))} \tag{2.29}$$

$$\dot{c}(t)^T c(t) + c(t)^T \dot{c}(t) = 0 \qquad \text{(differentiate with respect to } t\text{)} \tag{2.30}$$

$$\dot{c}(0)^T c(0) + c(0)^T \dot{c}(0) = 0 \qquad \text{(set } t = 0\text{)} \tag{2.31}$$

$$\dot{c}(0)^T = -\dot{c}(0) \qquad (c(0) = I)\ . \tag{2.32}$$

The conclusion is that every tangent vector to O(n), at I, is a skew-symmetric matrix. It is also possible to show that the space of skew-symmetric matrices has the same dimensionality as O(n). By appealing to a standard dimensionality argument, it follows that the converse is also true: every skew-symmetric matrix is a tangent vector. Note we would have gotten the exact same result had we worked with SO(n) rather than O(n). Consequently,

$$T_I \mathrm{SO(n)} = T_I \mathrm{O(n)} = \left\{A : A = -A^T\right\} \subset \mathbb{R}^{n \times n}\ . \tag{2.33}$$

If p is a point in SO(n) such that $p \neq I$, we characterize the elements $T_p \mathrm{SO(n)}$ by

$$T_p \mathrm{SO(n)} = \{pA : A \in T_I \mathrm{O(n)}\}\ , \tag{2.34}$$

i.e., $T_p \mathrm{SO(n)}$ (or $T_p \mathrm{O(n)}$) is exactly the set of all $\mathbb{R}^{n \times n}$ that can be written as the matrix

product of the matrix Lie group element p, and some $n \times n$ skew-symmetric matrix.

2.3.5 The Matrix Exponential and Matrix Logarithm

Recall the definition of the matrix exponential of a square matrix A:

Definition 2.3.13 (The matrix exponential)**.**

$$\exp : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n},\ \exp : A \mapsto \sum_{k=1}^{\infty} \frac{A^k}{k!} \tag{2.35}$$

where $A^k = \overbrace{AA\ldots A}^{k \text{ times}}$ is a sequence of matrix products.

For a real-valued square matrix A, we can define the matrix logarithm, denoted $\log(A)$ as a square matrix satisfying $\exp(B) = A$. The matrix exponential is generalization of simple exponential function from the scalar case. There however, some notable differences. For example, let A and B be some two $n \times n$ matrices. In general, $\exp(A)\exp(B) \neq \exp(A+B)$, with equality if and only if $AB - BA = 0$. Likewise, $\log(A^{-1}B) \neq -\log(A) + \log(B)$; however, as a first-order approximation, we have the following important result:

$$\log(A^{-1}B) \approx -\log(A) + \log(B)\ . \tag{2.36}$$

Note that:

1. If A is block-diagonal, then $\exp(A)$ is also block diagonal and each block can be exponentiated independently.

2. In the extreme case that matrix is diagonal, then we can simply use scalar exponentiation for each diagonal entry.

3. As a particular case, $\exp(0_{n\times n}) = I_{n\times n}$ and $\log(I_{n\times n}) = 0_{n\times n}$.

4. If A is nilpotent (so $A^k = 0_{n\times n}$ for some finite k), then the number of summands in Eqn. (2.3.13) is finite.

5. If A is diagonalizable, there is a simple way to exponentiate it using diagonalization and scalar exponentiation.

6. Sometimes, if A has a specific structure (such as, but not limited to, the structures mentioned above) it is possible to compute $\exp(A)$ without having to deal with the infinite sum.

7. If no structure on A can be utilized, then $\exp(A)$ can be efficiently approximated. We will return to this point on Chapter 4.

2.3.6 The Lie Algebras of Matrix Lie Groups

While Lie algebras are mathematical objects that are worth studying for their own merit, and while they can also be defined without any reference to Lie groups, our interest in them is that they provide an indispensable tool when working with Lie groups. In fact, one of main reasons for the attractiveness of Lie groups (and not just matrix Lie groups) is their Lie algebras. We avoid giving the most general definition of Lie algebras[7] and confine ourselves to Lie algebras of real-valued matrix Lie groups.

Definition 2.3.14 (Lie algebra)**.** Let G be a real-valued matrix Lie group. Its Lie algebra $\mathfrak{g}$ is given by the vector space

$$\mathfrak{g} = \exp^{-1}(G), \tag{2.37}$$

and $[\cdot,\cdot] : \mathfrak{g}\times\mathfrak{g} \to \mathfrak{g}$, the *Lie bracket* of $\mathfrak{g}$, is given by $A, B \mapsto AB - BA$.

Remark 2.3.1. Note that the notation $\exp^{-1}(G)$ does not imply that the map $\exp : \mathfrak{g} \to G$ is invertible; rather, it is merely the standard set-theoretic notation for preimage of the set G under the map $\exp : \mathfrak{g} \to G$, which means all those elements (of $\mathbb{R}^{n\times n}$) such that when we exponentiate them, we end up in G. In other words, $\exp^{-1}(G) \stackrel{\text{def}}{=} \{A : \exp(A) \in G\}$.

[7] The interested reader can find that definition in [89].

In particular, depending on G, the map $\exp : \mathfrak{g} \to G$ might not be surjective; *e.g.*, since it can be shown that $\det(\exp(A)) = e^{\mathrm{tr}(A)}$ (and we know the RHS is always positive), it follows that $\exp(A)$ always has a positive determinant. Consequently, if B has a negative determinant (hence $B \in \mathrm{GL(n)}$), then there does not exist an $A \in \mathbb{R}^{n\times n}$ such that $\exp(A) = B$. Moreover, depending on G, the map $\exp : \mathfrak{g} \to G$ might not be injective. This is the case with SO(3) for example.

As in the case of matrix Lie groups, we will usually avoid mentioning the Lie bracket, and simply refer to the vector space as the Lie algebra $\mathfrak{g}$. It turns out that the elements of $\mathfrak{g}$ coincide with those of $T_I G$, the tangent space at the identity. Thus, we will use $\mathfrak{g}$ and $T_I G$ interchangeably.

Definition 2.3.15 (The $\mathfrak{gl}(\mathfrak{n})$ Lie algebra)**.** The Lie algebra of both GL(n) and GL^+(n) is given by

$$\mathfrak{gl}(\mathfrak{n}) \stackrel{\text{def}}{=} \exp^{-1}(\mathrm{GL(n)}) = \exp^{-1}(\mathrm{GL}^+(\mathrm{n})) = \mathbb{R}^{n\times n} \,. \tag{2.38}$$

Definition 2.3.16 (The $\mathfrak{us}(n)$ Lie algebra)**.**

$$\mathfrak{us}(n) \stackrel{\text{def}}{=} \exp^{-1}(\mathrm{US(n)}) = \{Q : Q = sI_{n\times n},\ s \in \mathbb{R}\} \,. \tag{2.39}$$

Definition 2.3.17 (The $\mathfrak{g}_S$ Lie algebra)**.**

$$\mathfrak{g}_S \stackrel{\text{def}}{=} \exp^{-1}(G_S) = \mathbb{R} \,. \tag{2.40}$$

Definition 2.3.18 (The so(n) Lie algebra)**.**

$$\mathrm{so(n)} \stackrel{\text{def}}{=} \exp^{-1}(\mathrm{O(n)}) = \exp^{-1}(\mathrm{SO(n)}) = \left\{Q : Q = -Q^T\right\} \subset \mathfrak{gl}(\mathfrak{n}) \,. \tag{2.41}$$

And again, will single out the 3D case:

Definition 2.3.19 (The so(3) Lie algebra)**.**

$$\mathrm{so(n)} \stackrel{\text{def}}{=} \exp^{-1}(\mathrm{SO}(3)) = \left\{Q : Q = -Q^T\right\} \subset \mathfrak{gl}(\mathfrak{n}) \; . \tag{2.42}$$

More generally, *matrix exponential maps linear subspaces of the Lie algebra to matrix Lie subgroups.* This turns out to be very convenient for doing statistics.

Finally, the Lie algebra of a direct product of matrix Lie groups can be treated in terms that are direct analogs of the way we handle a direct product of matrix Lie groups; we omit the details.

2.3.6.1 The "Vee" and "Hat" Notation

Let G be a D-dimensional matrix Lie group of $n \times n$ matrices. Recall that the Lie algebra of G, denoted by $\mathfrak{g}$, is a D-dimensional linear subspace of $\mathbb{R}^{n \times n}$. There are times when it is convenient to regard elements of $\mathfrak{g}$ as elements of $\mathbb{R}^D$; *i.e.*, as vectors of dimension $D \leq n^2$ rather than $n \times n$ matrices. To this aim, we use the 'Vee" and "Hat" notation:

$$(\cdot)^\vee : \mathfrak{g} \to \mathbb{R}^D \qquad ; \qquad (\cdot)^\wedge : \mathbb{R}^D \to \mathfrak{g} \; . \tag{2.43}$$

As a mnemonic, the "Vee" vectorizes a matrix while its upside-down version, the "hat", is does the opposite. For a given $\mathfrak{g}$, the particular choice of ordering for the vector elements does not matter – as long as, within a particular Lie algebra, we use it consistently.

Example 2.3.2. For $\mathfrak{gl}(2)$, we can use:

$$\left(\begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix}\right)^\vee = \begin{bmatrix} A_{1,1} & A_{1,2} & A_{2,1} & A_{2,2} \end{bmatrix}^T \; ; \tag{2.44}$$

$$\left(\begin{bmatrix} A_{1,1} & A_{1,2} & A_{2,1} & A_{2,2} \end{bmatrix}^T\right)^\wedge = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix} \; . \tag{2.45}$$

Example 2.3.3. For so(3), we can use:

$$\left(\begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}\right)^{\vee} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}; \tag{2.46}$$

$$\left(\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}\right)^{\wedge} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}. \tag{2.47}$$

Remark 2.3.2. If $\mathfrak{g}$ is n^2-dimensional and A is in $\mathfrak{g}$, then

$$\|A\|_F = \left\|(A)^{\vee}\right\|_{\ell^2}.$$

If D is smaller than n^2 then these norms are only proportional to each other. This does not really matter as all norms on finite-dimensional real-valued vector spaces (of the same dimension) are topologically equivalent; however, when doing statistics over a Lie algebra which is a product of, say, so(n) and $\mathfrak{gl}(\mathfrak{n})$, one may want to weight their Frobenius norms differently. Implicitly, as we will see momentarily, this would also translate to different wights of the geodesic distances of the corresponding groups.

2.3.7 The Lie Algebra as a Tool for Utilizing Other Tangent Spaces

The matrix logarithm (or exponential) provides us with a way to move from G to $\mathfrak{g}$ (respectively, $\mathfrak{g}$ to G). Let p and q be in G. Suppose we are interested in expressing q as a tangent vector at T_pG. There are two ways (of interest to us) to do it.

2.3.7.1 Identifying Every Tangent Space with the Lie Algebra

One way to do it is as follows. We first left-translate q by p^{-1}, and then compute the matrix logarithm to get a tangent vector in T_IG:

$$x = \log(p^{-1}q) \in T_IG. \tag{2.48}$$

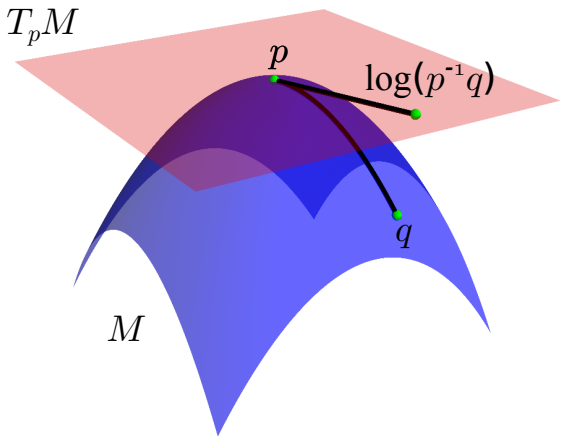


Figure 2.3: On a matrix Lie group M, when T_pM is identified with T_IM, $\log(p^{-1}q)$ can be interpreted as belonging to T_pM, although strictly speaking, it is in T_IM. Also, note that $q = p\exp(\log(p^{-1}q))$.

The result is indeed in T_IG, and not in T_pG. However, as we took the logarithm of the group difference between p and q, and as T_IG and T_pG are both D-dimensional Euclidean spaces, the result makes sense nonetheless. When, as above, we treat T_IG and T_pG as being "the same", we use the notation $x \in T_IG \cong T_pG$. Conversely, when we regard a Lie algebra element x as a vector of T_pG and want express it as a group element, we first map it to the manifold using the matrix exponential, and then left-translate the *group element* by p:

$$p\exp(x) \in G\,. \tag{2.49}$$

If x was computed by Eqn. (2.48), then the smooth curve $c : [0,1] \to G$, given by

$$c(t) = p\exp(tx) = p\exp(t\overbrace{\log(p^{-1}q)}^{x})\,, \tag{2.50}$$

satisfies $c(0) = p$ and $c(1) = q$. See Fig. 2.3 for an illustration.

2.3.7.2 Using the Lie algebra as an Intermediate Step

Here is the second approach. First compute the log of the difference as before, but then, instead of staying in $T_I M$, left-translate the result by p:

$$x = p \underbrace{\log(\overbrace{p^{-1}q}^{\in G})}_{\in T_I G} \in T_p G \; . \tag{2.51}$$

Conversely, to map a tangent vector x from $T_p G$ to G, we first left-translate the *vector* by p^{-1} to map it to $T_I G$, use the matrix exponential to map the result to G, and left-translate the *group element* by p:

$$p \underbrace{\exp(\overbrace{p^{-1}x}^{\in T_I G})}_{\in G} \in G \; . \tag{2.52}$$

If x was computed by Eqn. (2.51), then the expression

$$\begin{aligned} p \exp(tp^{-1}x) &= p \exp(tp^{-1} \overbrace{p \log(p^{-1}q)}^{x}) \\ &= p \exp(t \log(p^{-1}q)) \end{aligned} \tag{2.53}$$

gives the exact the same curve $[0, 1] \to G$ as in Eqn. (2.50).

2.3.8 Matrix Lie Groups as Riemannian Manifolds

For the remainder of Section 2.3, we will assume that G is a real-valued connected matrix Lie group. Matrix Lie groups can be endowed with a Riemannian metric – a term we will discuss and define later. In fact, there are infinitely many choices for a Riemannian metric, although some are usually more useful and popular than others. The question which one should be used is an active research area and the answer depends on both the particular group of interest and the application.

2.3.8.1 The Riemannian Metric Induced from the Ambient Space

Recall that every tangent space T_pG is in fact a set of $n \times n$ matrices. Thus, T_pG can inherit the standard inner-product of $\mathbb{R}^{n\times n}$:

$$\langle \cdot, \cdot \rangle_p : T_pG \times T_pG \to \mathbb{R};\ \langle \cdot, \cdot \rangle_p : (x, y) \mapsto \sum_{i=1}^{n}\sum_{j=1}^{n} x_{i,j}y_{i,j} = \mathrm{tr}(x^T y)\ . \tag{2.54}$$

The collection of these maps, $\{\langle \cdot, \cdot \rangle_p\}_{p\in M}$, is a particular example of Riemannian metric. For every p, the corresponding inner product enables us to define a norm:

$$\|\cdot\|_p : T_pG \to \mathbb{R}^+;\ \|\cdot\|_p : x \mapsto \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n} x_{i,j}^2}\ . \tag{2.55}$$

This norm coincides with the Frobenius norm (see Definition A.3.3, page 205) on $\mathbb{R}^{n\times n}$ and so we will omit the dependency on p (and there is a dependency here: the domain depends on p!) and just write $\|\cdot\|_F$. The inner-product, together with the norm, enables us to define angles (see Definition A.3.5, page 205) between tangent vectors in T_pG.

Let p and q be two points in G, and let c be some smooth curve between these two points satisfying $c(0) = p$ and $c(1) = q$. We define the length of c:

$$\mathrm{length}(c) = \int_{[0,1]} \|\dot{c}(t)\|_F\ dt\ . \tag{2.56}$$

Regarding length as a functional over such curves, we call a curve geodesic if it is a local minimizer. By a slight abuse of notation we will usually refer to the shortest geodesic as *geodesic* between p and q. Also note that even a global minimizer need not be unique although it is guaranteed to exit (this is not true for Riemannian manifolds in general). The length of the shortest curve is called the *geodesic distance*, also known as the *Riemannian distance* between p and q and is denoted by $d(p, q)$.

2.3.8.2 A Lie-Algebraic Riemannian Metric

Alternatively, another popular type of Riemannian metric on matrix Lie groups is obtained by first defining an inner-product on the Lie algebra, and then deriving all other inner-products from it. For example, if $p = I$, we set

$$\langle \cdot, \cdot \rangle_I : T_I G \times T_I G \to \mathbb{R};\ \langle \cdot, \cdot \rangle_I : (x, y) \mapsto \sum_{i=1}^{n} \sum_{j=1}^{n} x_{i,j} y_{i,j} = \mathrm{tr}(x^T y)\ . \tag{2.57}$$

Otherwise:

$$\langle \cdot, \cdot \rangle_p : T_p G \times T_p G \to \mathbb{R};\ \langle \cdot, \cdot \rangle_p : (x, y) \mapsto \left\langle p^{-1}x, p^{-1}y \right\rangle_I\ . \tag{2.58}$$

Note this is well defined: if x is in $T_p G$, then $p^{-1}x$ is in $T_I G$. A norm on $T_p G$ is now defined as $x \mapsto \|p^{-1}x\|$, and the length of a curve $c : [0, 1] \to G$ is defined by:

$$\mathrm{length}(c) = \int_{[0,1]} \left\| c^{-1}(t)\dot{c}(t) \right\|_F\ dt\ . \tag{2.59}$$

Note that $\dot{c}(t)$ is in $T_{c(t)}G$, that $c^{-1}(t)$ is in G, and that $c^{-1}(t)\dot{c}(t)$ is in $T_I G$.

Let p and q be points in G. We call this Riemannian metric *left-invariant*, since it renders the left-translation of vectors from $T_p G$ to $T_q G$ by qp^{-1} an inner-product preserving map:

$$\begin{aligned} \left\langle qp^{-1}x, qp^{-1}y \right\rangle_q &= \left\langle q^{-1}qp^{-1}x, q^{-1}qp^{-1}y \right\rangle_I \\ &= \left\langle p^{-1}x, p^{-1}y \right\rangle_I \\ &= \left\langle pp^{-1}x, pp^{-1}y \right\rangle_p \\ &= \langle x, y \rangle_p\ . \end{aligned} \tag{2.60}$$

Lastly, note that we could have also defined a different inner-product $\langle \cdot, \cdot \rangle_I$ using any SPD matrix and get similar results.

2.3.8.3 Geodesic Distances and Geodesic Curves Using the First Riemannian Metric

If the Riemannian metric is given by Eqn. (2.54), the geodesic curve and geodesic distance between two points p and q in G is given by:

$$c(t) = p\exp(t\log(p^{-1}q))\ ; \tag{2.61}$$

$$d(p,q) = \left\|p\log(p^{-1}q)\right\|_F\ . \tag{2.62}$$

In other words, the initial velocity is $\log(p^{-1}q)$ and the (constant) speed is $\left\|\log(p^{-1}q)\right\|_F$.

2.3.8.4 Geodesic Distances and Geodesic Curves Using the Second Riemannian Metric

Now suppose instead that the Riemannian metric is given by Eqn. (2.58). A recent result by Andruchow *et al.* [5] shows that if $G = \mathrm{GL(n)}$, $p \in G$, and $x \in \mathfrak{gl}(\mathfrak{n})$, then the constant-speed geodesic emanating from p at velocity $px \in T_pG$ is

$$c(t) = p\exp(tx^T)\exp(t(x - x^T))\ . \tag{2.63}$$

Remark 2.3.3. Importantly, in general $c(t) \neq p\exp(tx)$. We will return to this issue later when we discuss Lie-algebraic techniques for dimensionality reduction.

If $x = -x^T$ (*i.e.*, $x \in \mathrm{so(n)}$, then $\exp(t(x - x^T)) = I$ (since $x - x^T = 0_{n\times n}$) and thus $c(t) = p\exp(tx)$ – colliding with a well known result for geodesics on SO(n). More generally we have the following result. Recall that if A and B are two square matrices, then in general $\exp(A+B) \neq \exp(A)\exp(B)$; however, if their Lie bracket (see Definition 2.3.14, page 41) vanishes, then $\exp(A+B) = \exp(A)\exp(B)$. Now, if x is a normal matrix

(namely, $x^T x = x x^T$), then $[x, -x^T] = 0$. Consequently

$$\exp(tx^T)\exp(t(x-x^T)) = \overbrace{\exp(tx^T)\exp(-tx^T)}^{=I}\exp(tx) = \exp(tx)\ , \tag{2.64}$$

and thus $c(t) = p\exp(tx)$. This generalizes the result we had for $x \in \mathrm{so(n)}$. In fact, Andruchow *et al.* [5] showed the converse is also true : if $c : [0,1] \to G$, $c : t \mapsto p\exp(tx)$, is a geodesic, then x must be a normal matrix. For stronger related results see [5].

Consequently, if the Lie algebra of G contains only normal matrices, then all geodesics between two points p and q have form $c(t) = p\exp(t\log(p^1 q))$. This includes SO(n), SPD, and many other groups.

If $c(t) = p\exp(tx)$, then $\dot{c}(t) = p\exp(tx)x = px\exp(tx)$. This, together with Eqn. (2.59), implies that

$$\mathrm{length}(c) = \int_{[0,1]} \|x\|_F \ dt = \|x\|_F \int_{[0,1]} dt = \left\|\log(p^1 q)\right\|_F \ . \tag{2.65}$$

Consequently, for such a matrix Lie group, geodesic distances are given by

$$d(p,q) = \left\|\log(p^{-1}q)\right\|_F \ . \tag{2.66}$$

In which case, by Eqn. (2.36), we can use an approximation:

$$d(p,q) \approx \|-\log(p) + \log(q)\|_F \ . \tag{2.67}$$

Example 2.3.4 (Geodesic distance on G_S)**.** Recall that G_S is nothing more than $\mathbb{R}^+$ with scalar multiplication. Let p and q be two positive numbers. With the Riemannian metric given by Eqn. (2.58), we see that $d(p,q) = |\log(p/q)| = |\log(p) - \log(q)|$; *i.e.*, here Eqn. (2.67) is not an approximation but an exact equality. This is the result of G_S being Abelian[8].

[8] If two matrices p and q do not commute, then $\log(p^{-1}q) \neq -\log(p) + \log(q)$.

Remark 2.3.4. Even when the geodesic distance is not given by Eqn. (2.66), the distance from Eqn. (2.66) is a perfectly legitimate distance function $G \times G \to G$. In fact it is even a metric. It is just not tied to the Riemannian metric of the manifold.

Note that the distance in Eqn. (2.66) is left-invariant: $d(p_1, p_2) = d(p_3 p_1, p_3 p_2)$ for any triplet of points in G. In contrast, the distance in Eqn. (2.62) is not left-invariant.

Finally, note that if G is a group of isometries (when regarded as maps $T_pG \to T_qG$), then Eqn. (2.66) and Eqn. (2.62) coincide: $\|p \log(p^{-1}q)\|_F = \|\log(p^{-1}q)\|_F$. This happens for SO(n) for example; in which case, the distance in Eqn. (2.62) also becomes left-invariant.

2.4 Elements of Manifold Theory

In most introductory undergraduate-level courses on manifolds, these mathematical objects are usually defined with respect to some ambient Euclidean space. Indeed, in many applications – including the examples used in this work – the manifold of interest can be formally defined as a certain kind of subset of a Euclidean space. While useful for acquiring initial understanding and geometric intuition it turns out that this approach not only renders many notions and operations needlessly obfuscated and cumbersome but also forces readers (and authors...) to carry plenty of redundant technical baggage with them as the theory is being developed.

Fortunately, there is a better approach.

This modern approach to manifold theory is more abstract and requires a higher level of mathematical maturity. Besides being more general, this approach enables us to gain a cleaner and deeper insight into many manifold-related notions, including those covered in this work. This is one of those cases in mathematics, where even when abstract generality is not strictly required, it actually makes things easier and enhances clarity[9]. That being

[9] Some authors may disagree with this view.

said, *the reader may want to skip the current section (Section 2.4) at first reading and return to it later if needed.*

In the current section we provide a short introduction to this abstract theory, covering several of its key elements. A more thorough treatment can be found in standard textbooks such as [27, 88–90]. Additionally, a gentler introduction to the topic can be found in [98].

A *Riemannian manifold* is a generalization of a Euclidean space. Informally, Riemannian manifolds are *smooth* spaces on which we can generalize the notions of angles and distances from $\mathbb{R}^n$. Under an additional condition[10], a Riemannian manifold is also a *metric space*. Importantly, a manifold need not be linear, let alone an inner-product space; however, at least locally it is equivalent to $\mathbb{R}^n$ in some topological sense. A more formal definition of a Riemannian manifold will be given at Section 2.4.3. The main purpose of the current section is to provide us with the building blocks that are necessary for understanding that definition.

Riemannian manifolds, like Ogres and onions, have layers[11]. The first layer is topological [90, 98], the second is a *smooth structure* [89], and the third is the *Riemannian metric* [27, 88]. To avoid tears, we will peel the layers one by one.

2.4.1 Topological Manifolds

Let us clarify in what topological sense a manifold is at least locally equivalent to $\mathbb{R}^n$.

Definition 2.4.1 (Locally Euclidean space of dimension n)**.** A topological space (see Definition A.1.1, page 199) M is called an n-dimensional *locally Euclidean* space if every point in M has an open neighborhood that is *homeomorphic* to an open subset of $\mathbb{R}^n$. In other words, for every $p \in M$, there exist:

[10] This condition holds for all manifolds relevant to this work.
[11] See [Shrek, 2001] and maybe [Shrek II, 2004]; do not bother seeing the other sequels.

1. an open set $U \subset M$ containing p;

2. an open set $\widetilde{U} \subset \mathbb{R}^n$;

3. a *homeomorphism* (see Definition A.1.8, page 203) $\varphi : U \to \widetilde{U}$.

A topological manifold is a locally Euclidean space that also satisfies two more properties.

Definition 2.4.2 (Topological Manifold)**.** A topological space M is called an *n-dimensional topological manifold* if all of the following three properties hold:

1. M is an n-dimensional locally Euclidean space;

2. M is a *Hausdorff space*;

3. M is *second countable*.

Definition 2.4.2 utilizes the topological notions of a Hausdorff space and second countability. These concepts are defined in Section A.1 in the Appendix. Informally, the Hausdorff space property ensures us that we have "enough" open sets and the second countability property ensures us that we do not have "too many" open sets. These particular topological considerations are only peripheral to the current work; we avoid going into further details[12].

It is worth noting what is *absent* in Definition 2.4.2: the notion of an ambient space. Namely, there is nothing in the definition that forces M to be a subset of $\mathbb{R}^{n'}$ (for some $n' \geq n$). In many undergraduate-level textbooks, a less abstract definition is often used – one that assumes the presence of an ambient space, and that M is homeomorphic to one of its subsets. The definition we use here is more general. Before proceeding with our discussion based on this definition, let us point out a particular space for which the

[12]The interested reader can find an in-depth treatment of these concepts in topology textbooks such as [108] while their implications on manifolds can be found in differential topology textbooks such as [90].

ambient space approach is not very useful. This space is called the projective plane and is of great importance in computer vision; *e.g.*, see Hartley and Zisserman [61].

Example 2.4.1 (The projective space)**.** Let $\mathbb{R}^3 - \{0\}$ denote the set of all points in $\mathbb{R}^3$ but the origin. Similarly, let $\mathbb{R} - \{0\}$ denote the real line without zero. Consider the real projective plane:

$$\mathbb{RP}^2 = \left\{[x] : \ x \in \mathbb{R}^3 - \{0\}\right\} , \tag{2.68}$$

where $[x] = \{y : y \in \mathbb{R}^3 - \{0\}, \ \exists \lambda \in \mathbb{R} - \{0\} \text{ such that } y = \lambda x\}$. In other words, $[x]$ is made of all 3D lines that pass through the origin and are parallel to x. Clearly, according to any reasonable definition of a dimension, $\mathbb{RP}^2$ is two-dimensional. It turns out, however, that is impossible to embed $\mathbb{RP}^2$ in $\mathbb{R}^3$. It is possible to embed it $\mathbb{R}^4$ – so the ambient approach would apply here – but this is not so obvious from the definition and leads to needless complications. Using the abstract definition, however, leads to a simpler and more practical way to work with this space as a manifold; we omit the details. The statistical tools we will describe in Section 2.5, as well as the new one we present in Chapter 5, apply to this manifold as well.

We proceed to the notions of *charts* and local coordinates.

Definition 2.4.3 (Coordinate charts; coordinate maps; local coordinates)**.** Let M be an n-dimensional topological manifold. A *coordinate chart* on M is a pair (U, φ), where U is an open subset of M and $\varphi : U \to \widetilde{U}$ is a homeomorphism from U to an open subset $\widetilde{U} = \varphi(U) \subset \mathbb{R}^n$. φ is called a *(local) coordinate map* and the component functions $(\widetilde{x}^1, \ldots, \widetilde{x}^n)$ of φ (each one of which is a function from M to $\mathbb{R}$), defined by $\varphi(p) = (\widetilde{x}^1(p), \ldots, \widetilde{x}^n(p))$, are called *local coordinates* on U.

Remark 2.4.1. Our use of superscripts to denote different coordinates is in no contradiction with our use of subscripts mentioned earlier (Remark 2.1.1), as there we referred to the components of a data point; in contrast, here we refer to the coordinates of a chart. To make this distinction stand out even further, note that whenever we refer to coordinates of a chart we also use the tilde notation (as in $\widetilde{x}_1$). We will retire this notation at the end

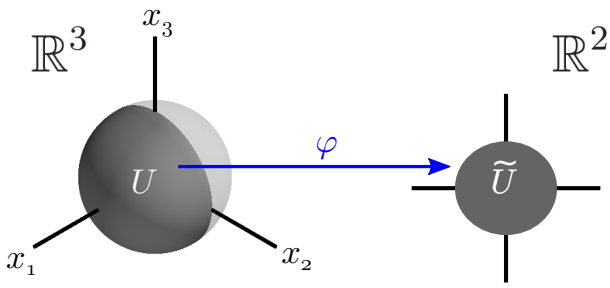


Figure 2.4: A chart on S^2.

of this section, as from that point onward we will no longer need a direct reference to local coordinates.

Example 2.4.2 (A chart on S^2)**.** Let $M = S^2$ (namely, the unit sphere in $\mathbb{R}^3$). Let $U = \{(x_1, x_2, x_3) \in M,\ x_1 > 0\}$, and let $\varphi : U \to \widetilde{U} = \{(x_1, x_2) : x_1^2 + x_2^2 < 1\} \subset \mathbb{R}^2$ be defined by $\varphi : (x_1, x_2, x_3) \mapsto (x_2, x_3)$. See Fig. 2.4 for an illustration. It can be shown that φ is a homeomorphism and thus the pair (U, φ) is a coordinate chart (note that U and $\widetilde{U}$ are open subsets of M and $\mathbb{R}^2$ respectively). Let $p = (x_1, x_2, x_3) \in U$. In terms of coordinates, we have that $\varphi(p) = (\widetilde{x}^1(p), \widetilde{x}^2(p))$ with

$$\begin{aligned} &\widetilde{x}^1 : U \to \mathbb{R},\ \widetilde{x}^1 : (x_1, x_2, x_3) \mapsto x_2; \\ &\widetilde{x}^2 : U \to \mathbb{R},\ \widetilde{x}^2 : (x_1, x_2, x_3) \mapsto x_3. \end{aligned} \tag{2.69}$$

Considering Example 2.4.2, it is easy to see that there exist many other possible choices (besides φ) for a homeomorphism from U to some open subset of $\mathbb{R}^2$ (and this subset may or may not coincide with $\widetilde{U}$). Moreover, if V is some open subset of U, we can construct two charts, (U, φ) and (V, ψ), such that the restriction of φ to V does not coincide with ψ. The take-home message is simple: *charts are not unique*. We conclude our discussion of topological manifolds with the definition of a *curve*.

Definition 2.4.4 (Curve)**.** A *(parametrized) curve* in M is a continuous map $c : J \to M$, where $J \subset \mathbb{R}$ is an interval[13].

[13] As both M and J are topological spaces, the continuity of c is well defined; see Definition A.1.6, page 201.

2.4.2 Smooth Manifolds

In most applications, we are interested in performing calculus operations such as taking derivatives, defining smooth curves *etc*. To that aim, topological properties do not suffice and a notion of smoothness is required. The main reference for this section is [89].

2.4.2.1 Smooth Structures

Let M be an n-dimensional topological manifold, and let $f : M \to \mathbb{R}^m$ be an $\mathbb{R}^m$-valued function (in particular, if $m = 1$, then f is a real-valued function). With only Definition 2.4.2 at our disposal, it is unclear how to define the derivative of f at some point p in M. A plausible idea is to try to exploit the local-Euclidean property; *i.e*., we would first define a new function $\widetilde{f}$ by

$$\widetilde{f} : \widetilde{U} \to \mathbb{R}^m, \ \widetilde{f} : x \mapsto f(\varphi^{-1}(x)), \tag{2.70}$$

(reusing the notation from Definition 2.4.1) where $x \in \widetilde{U} \subset \mathbb{R}^n$. In other words, $\widetilde{f} = f \circ \varphi^{-1}$. Then, since $\widetilde{f}$ is a map from (an open subset of) $\mathbb{R}^n$ to $\mathbb{R}^m$, we may try to use the multivariate-calculus definition of a derivative of $\widetilde{f}$ as the definition of derivative of f – provided of course that $\widetilde{f}$ is differentiable in the sense of ordinary calculus. There is, however, a problem with this solution: charts are not unique. This is true even if we restrict the discussion to charts that would render $\widetilde{f}$ differentiable. Thus, this solution would have led us to an ambiguous definition of the derivative of f. To deal with this difficulty, what we need is a definition of a derivative that is invariant to the particular choice of a chart. For this, we need the notion of a *smooth structure*.

Let (U, φ) and (V, ψ) be two charts on M such that $U \cap V \neq \emptyset$. The map $\psi \circ \varphi^{-1} : \varphi(U \cap V) \to \psi(U \cap V)$ is called the *transition map* (from φ to ψ) and is a homeomorphism between $\varphi(U \cap V) \subset \mathbb{R}^n$ and $\psi(U \cap V) \subset \mathbb{R}^n$, being the composition of homeomorphisms (see Corollary A.1.1, page 201). See Fig. 2.5.

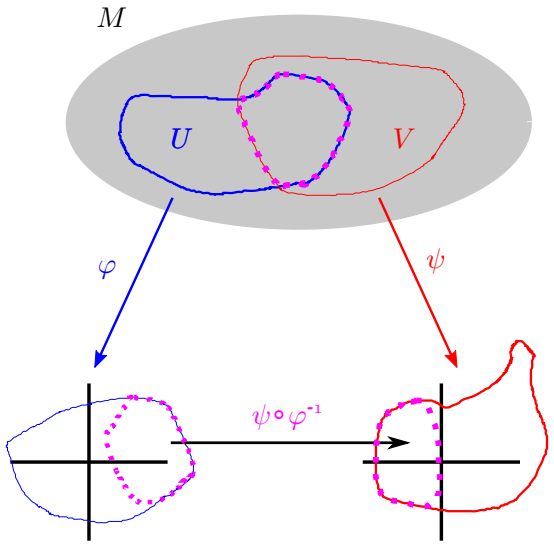


Figure 2.5: An illustration of a transition map between charts.

Two charts (U, φ) and (V, ψ) are said to be *smoothly compatible* if either $U \cap V = \emptyset$ or the transition map $\psi \circ \varphi^{-1}$ is a diffeomorphism[14]. Note that since differentiability implies continuity, a diffeomorphism is also a homeomorphism. An *Atlas* (for M) is any collection of charts whose domains form an *open cover* (see Definition A.1.3, page 200) of M. An atlas is said to be a *smooth atlas* if any two charts in it are smoothly compatible with each other. A smooth atlas $\mathcal{A}$ on M is said to a *maximal smooth atlas* if it is not contained in any strictly larger smooth atlas; *i.e.*, any chart that is smoothly compatible with every chart in $\mathcal{A}$ is also a member of $\mathcal{A}$. A maximal smooth atlas on an n-dimensional topological manifold M is called a *smooth structure* on M. At last, we now define what a *smooth manifold* is.

Definition 2.4.5 (Smooth manifold)**.** A *smooth manifold* M is a pair $(M, \mathcal{A})$, where M is an n-dimensional topological manifold and $\mathcal{A}$ is a smooth structure on M.

When $\mathcal{A}$ is understood or when its details are irrelevant to our discussion, we usually omit it and say that "M is a smooth manifold". Henceforth, whenever we refer to a chart, we implicitly mean a chart that belongs to the smooth structure.

Remark 2.4.2. While every smooth atlas on M is contained in a unique maximal smooth

[14] As a map between two open subsets of $\mathbb{R}^n$; see Definition A.2.2, page 203.

atlas, a maximal smooth atlas is not unique. For example, there can be two maximal smooth atlases with an empty intersection (*i.e.*, no chart appears in both). Consequently, there can be more than one smooth structure on M.

On a smooth manifold, we have meaningful definitions for differentiable or smooth maps from M to $\mathbb{R}^m$. Let M be a smooth manifold, let $p \in M$. We say that $f : M \to \mathbb{R}^m$ is differentiable (or smooth) at p if there exists a chart (U, φ) such that $p \in U$ and $f \circ \varphi^{-1} : \varphi(U) \to \mathbb{R}^m$ is differentiable (or smooth) at $\varphi(p)$. If f is differentiable (or smooth) at every $p \in M$ then we simply say that f is differentiable (or smooth). The important point here is that it can be shown that these definitions of differentiability and smoothness of f do not depend on the particular choice of the chart (U, φ). In fact, we do not need to restrict ourselves to maps whose codomain is a Euclidean space.

Definition 2.4.6 (Differentiable and smooth maps between manifolds)**.** Let M and N be two smooth manifolds, and let $f : M \to N$ be any map. We say that f is differentiable (or smooth) if for every $p \in M$ there exist smooth charts (U, φ) and (V, ψ), on M and N respectively, such that:

1. $p \in U \subset M$;
2. $f(U) \subset V \subset N$;
3. The map $\psi \circ f \circ \varphi^{-1} : \varphi(U) \to \psi(V)$ is differentiable (or smooth) in the Euclidean sense.

Note this definition includes the cases where M or N are Euclidean spaces. Also, since an interval $J \subset \mathbb{R}$ can be seen as a smooth manifold[15], it also makes sense to talk about differentiable (or smooth) curves of the form $c : J \to M$ where here the triplet (c, J, M) takes the role of (f, M, N) from Definition 2.4.6.

[15] In this work we are not going to trouble ourselves with the distinction between manifolds with and without a boundary – but note that many of the ideas covered in this chapter can be adapted to manifolds with boundaries.

Other definitions and results follow naturally. For example, a *diffeomorphism* between two manifolds is a smooth bijective map whose inverse is also smooth. Another example is that the composition of smooth maps is smooth. In particular, if c is a smooth curve in M and $f : M \to \mathbb{R}$ is smooth, then so is $f \circ c : J \to \mathbb{R}$. The matrix exponential (see Definition 2.3.13, page 40), is a smooth map from a Euclidean space (*i.e.*, the tangent space at the identity) to the manifold (*i.e.*, the matrix Lie group). In fact, this map can also be used to construct the charts that make up the smooth structure. Examples of smooth manifolds include: matrix Lie groups; $\mathbb{R}^n$; the projective plan; the unit sphere; SPD matrices. Henceforth, unless stated otherwise, whenever we refer to a manifold we mean a smooth manifold.

Manifolds are usually nonlinear. To cope with the nonlinearity, local linearizion is usually employed. At every point p in a smooth manifold M, there is a linear space, denoted by T_pM, which is *tangent to the manifold* at p. The elements of T_pM are called *tangent vectors*.

Remark 2.4.3. It may seem obvious that an element of a tangent space is a tangent vector – but recall we have yet to *define* what a tangent vector is. Thus, at this point we need to be content with merely stating that if an element belong to T_pM, then we call it a tangent vector. The actual, and somewhat surprising definition, will follow shortly.

There is a important class of manifolds that have an additional, algebraic, structure. These are known as Lie groups.

Definition 2.4.7. A *Lie* group G is a group that is also a smooth manifold and whose operations, composition ($G \times G \to G$) and inversion ($G \to G$), are smooth.

In particular, matrix Lie groups are Lie groups. All the notions we discussed in Section 2.3, can be generalized to Lie groups. For example, the matrix exponential is a particular example the *Lie group exponential map*. It also turns out that the Lie group exponential map can be used for constructing a smooth structure.

In many senses, Matrix Lie groups are easier to understand and work with than the more general Lie groups. In this work, all Lie groups are matrix Lie groups.

2.4.2.2 Tangent Spaces and Tangent Vectors: An Informal Discussion

When we discussed matrix Lie groups we mentioned that tangent vectors coincide with (equivalence classes of) derivatives of smooth curves. We also used our geometrical intuition from $\mathbb{R}^n$, suggesting that T_pM is a linear subspace, whose origin is p, that provides a linear approximation to M at p (recall Fig. 2.2). Following these lines, we think of tangent vectors as "arrows" attached to M at p.

The tricky part is that tangency is not so simple as it may seem at first. Section 2.4.2.3, which provides a glimpse into this issue, is more abstract than the rest of this work and may be omitted during reading.

2.4.2.3 Tangent Spaces and Tangent Vectors: An Abstract Definition

> "Everybody thinks they know what a tangent vector is - but only till hearing the actual definition."
> *Prof. Basilis Gidas.*

The intuitive way of thinking of tangent vectors we have just described relies on the presence of an ambient space. As discussed earlier, the latter is not guaranteed to exist, and even if it does exists, we are better off without resorting to it. Once we realize ~~there is no spoon~~ there is no reference to an ambient space, it becomes clear we can expect a more powerful notion of tangency. There are several, equivalent, abstract ways to define tangent vectors. The following one is relying on the notion of *derivations*.

Definition 2.4.8 (Tangent space and tangent vectors)**.** Let $C^\infty(M)$ denote the class of all smooth maps from M to $\mathbb{R}$. A linear map $X : C^\infty \to \mathbb{R}$ is called a *derivation* at p if it

satisfies

$$X(fg) = f(p)X(g) + g(p)X(f) \,, \tag{2.71}$$

for all $f, g \in C^\infty(M)$. The set of all derivations at p is a vector space denoted by T_pM. This space is called a *tangent space* (to M, at p). An element of T_pM is called a *tangent vector*.

Let us go over the notation in Eqn. (2.71). fg is a smooth map defined as the pointwise (scalar) multiplication of f and g; *i.e.*, $fg : M \to \mathbb{R}$, $fg : p \mapsto f(p)g(p)$. Consequently, $X(fg)$ just means we apply X to the smooth map $fg : M \to \mathbb{R}$. Thus, the LHS of the equation is a real number. The pair $f(p)$ and $g(p)$ are real numbers too: the evaluations of f and g at p. Since X is a functional over smooth functions, the notation $X(g)$ reads as the real number that X assigns to g. Similarly, $X(f)$ is another real number.

In the interest of space, we will not go further into this topic, but here is the important thing to understand: rather than viewing a tangent vector as a vector hovering in some (perhaps nonexistent) ambient space, it is seen as an operator that acts on real-valued smooth functions on M.

Suppose there is indeed some ambient space. Earlier we regarded T_pM as an n-dimensional subspace of that space. Loosely speaking, here is the connection between the two notions. Let $\{e_1, e_2, \ldots, e_n\}$ denote an orthonormal basis of this subspace. The tangent vector X, mentioned above, is in fact a differential operator that can be interpreted as a linear combination of n directional derivatives. The direction of the i-th directional derivative coincides with e_i. If we define a new linear combination of the e_i's (instead of the directional derivatives) using the same weights, we will get a vector in that subspace.

Another equivalent abstract definition of tangent vectors can be made in terms of one-dimensional derivatives of smooth functions $M \to \mathbb{R}$ when restricted to smooth curves that pass through p. Again, we get that a tangent vector is a differential operator, and it can be related to a derivative of a curve with respect to its parameter. Thus, similarly to what

we saw in Section 2.3.4.2, if $c : J \to M$ is a smooth curve, then (an equivalence class of) $\dot{c}(t)$ is always a tangent vector; *i.e.* $\dot{c}(t) \in T_pM$.

See [89] for an in-depth treatment.

2.4.3 Riemannian Manifolds

Smooth manifolds enable us to do calculus, but that is not enough: we would like to be able to define angles and measure distances. For this we need Riemannian geometry[16] of which we here provide only a few concepts. A more thorough treatment can be found in Riemannian geometry textbooks such as [27, 88].

Let M be an n-dimensional smooth manifold. A matrix-valued function $M \to \mathbb{R}^{n\times n}$ is said to *vary smoothly* (on M) if its entries are smooth functions $M \to \mathbb{R}$.

Definition 2.4.9 (Riemannian metrics and Riemannian manifolds)**.** Let M be a smooth manifold, and let $\{\langle\cdot,\cdot\rangle_p\}_{p\in M}$ be a collection of inner-products,

$$\langle\cdot,\cdot\rangle_p : T_pM \times T_pM \to \mathbb{R} \; ; \tag{2.72}$$

$$\langle\cdot,\cdot\rangle_p : (x, y) \mapsto x^T A_p y \; , \tag{2.73}$$

such that A_p is a symmetric positive-definite matrix depending on p. If A_p varies smoothly, then $\{\langle\cdot,\cdot\rangle_p\}_{p\in M}$ is called a *Riemannian metric* on M and M is called a *Riemannian manifold*. Equivalently, we say that a Riemannian metric is a smoothly-varying inner-product.

Compare Definition 2.4.9 with Eqns. (2.14), (2.54), and (2.58).

Let $c : [0, 1] \to M$ be a smooth curve. Recall that $\dot{c}(t) \in T_{c(t)}M$. Thus, the norm $\|\dot{c}(t)\|_p = \langle\dot{c}(t), \dot{c}(t)\rangle_p^{1/2}$ is well defined. This norm is integrable along c, providing a notion

[16] Riemannian geometry also provides a way to measure the curvature of the manifold – but this is beyond the scope of the current work.

of curve length:

$$\text{length}(c) = \int_{[0,1]} \|\dot{c}(t)\|_p \, dt \; . \tag{2.74}$$

Compare Definition 2.74 with Eqns. (2.56) and (2.59).

Let p and q be in M. Let $C_{p,q}$ denote the class of all smooth curves $[0,1] \to M$ such that $c(0) = p$ and $c(1) = q$.

The *geodesic distance* between p and q is defined as:

$$d(p,q) = \inf_{c \in C_{p,q}} \text{length}(c) \; . \tag{2.75}$$

By possible re-parametrization, c can become a curve of constant speed; *i.e.*,

$$\|\dot{c}(t)\|_p = \text{const} \; \forall t \in [0,1] \; .$$

The constant depends on the curve. Such re-parametrization does not change the length of the curve and so we may assume that all curves in $C_{p,q}$ are of constant speed. Note that in general, $C_{p,q}$ might be empty. By convention, the infimum of an empty set is $+\infty$.

Example 2.4.3 (A geodesic distance might be infinite)**.** If the manifold is GL(n), and $\det p > 0$ while $\det q < 0$, then $C_{p,q} = \emptyset$. Thus, $d(p,q) = +\infty$. Of course, in this example the manifold is not connected.

A *geodesic curve* between p and q is an element of $C_{p,q}$ which is a local minimizer of the function that maps a curve c to its length. Note that even if $C_{p,q}$ is nonempty, the infimum in Eqn. (2.75) might not be achievable (hence the use of inf instead of min); *i.e.*, the fact that the points can be connected by a smooth curve does not imply the existence of a geodesic curve.

Example 2.4.4 (Even if a geodesic distance is finite, a geodesic curve need not exist)**.** Consider the manifold $\mathbb{R}^2 - \{0\} \stackrel{\text{def}}{=} \{x : x \in \mathbb{R}^2, x \neq (0,0)\}$. Let $p = (1,0)$ and let $q = (-1,0)$. Assuming the standard Riemannian metric inherited from the ambient space

$(\mathbb{R}^2)$, it is clear that $d(p,q) = 2$. It is easy to construct a sequence $\{c_i\}_{i=1}^{\infty}$ in $C_{p,q}$ such that the sequence length$(c_i) \xrightarrow{i\to\infty} 2$. Yet, no element of $C_{p,q}$ attains that length.

Consequently, it is convenient to restrict discussion to manifolds that are well-behaved.

Definition 2.4.10 (Geodesically-complete Riemannian manifolds)**.** If for every two points in M there exists a geodesic curve between them that connects the two, then M is called *geodesically complete*.

Importantly, if M is a geodesically-complete Riemannian manifold, then $d(\cdot,\cdot) : M \times M \to \mathbb{R}^+$ makes M a metric space.

2.4.3.1 Riemannian Exponential Map and Riemannian Logarithm

Henceforth M will be assumed to be a geodesically-complete Riemannian manifold. If $(p,x) \in M \times T_pM$, then there exists a unique geodesic c such that $c(0) = p$, $\dot{c}(0) = x$; *i.e.*, c is a particular geodesic between $c(0)$ and $c(1)$, and we say that c is a geodesic emanating from p with the initial direction x. The Riemannian exponential map, denoted by $\mathrm{Exp}_p : T_pM \to M$, maps a tangent vector x to $c(1)$, where c is the unique geodesic associated with p and x. The Riemannian logarithm, denoted by $\mathrm{Log}_p : M \to T_pM$, does the opposite; *i.e.*, if p and q are in M, then $\mathrm{Log}_p(q)$ is a tangent vector x satisfying $\mathrm{Exp}_p(x) = q$. It follows that the associated geodesic between p and q is given by

$$c : [0,1] \to M,\ c :\mapsto \mathrm{Exp}_p(t\mathrm{Log}_p(q)) \ . \tag{2.76}$$

Compare Eqn. (2.76) with Eqns. (2.20) and (2.61). Likewise, the geodesic distance between p and q is given by

$$d(p,q) = \left\|\mathrm{Log}_p(q)\right\|_p \ . \tag{2.77}$$

Compare Eqn. (2.76) with Eqns. (2.21), (2.62), and (2.66).

Note that the matrix Lie group exponential map (namely, the matrix exponential) should not be confused with the Riemannian exponential map, although they are not unrelated. For example, in sections 2.3.7.1 and 2.3.7.2 we saw that if M is a matrix Lie group, then the matrix exponential/logarithm pair enables two ways to utilize T_pM. The method from Section 2.3.7.1 was purely Lie-algebraic and relied on the identification of T_IM with T_pM. We now interpret the method from Section 2.3.7.2 to coincide with the Riemannian approach when the Riemannian metric is given by Eqn. (2.54). In which case, the Riemannian exponential and logarithm maps are given by

$$\mathrm{Log}_p : M \to T_pM \quad , \quad \mathrm{Log}_p : q \mapsto p\log(p^{-1}q)\ ; \tag{2.78}$$

$$\mathrm{Exp}_p : T_pM \to M \quad , \quad \mathrm{Exp}_p : x \mapsto p\exp(p^{-1}x)\ . \tag{2.79}$$

2.4.3.2 Geodesic Subspaces

Definition 2.4.11 (Geodesic subspaces)**.** Let $V = [V_1, V_2, \ldots, V_k]$ denote a matrix whose columns form an orthonormal basis of a k-dimensional linear subspace of T_pM. The (usually nonlinear) subset of M, defined by

$$\left\{q \in M :\ q = \mathrm{Exp}_p(V\alpha^T),\ \alpha \in \mathbb{R}^K\right\}\ , \tag{2.80}$$

is called a *geodesic subspace* of M.

For example, if M is a matrix Lie group, if $p = I$, and if Exp_p is given by $\mathrm{Exp}_p : x \mapsto \exp(p^{-1}x)$, then a matrix Lie subgroup constructed from a subspace of the Lie algebra (as was discussed in Section 2.3.6) is a geodesic subspace.

2.5 Statistics on Manifolds

2.5.1 Statistics on Manifolds: What It Is and What It Is Not

When it comes to manifolds, there are several different fields in statistics and machine learning that share similar terminology. While all these fields are not entirely unrelated to each other, it is important to understand the key differences between them. The field that is of interest to us, and whose origins go back to Ronald Fisher's work on spherical data [33], is called *statistics on manifolds*. In this section we try to clarify the differences between this field and two others who use similar terminology.

2.5.1.1 Statistics on Manifolds *vs*. Manifold Learning

Importantly, statistics on manifolds is very different from *manifold learning*. The latter is a branch of machine learning where the goal is to learn a latent manifold from $\mathbb{R}^n$-valued data. Typically, the dimension of the sought-after latent manifold is less than n. The latent manifold may be linear or nonlinear, depending on the particular method used.

In sharp contrast, in the field of statistics on manifolds, the manifold is known and the goal is to study statistics of data restricted to this manifold; *i.e.* its wide scope encompasses every flavor of statistics – be it descriptive statistics or statistical inference – as long as we are concerned with *manifold-valued data*. This includes, for example, parameter estimation, regression, nonparametric statistics, and hypothesis testing. In particular, note that a manifold does not necessarily have to be of lower dimension than its ambient space (should such a space exist) and that the statistical task may not be related to dimensionality reduction (which is usually a primary goal in manifold learning).

Example 2.5.1. Consider $\mathbb{R}^{n\times n}$ and its subset, the $\mathrm{GL}^+(\mathrm{n})$ manifold. Note that the dimension of $\mathrm{GL}^+(\mathrm{n})$, just like that of $\mathbb{R}^{n\times n}$, is n^2. Given $\mathrm{GL}^+(\mathrm{n})$-valued data, it is unwise to regard the data as $\mathbb{R}^{n\times n}$-valued, as by doing so we *discard our knowledge of the structure*

of the manifold and its distance function(s). Instead, this knowledge should be exploited by incorporating it into our statistical procedures. For a concrete example, see [104], where the authors show a practical method for nonparametric density estimation on $\mathrm{GL}^+(\mathrm{n})$, and its advantages over similar estimators that fail to take advantage of the manifold structure.

Of course, on the known manifold, a lower-dimensional submanifold can still be learned using techniques that utilize the structure and distances on the original known manifold, e.g., principal geodesic analysis (PGA) which generalizes principal component analysis (PCA). More on that later (Section 2.5.3.1). The take-home message is simple: *Use what you know. Learn (statistically) what you do not.* Putting it differently, manifold learning is a good hammer, but not all problems are nails.

2.5.1.2 Statistics on Manifolds *vs*. Information Geometry

Statistics on manifolds and information geometry are two different ways in which differential geometry meets statistics. While in statistics on manifolds, it is the *data* that lie on a manifold, in information geometry the data are in $\mathbb{R}^n$, but the parameterized family of probability density functions of interest is treated as a manifold. Such manifolds are known as *statistical manifolds*. For example, if we parameterize the family of one-dimensional Gaussians by $M = \{\mathcal{N}(\mu, \sigma^2) : (\mu, \sigma^2) \in \mathbb{R} \times \mathbb{R}^+\}$ then it is clear that M is nonlinear. Moreover, it turns out that statistical or information-theoretic concepts such as Fisher's information matrix or the Kullback-Leibler divergence are related to the way distances are measured on M. Loosely speaking, in statistics on manifolds one tries to do statistics while taking the differential geometry of the space in which the data lie into account, while in information geometry one uses tools from differential geometry to solve problems in Euclidean statistical inference. In this thesis we will not touch upon information geometry; the interested reader is referred to one of the few standard textbooks in this field: [3, 109]. See also Guy Lebanon's thesis [87].

2.5.2 Basic Concepts

An introduction to statistics on Riemannian manifolds can be found in [113]. See also [10] for a recent book on nonparametric statistics on manifolds. For the remainder of Section 2.5, we will assume that M is a D-dimensional geodesically-complete Riemannian manifold, and that we have an M-valued dataset denoted by $p_1, p_2, \ldots, p_N$.

We start with generalizing the Euclidean notion of sum of squared distances to sum of squared geodesic distances.

Definition 2.5.1 (The sample Fréchet function)**.** Let p be a point in M. The sum of squared distances between the data and p is called the sample *Fréchet function* defined by

$$\mathrm{SSGD}(p) \stackrel{\mathrm{def}}{=} \frac{1}{N} \sum_{i=1}^{N} d(p, p_i)^2 \ . \tag{2.81}$$

Next we generalize the Euclidean notion of the sample mean.

Definition 2.5.2 (Intrinsic mean)**.** The unique global minimizer of the function SSGD : $M \to \mathbb{R}^+$, if it exists, is called the (sample) *intrinsic mean*. It is also known as the (sample) *Fréchet mean*. Any local minimizer is called the *Karcher mean* [77].

As is common in applications, we use the Karcher mean. In practice, the Karcher mean can be efficiently computed using an iterative algorithm [113].

Let μ denote the intrinsic mean. The value the (sample) Fréchet function attains at μ,

$$\frac{1}{N} \sum_{i=1}^{N} d(\mu, p_i)^2 \ , \tag{2.82}$$

is called the *geodesic variance*.

The (sample) covariance is defined through the Euclidean (sample) covariance of the

data as expressed in $T_\mu M$:

$$\mathrm{Cov}(\{p_i\}_{i=1}^N) \stackrel{\mathrm{def}}{=} \frac{1}{N-1}\sum_{i=1}^{N} \mathrm{Log}_\mu(p_i)\mathrm{Log}_\mu(p_i)^T \ . \tag{2.83}$$

Note that the point of tangency is the intrinsic mean, μ. This echoes (and in fact, generalizes) the construction of the Euclidean (sample) covariance in a Euclidean space, which is built from summing outer-products of vectors following the subtraction of the Euclidean (sample) mean:

$$\mathrm{Cov}(\{p_i\}_{i=1}^N) \stackrel{\mathrm{def}}{=} \frac{1}{N-1}\sum_{i=1}^{N} \left(p_i - \frac{1}{N}\sum_{j=1}^{N} p_j\right)\left(p_i - \frac{1}{N}\sum_{j=1}^{N} p_j\right)^T \ . \tag{2.84}$$

2.5.3 Generalizing PCA

2.5.3.1 Two Different Generalizations of PCA

It is well known that in $\mathbb{R}^n$, PCA is the solution to (at least) two optimization problems over subspaces: 1) The maximization of variance captured by the subspace; 2) Minimization of the squared norm of the error caused by projecting the data into the subspace. While both problems can be generalized to manifolds, where the optimization is done over geodesic subspaces (see Definition 2.4.11, page 65), their optimizers need not coincide. In this work we restrict discussion to the first generalization, known as principal geodesic analysis (PGA). See [73] for the second type, known as geodesic principal component analysis (GPCA).

2.5.3.2 Principal Geodesic Analysis

PGA was first introduced in by Fletcher *et al.* [36,37] and was formulated in a Lie-algebraic setting. We will discuss this formulation in Section 2.6, while here we focus on the Riemannian formulation, also by Fletcher *et al.* [38].

Let V denote a k-dimensional subspace of $T_\mu M$, $k < D$. Let $\mathrm{Exp}_\mu(V)$ denote the geodesic subspace generated from V through the Riemannian exponential map :

$$\mathrm{Exp}_\mu(V) = \left\{\mathrm{Exp}_\mu(x) : \ x \in V \subset T_\mu M\right\} \subset M \ .$$

If W is a subset of M, let $\pi_W : M \to W$ denote the projection of M on W; *i.e.*, if p is in M, then $\pi_W(p)$ is a point in W that minimizes the geodesic distance to p. The PGA problem is as follows:

$$\text{maximize} \ \ f(V) = \frac{1}{N}\sum_{i=1}^{N} d(\mu, \pi_{\mathrm{Exp}_\mu(V)}(p_i))^2 \tag{2.85}$$

$$\text{subject to:} \ \ V \text{ is a } k\text{-dimensional subspace of } T_\mu M \ . \tag{2.86}$$

This is a generalization of PCA: we are looking for the geodesic subspace that captures as much geodesic variance as possible. The main difficulty in solving this problem is related to the projection map. To make the problem more tractable, Fletcher *et al.* [38] replaced the problem with its linearized version.

Recently, Sommer *et al.* [136] have developed a framework for solving the original problem. They term the original problem *exact PGA*, while referring to the approach from Fletcher *et al.* [38] as linearized PGA. While this terminology makes sense, in the literature the method from Fletcher *et al.* [38] is widely known as PGA (*i.e.*, without the "linearized"). In this work, whenever we refer to PGA, we mean the linearized version.

Remark 2.5.1. The solution from [136] requires more effort than the one in [38]; in [135], Sommer *et al.* suggest indicators for determining if the expected gain in solving the harder problem is substantial enough to justify the effort.

Back to Fletcher *et al.* [38]. The original optimization problem is modified as follows. Instead of working on the geodesic variance, they worked on the Euclidean variance in $T_\mu M$.

The problem becomes a standard PCA (on zero-mean data) in the Euclidean tangent space:

$$\text{maximize} \quad f(V) = \frac{1}{N}\sum_{i=1}^{N}\left\|\text{Log}_{\mu}(p_i)\right\|_p^2 \tag{2.87}$$

$$\text{subject to:} \quad V \text{ is a } k\text{-dimensional subspace of } T_{\mu}M \ . \tag{2.88}$$

We refer the reader to [38] for the steps justifying this approach. Consequently, the algorithm for computing PGA is simple: First, compute the intrinsic mean μ. Then, express the data $\{p_i\}_{i=1}^N$ at $T_{\mu}M$ by setting $g_i = \text{Log}_{\mu}(p_i)$. Finally, do ordinary PCA on the vectors $\{g_i\}_{i=1}^N$. Let us denote the subspace found using this method by $V \subset T_{\mu}M$. As V is a linear subspace, we may regard it as a $D \times k$ matrix, satisfying $V^TV = I_{k\times k}$. Synthesis of a manifold point from the geodesic subspace $\text{Exp}_{\mu}(V)$ is done as follows. If $\alpha \in \mathbb{R}^{1\times k}$ is a set of coefficients, we generate a point $q \in M$ by:

$$q = \text{Exp}_{\mu}\left(V\alpha\right) \ . \tag{2.89}$$

2.6 Statistics on Matrix Lie groups

Throughout this section, let G be a $D-$dimensional real-valued connected matrix Lie group that is also a subgroup of GL(n), let $\{p_i\}_{i=1}^N$ denote the our G-valued data, and let $\mathfrak{g}$ denote the Lie algebra of G. As matrix Lie groups are also Riemannian manifolds, the Riemannian techniques from Section 2.5 apply transparently to them as well.

Before we proceed we need to address a notational issue. Recall that elements of $\mathfrak{g}$ are $n \times n$ matrices. However, the notations in Section 2.4.2 treat the tangent vectors as vectors. Thus, an expression such as

$$\text{Cov}(\{p_i\}_{i=1}^N) \overset{\text{by def.}}{=} \frac{1}{N-1}\sum_{i=1}^{N}\text{Log}_{\mu}(p_i)\text{Log}_{\mu}(p_i)^T \ , \tag{2.90}$$

may be a bit confusing, as the "T" symbol might be incorrectly parsed as transposing an

$n \times n$ matrix, while $\mathrm{Log}_\mu(p_i)\mathrm{Log}_\mu(p_i)^T$ might be parsed as the matrix product of two $n \times n$ matrices. In truth, however, we should understand this equation as

$$\mathrm{Cov}(\{p_i\}_{i=1}^N) \overset{\text{by def.}}{=} \frac{1}{N-1}\sum_{i=1}^{N}\left(\mathrm{Log}_\mu(p_i)\right)^\vee\left(\left(\mathrm{Log}_\mu(p_i)\right)^\vee\right)^T \ , \tag{2.91}$$

where we utilize the notation from Section 2.3.6.1. Thus, $\left(\mathrm{Log}_\mu(p_i)\right)^\vee$ is a column vector of length D^2 while $\left(\left(\mathrm{Log}_\mu(p_i)\right)^\vee\right)^T$ is its transpose. Consequently, the summands are $D \times D$ matrices, hence so is the covariance – meeting our expectations from a covariance of D-dimensional data. Likewise, when we want to create a $D \times N$ matrix containing our N data points as expressed at some tangent space T_pG, we should write $[\left(\mathrm{Log}_p(p_1)\right)^\vee, \ldots, \left(\mathrm{Log}_p(p_N)\right)^\vee]$ rather than $[\mathrm{Log}_p(p_1), \ldots, \mathrm{Log}_p(p_N)]$. Having said that, this notation gets tiresome fairly quickly. Thus, at the small risk of causing confusion, we usually avoid the "Vee"/"Hat" notation, and the decision whether a tangent vector is regarded as a vector or a matrix should be inferred from the context.

If we pick the Riemannian metric from Eqn. (2.54), The (sample) *Fréchet function* becomes

$$\mathrm{SSGD}(p) = \sum_{i=1}^{N}\left\|p\log(p^{-1}p_i)\right\|_F^2 \ . \tag{2.92}$$

In contrast, if we choose the distance from Eqn. (2.66), which may or may not be the geodesic distance associated with the Riemannian metric from Eqn. (2.58), then the (sample) *Fréchet function* becomes

$$\mathrm{SSGD}(p) = \sum_{i=1}^{N}\left\|\log(p^{-1}p_i)\right\|_F^2 \ . \tag{2.93}$$

By Eqn. (2.67), this can be approximated by

$$\mathrm{SSGD}(p) \approx \sum_{i=1}^{N}\|-\log(p)+\log(p_i)\|_F^2 \ . \tag{2.94}$$

2.6.1 PGA on Matrix Lie groups

2.6.1.1 Riemannian PGA

In the context of Lie groups, when the chosen Riemannian metric is given by Eqn. (2.54), we will use term Riemannian PGA for the method described in Section 2.5.3.2. In particular, note that the learned subspace is in $T_\mu G$.

2.6.1.2 Lie-Algebraic PGA

Now let us now switch to a more Lie-algebraic theoretic mindset. Recall that the matrix exponential is mapping linear subspaces of $\mathfrak{g}$ to matrix Lie subgroups of G. In fact, every connected matrix Lie subgroup of G can be built this way. In terms of Section 2.4.3.2, connected matrix Lie subgroups that are built from subspaces are geodesic subspaces (with respect to I, the identity of G). Now suppose that among all such subgroups that are of dimension D, we want to find one that is optimal in some sense. Instead of optimizing over this set of subgroups, we define our cost functions in terms of the subspaces that generated them.

Originally, Fletcher *et al*. [36,37] formulated their PGA over Lie groups and not for the general Riemannian setting. Rather than explicitly working in $T_\mu G$, they identified $T_\mu G$ with $T_I G$ (as explained in Section 2.3.7.1) and did the PCA there. Their method boiled down to the following steps:

1. Compute the interior mean μ;
2. Set $g_i = \log(\mu^{-1} p)$;
3. Compute PCA on $\{g_i\}_{i=1}^N$ to produce a $k-$dimensional subspace of $T_I G$, denoted by V.

Here too, as V is a subspace, we may regard it as a $D \times k$ matrix, satisfying $V^T V = I_{k\times k}$. The set

$$\{\exp(x) : \ x \in V \subset T_I G\}$$

is a k-dimensional matrix Lie group (and a subgroup of G). Synthesis of a manifold point (equivalently, a group element) is done as follows. If $\alpha \in \mathbb{R}^{1\times k}$ is a set of coefficients, we generate a point $q \in M$ by:

$$\mu \exp(V\alpha) \ . \tag{2.95}$$

That is, we *compose* μ with the exponent of a linear combination of vectors in the Lie algebra. Note that if G is not Abelian, this is not the same as thing as $\exp(\log \mu + V\alpha)$.

When $||\log(p^{-1}q)||$ is indeed a geodesic distance implied by the Riemannian metric in Eqn. (2.58), then this method can be justified on a Riemannian basis. The learned basis vectors are in $T_I G$ and not $T_\mu G$, but by the construction of the this Riemannian metric, the left-translation map $T_I G \to T_\mu G$, $x \mapsto p^x$ preserves inner-products and distances.

However, when $||\log(p^{-1}q)||$ is not a true geodesic distance, this method can be justified in terms of distances, but these are not geodesic (also known as Riemannian) distances. In particular, if x is a basis vector, then the one-dimensional geodesic subspace, $p\exp(tx)$, is not a geodesic curve.

In either way, we will refer to this kind of PGA, *i.e.*, PGA in the Lie algebra, as Lie-algebraic PGA.

2.6.2 Lie-Algebraic PGA vs Riemannian PGA: An Optical Flow Example

We here demonstrate that the two models, the Lie-algebraic PGA and Riemannian PGA, can lead to rather different geodesic subspaces. In fact, with several notable exceptions (*e.g.*, on SO(3)) these subspaces usually differ.

We here show a simple case, using a Lie group of 2D image deformations, where the resulting 1D geodesic subspaces are noticeably different. This is illustrated in Figs. 2.6 and 2.7. Recall that $\exp : T_I M \to M$ denotes the matrix exponential (in effect, the Lie group exponential) while $\mathrm{Exp}_\mu : T_\mu M \to M$ denotes the Riemannian exponential map with respect to the point of tangency μ. The left panels of Fig. 2.7 show the first eigenvector of the Lie-algebraic model (denoted by $v_1 \in T_I M$), visualized by $\mu \exp(+v_1)$ (top) and $\mu \exp(-v_1)$ (bottom) acting on the template. Similarly, the right panels of Fig. 2.7 show the first eigenvector of the Riemannian model (denoted by $u_1 \in T_\mu M$), visualized by $\mathrm{Exp}_\mu(+u_1) \overset{\text{by def.}}{=} \mu \exp(\mu^{-1} u_1)$ (top) and $\mathrm{Exp}_\mu(-u_1) \overset{\text{by def.}}{=} \mu \exp(-\mu^{-1} u_1)$ (bottom) acting on the template.

We stress that the fact that in this particular example the 1D Riemannian model captured horizontal scaling while the 1D Lie model captured shearing is coincidental. It is very easy to come up with different data where things would have been exactly the opposite. Our only point here is that the two models are usually different.

While we avoid making any claim about which approach is better in general – as this may depend on the application – we argue that practitioners should be well aware of this delicate point. For example, it is quite possible that the Riemannian approach may result in better optical flow models than the Lie algebraic approach as it is tied more strongly to distances. Specifically for the CT framework we presented in Chapter 5, the Riemannian PGA is preferred for the reasons mentioned in that chapter.

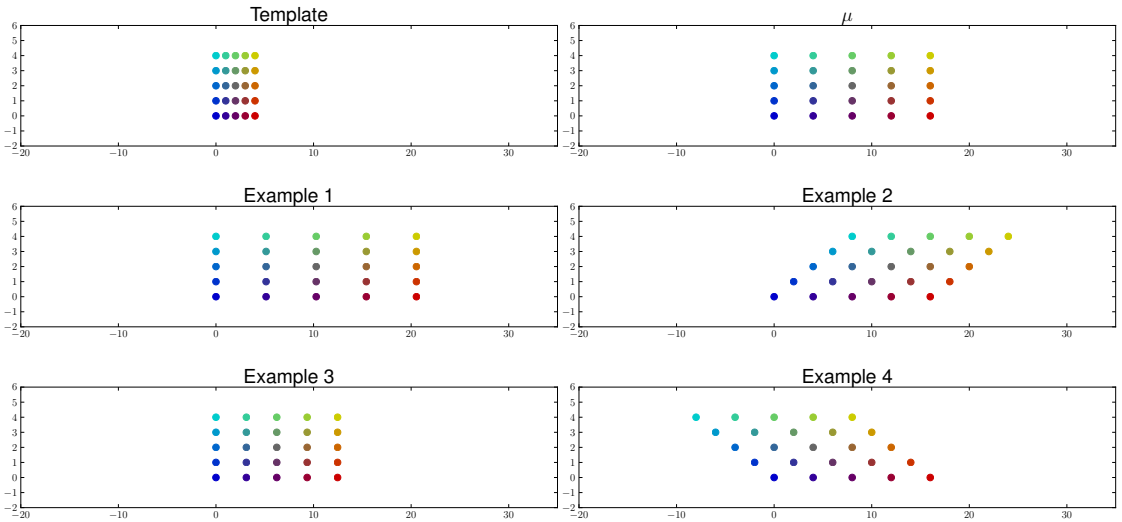


Figure 2.6: Template, intrinsic mean (denoted by μ), and the four examples from which the intrinsic mean was computed. The colors indicate point-to-point correspondences. Note the scales of the figures.

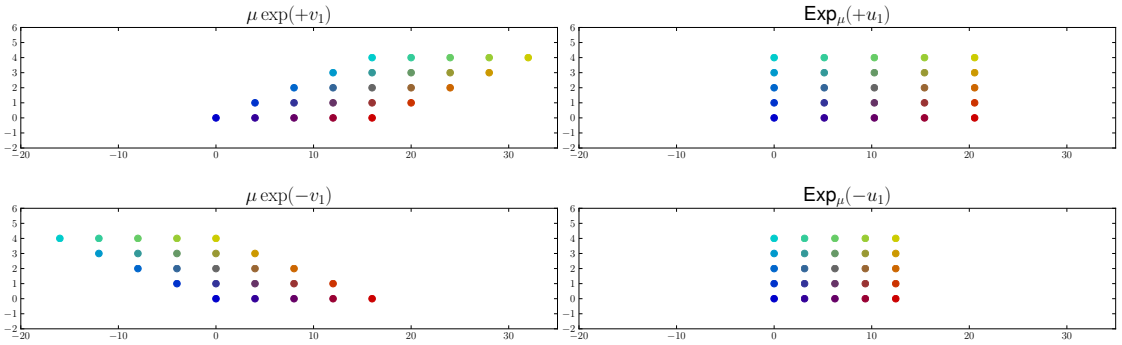


Figure 2.7: PGA models. Left: the Lie-Algebraic 1D model. Right: the Riemannian model. See text for more details.

Chapter 3

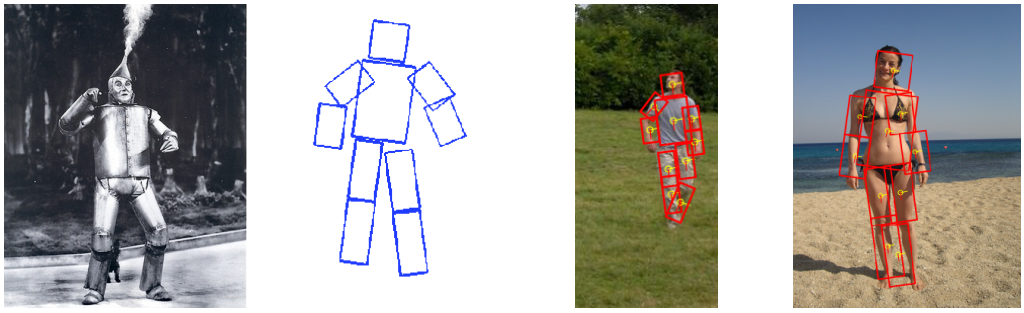
Contour People

In the last few decades, the computer vision community has put a lot of effort into developing methods for solving the task of pose estimation of people from images. In this task, a 2D model of articulated human shape is a central building block. As was mentioned in Section 1.1.2, traditional models are based on a simple shape representation comprised of crude geometrical shapes (*e.g.*, rectangles or trapezoids) describing each of the individual body parts; see Fig. 3.1b for an illustration of a typical model. We will refer to this kind of shape representation as Pictorial Structures (PS), using the term from Fischler and Elschlager's seminal work [32].

In a PS representation, each single body part is only allowed to rigidly deform[1], although different parts may undergo different rigid transformations so articulation can still be achieved.

Remark 3.0.1 (Shape Representation *vs*. Statistical Shape Model). Throughout this we chapter will maintain a distinction between the terms "shape representation" (or, for short, "representation") and "statistical shape model" (or "model"). A statistical model, of either shape or other entities, is always defined with respect to some particular representation. In the context of the current chapter, a statistical model of (human) shape must be defined

[1] Some variants of PS allow for more general transformations such as affine or projective.



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

Figure 3.1: Pictorial structures are for the heartless (this particular choice of words carries the risk of someone saying that detailed shape models are for the brainless, but we shan't be cowards): with some notable exceptions (a), simple polygonal shapes (b) provide a poor explanation for human shape (c-d).

with respect to some shape representation. For example, when we refer to a *PS representation*, we mean that the human shape is represented as described in the previous paragraph. A *PS model* is a statistical distribution over the configurations of the body parts in a PS representation, and, perhaps, over the crude shape of these parts (*e.g.*, the length or width of a part). As it is reasonable to expect some relations between the different body parts, a typical PS model is based on a *probabilistic graphical model*[2] that expresses these relations through pairwise potentials capturing the interactions between adjacent parts (*e.g.*, an upper arm and a lower arm). These potentials are typified by "springs", meaning it is unlikely the parts will be too far from each other. Higher-order cliques may be defined as well, but, as we will see shortly, this fact is orthogonal to our discussion. Finally, we will often use the terms "2D model" and "3D model" as shorthands for "statistical model of 2D shape" and "statistical model of 3D shape", respectively.

The main motivations for PS representations are their simplicity and low dimensionality. Consequently, computer vision researchers and practitioners working on pose estimation have taken for granted that PS should be the weapon of choice for shape representation and usually focus on improving the PS statistical models or inference with such models.

There is, however, a major limitation to the PS approach for shape representation:

[2]See [84] for a modern textbook on probabilistic graphical models.

crude geometrical shapes provide a poor way to describe a human shape, although some exceptions may come to mind; see Fig. 3.1a. In other words, PS representations lack realism in the sense they do not account for the modeling of a *detailed shape*.

This means that models learned for a PS representation cannot provide us with a good generative model to "explain away" image evidence. Thus, plenty of geometric image evidence – such as the exact outline of the person's body – is being discarded, leading to image likelihood models of needlessly limited power. Consequently, inference in PS models depends heavily on their appearance[3] model component and/or the pose prior while information about the actual real-world shape of the person can contribute very little, if any, to the process.

Given this shortcoming, and in sharp contrast to the amount of progress the community has made on advancing statistical models and inference methods associated with PS representations, it is striking to realize how little the shape representations of articulated 2D models have changed throughout the last four decades since Fischler and Elschlager's work. We thus seek both a better representation and a better generative model of articulated human shape. The representation should be able to capture detailed shape and the model should capture the shape variation. Together they will not only yield more accurate pose estimates but also provide a way to move much of the inferential work into the shape component of the image likelihood model. For example, we should be able to compare observed image edges against the curves of an hypothesized articulated shape drawn from the model – an approach fitting well into the *analysis-by-synthesis* paradigm.

Importantly, in addition to improving pose estimation, there are other applications for a model of articulated detailed shape. For instance, it can help in the following computer vision tasks:

1. segmenting a person from an image;

[3]Appearance here means the color-intensity image patten or its derived features.

2. person recognition;
3. classification (*e.g.*, of gender);
4. tracking (*e.g.*, if the model factors body shape from pose, we can keep the estimate of the body shape fixed across frames while allowing the pose to vary);
5. estimating optical flow generated by the motion of people (*e.g.*, see [154]);
6. activity recognition (*e.g.*, see [75]).

The obvious question is: how can we achieve the goal of building better models of two-dimensional articulated detailed human shape? Our short answer is that the we can take advantage of the tremendous amount of progress that has been made for models of *three-dimensional* articulated detailed human shape.

These 3D models were first introduced in the computer graphics community, where high realism is one of the primary objectives. They are learned from 3D laser scans of multiple human bodies. Thus, they capture variability across a population of real human shapes. From these 3D models, to be discussed at greater length in Chapter 4, we can generate a training dataset of 2D contours by first drawing 3D shapes from the 3D model, and then projecting them to the image plane. Then, we can take the training contours and learn a model of 2D detailed shape.

The longer answer has three components

1. how to generate the training 2D shape data from a given 3D model;
2. how to represent the 2D shape;
3. how to model the statistical variation of the 2D shapes.

These components will be discussed later in this chapter. Together they enable us to define a new model of 2D articulated human shape that provides a happy medium between the

traditional simple-but-crude 2D models and computationally-demanding-but-detailed 3D models (as was illustrated earlier in Fig. 1.2). We call our model the *Contour Person* (CP) model.

The CP model factors changes in 2D human shape into a number of causes, with each cause represented via a low-dimensional parametric model. These include shape changes due to

1. viewing direction;
2. body shape (*i.e.*, the "physique" of the person);
3. rigid articulation;
4. non-rigid deformation due to articulation.

This model is similar to recent work on 3D body shape representation using the SCAPE model [6]. In fact our 2D model is created from a 3D SCAPE model of the human body. However, rather than model deformations of triangles on a 3D mesh, we model deformations of line segments in 2D; this results in a simpler and lower-dimensional body model. Additionally, a successful application of SCAPE in computer vision application is usually limited to a setting where the calibration of the camera (or cameras) is known [15, 16], we explicitly model statistics of 2D shape deformations due to camera variation and so our model is designed for the more challenging setting of uncalibrated monocular-view images.

We envision many applications of the CP model and its associated shape representation. This chapter focuses on the development of the CP model and the issues involved with representing an inherently 3D shape in 2D while maintaining realism and accuracy. To illustrate the application of the model we present initial results in pose estimation and segmentation. To do so, we build on an existing state of the art person detector that uses a PS representation [4]. This existing technique is used to initialize our model and then both the pose and shape of the CP model are refined using a parametric form of GrabCut [124].

Results of pose estimation and segmentation are shown on a variety of images and compared with PS for pose estimation and with traditional GrabCut for segmentation.

To summarize, in this chapter we define a new representation of 2D shape on which we build a statistical model that has an expressive power akin to that of a detailed 3D model and the computational benefits associated with a simple 2D part-based model. This self-coined *contour person* (CP) model is learned from a 3D model of the human body that captures natural shape and pose variations; the projected contours of this model form the training set. Importantly, the CP model factors deformations of the body into three components: shape variation, camera change, and part rotation. This model also incorporates a learned non-rigid deformation model. The result is a 2D articulated model that is compact to represent, simple to compute with and more expressive than previous models. We demonstrate the value of such a model in 2D pose estimation and segmentation. Given an initial pose from a standard PS method, we refine the pose and shape using an objective function that segments the scene into foreground and background regions. The result is a parametric, human-specific, image segmentation.

3.1 Previous Work

Two-dimensional models of the human body are popular due to their representational and computational simplicity. Existing models include articulated PS models, active shape models (or point distribution models), parametrized non-rigid templates, and silhouette models.

We focus here on models that explicitly represent shape with contours and, furthermore, those that have been used to represent non-rigid human shape and pose. There is an extensive literature on general contour models for object representation and recognition that we do not consider here.

3.1.1 2D Articulated Person Models

Most 2D articulated person models have focused on estimating human pose and have ignored body shape. We argue that a good body shape representation can improve pose estimation by improving the fitting of the model to image evidence.

The first use of a human "puppet" model was due to Hinton [71] and there have been many related models since. The classic 2D model is the "cardboard person" [76], defined by a kinematic tree of polygonal regions, where each limb may be rotated or scaled in 2D. Similarly the scaled-prismatic model (SPM) treats the limbs as rigid templates that can be scaled in length [18]. Both the cardboard person and SPM approximate foreshortening caused by motion of the limbs in depth.

More restricted models, with only rotation at the joints (and a global scale), form the basis of most of the current PS models [32] used for detecting and tracking people in monocular imagery [4, 30, 31, 86, 119]. These models admit efficient search with belief propagation (BP) due to the simplification of the representation. Sigal and Black [133] use a 2D model that includes foreshortening and do inference with BP. The advantage of the richer model is that it allows better prediction of 3D pose from the estimated 2D model [133].

Our work is related to the PS camp but increases the realism beyond previous methods by modeling shape variation across bodies as well as non-rigid deformation due to articulated pose changes.

3.1.2 Active Shape and Contour Models

Active shape models (ASMs) capture the statistics of contour deformations from a mean shape using principal component analysis (PCA) [22]. PCA can be performed on points, control points or spline parameters. These models have been used extensively, particularly

for representing human faces and their deformations [22]. Note that facial features deform in such models but there is no explicit representation of part rotation, they have little depth variation relative to each other, and there is no self occlusion. The articulated human body has all these issues, compromising the applicability of ASMs

Baumberg and Hogg were the first to use ASMs for representing the full human body [8]. Given a training set of pedestrians segmented from the background, they define contours around each with the same number of points and roughly the same starting locations. They analyze the modes of variation in this contour using PCA and use this model to track pedestrians. In such a model, changes in body shape and pose are combined in one PCA representation. Furthermore, with no notion of body parts, the alignment between training body contours is difficult to establish. This results in principal components that capture the non-informative sliding of points along the contour. Finally this simple PCA model does not directly encode articulated body pose, limiting its use for human motion and gesture analysis. In contrast, our framework eliminates the misalignment problem and factors deformations due to shape and pose (and camera variation).

Related to the model from [8] are eigen-points models by Covell and Bregler [23, 24]. These capture variability in a set of 2D landmarks using a single PCA model. As in [8], these models do not lend themselves to factorization of different sources of deformations.

Ong and Gong [112] extend these point distribution models to deal with articulated 3D human motion of the upper body. They construct a training vector that includes the contour points of the upper body, 2D points corresponding to the locations of the hands and head, and the 3D joint angles of an underlying articulated body model. To deal with the nonlinearity of the contour with respect to pose, they use a hierarchical PCA method that finds linear clusters in the nonlinear space. In contrast, we explicitly model the parts of the body and do not use PCA to capture articulations. Rather we use use it to capture body shape variations (and camera view changes). This provides a blend between the PS models and the active contour methods.

Grauman *et al*. [50] map multi-view silhouettes to contours and learn a low dimensional shape representation in conjunction with 3D body pose. Like other methods they model shape in terms of the contour points. Our model differs in that it models *deformations* of 2D contours and this representation is important for explicitly modeling articulation and for factoring different types of deformation.

3.1.3 Human Models and Segmentation

We test our model on the problem of segmentation; the contour of the body defines the region inside (and outside) the body.

In early work, Kervrann and Heitz [81] define a non-rigid model of the hand and estimate both its pose and segmentation using motion and edge cues. The model is not part-based, the deformations are not learned, and it has a limited range of motion. Alternative formulations have explored template-based models of the body [46, 95] that are not fully articulated and do not factor shape and pose.

Of particular relevance is the recent work of Ferrari and Zisserman [31] that uses a weak detector to obtain a crude estimate of human pose in an image. This pose is then used to initialize GrabCut [124] segmentation. Given an initial segmentation of the scene into a foreground person and a background, they fit a more detailed PS body model.

We use this idea of an initial guess followed by GrabCut but with a much more detailed model. Rather than end with a PS model, we begin with one. We use the method in [4] to fit a PS body model to the image. This 2D body model is used to initialize the pose and scale of our contour-person model. We then refine the parameters of the model (pose, view and shape) to improve the segmentation using a form of parametric GrabCut.

This parametric GrabCut idea is similar to the PoseCut framework suggested by Bray *et al*. [12]; however, they use a 3D articulated skeleton model and a distance transform

from it to define 2D body shape. The result is a crude depiction of the body shape in 2D but the interesting element of their work is the integration of 3D pose estimation with segmentation. We also integrate parametric body shape and pose estimation with segmentation but do it in 2D with a much richer model of body shape.

3.2 Shape Representation

Throughout this work we adopt a pattern-theoretic approach for shape deformation. In particular, our representation of 2D shapes is based on an existing shape representation suggested by Grenander [51] that was further studied by Grenander and Miller [53]; however, we modify it in a way that has many important practical advantages.

Any discretized contour $\mathcal{C}$ can be represented by a polygon of N_{pts} points, denoted $\{v_i\}_{i=1}^{N_{pts}}$, $v_i = (x_i, y_i)$. The shape representation, however, is not based on contours. Rather, it is based on *transformations acting on a reference contour*. This reference contour is called the *template* and is denoted by $\mathcal{T}$. The core of this shape representation is a local deformation acting on a directed line segmented connecting two adjacent points along the polygon.

3.2.1 Directed Line Segments

We regard $\mathcal{C}$ as a column vector of length $2N_{pts}$ that has the following form:

$$\mathcal{C} = [x_1, y_1, x_2, y_2, \ldots, x_{N_{pts}}, y_{N_{pts}}]^T \in \mathbb{R}^{2N_{pts}} \ . \tag{3.1}$$

Similarly, we can write the template $\mathcal{T}$ as

$$\mathcal{T} = [x_1^{\text{ref}}, y_1^{\text{ref}}, x_2^{\text{ref}}, y_2^{\text{ref}}, \ldots, x_{N_{pts}}^{\text{ref}}, y_{N_{pts}}^{\text{ref}}]^T \in \mathbb{R}^{2N_{pts}} \ , \tag{3.2}$$

where $\left\{v_i^{\text{ref}}\right\}_{i=1}^{N_{pts}}$, $v_i^{\text{ref}} = (x_i^{\text{ref}}, y_i^{\text{ref}})$, represent the points in $\mathcal{T}$.

The contours of interest to us are closed. Let $\{e_i\}_{i=1}^{N_{pts}}$ and $\{l_i\}_{i=1}^{N_{pts}}$ denote the directed line segments of $\mathcal{C}$ and $\mathcal{T}$ respectively, where

$$e_i = \begin{cases} v_{i+1} - v_i & , \text{ if } \quad i \in \{1, 2, \ldots, N_{pts} - 1\} \\ v_1 - v_i & , \text{ if } \quad i = N_{pts} \end{cases} \tag{3.3}$$

$$l_i = \begin{cases} v_{i+1}^{\text{ref}} - v_i^{\text{ref}} & , \text{ if } \quad i \in \{1, 2, \ldots, N_{pts} - 1\} \\ v_1^{\text{ref}} - v_i^{\text{ref}} & , \text{ if } \quad i = N_{pts} \end{cases} . \tag{3.4}$$

We will always assume that e_i and l_i have nonzero length[4]: $\|e_i\|_{\ell^2} > 0$, $\|l_i\|_{\ell^2} > 0$.

Let $E : \mathbb{R}^{2N_{pts}} \to \mathbb{R}^{2N_{pts}}$ be the linear map that maps C to $[e_1^T, e_2^T, \ldots, e_{N_{pts}}^T]^T$. E can be represented by a sparse matrix which takes values in $\{-1, 0, 1\}$.

Example 3.2.1 (Constructing E for a closed contour)**.** Let $N_{pts} = 4$. Then E has form as in the following equation:

$$\begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{pmatrix} = \begin{pmatrix} v_2 - v_1 \\ v_3 - v_2 \\ v_4 - v_3 \\ v_1 - v_4 \end{pmatrix} = \begin{pmatrix} x_2 - x_1 \\ y_2 - y_1 \\ x_3 - x_2 \\ y_3 - y_2 \\ x_4 - x_3 \\ y_4 - y_3 \\ x_1 - x_4 \\ y_1 - y_4 \end{pmatrix} = \overbrace{\begin{pmatrix} -1 & 0 & +1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & +1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & +1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & +1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & +1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & +1 \\ +1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & +1 & 0 & 0 & 0 & 0 & 0 & -1 \end{pmatrix}}^{E} \overbrace{\begin{pmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \\ x_4 \\ y_4 \end{pmatrix}}^{\mathcal{C}, \text{ by def.}} . \tag{3.5}$$

The generalization from Example 3.2.1 to other values of N_{pts} is straightforward. Note that E does not depend on the value of $\mathcal{C}$, and we so use the same E for every contour of

[4]Note this is a weaker assumption than assuming that all of the points in $\mathcal{C}$ (or $\mathcal{T}$) are distinct.

N_{pts} – including the template $\mathcal{T}$.

3.2.2 Local Shape Deformations

In pattern-theoretic terminology [52], the l_i's are called *primitives* upon which local transformations can act. In our case, e_i is seen as the result of a transformation of *scaled rotation*, denoted by Q_i, acting on l_i. Scaled rotations are also called *similitudes*, and together they form a matrix Lie subgroup of GL(2).

Definition 3.2.1 (The Similitude group of order 2)**.** The Abelian two-dimensional matrix Lie group, given by

$$\mathrm{Sim}(2) = \left\{ Q : Q = SR,\ (R,S) \in \mathrm{SO}(2) \times \mathbb{R}^+ \right\} , \tag{3.6}$$

is called the *Similitude group (of order 2)*. Equivalently, Sim(2) is given by

$$\mathrm{Sim}(2) = \left\{ \begin{pmatrix} A & -B \\ B & A \end{pmatrix} ,\ A, B \in \mathbb{R}, A^2 + B^2 > 0 \right\} . \tag{3.7}$$

We may think of Q_i as a *local deformation*. The word "local" does not mean that the deformation is small. Rather, it means that Q_i pertains to a particular directed line segment e_i and not to the entire contour. As indicated by Definition 3.2.1, Q_i is fully defined by an angle θ_i and scale S_i, or equivalently, by $(S_i \cos\theta_i,\ S_i \sin\theta_i)$:

$$Q_i = S_i \begin{pmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{pmatrix} = \begin{pmatrix} S_i\cos\theta_i & -S_i\sin\theta_i \\ S_i\sin\theta_i & S_i\cos\theta_i \end{pmatrix} . \tag{3.8}$$

Given l_i and Q_i we can create a new e_i using the *directed-line-segment synthesis* equation:

$$e_i = Q_i l_i . \tag{3.9}$$

Conversely, given l_i and e_i we first rewrite $Q_i l_i$ as

$$Q_i l_i = \begin{pmatrix} S_i \cos\theta_i & -S_i \sin\theta_i \\ S_i \sin\theta_i & S_i \cos\theta_i \end{pmatrix} l_i = \begin{pmatrix} l_{i,1} & -l_{i,2} \\ l_{i,2} & l_{i,1} \end{pmatrix} \begin{pmatrix} S_i \cos\theta_i \\ S_i \sin\theta_i \end{pmatrix}, \tag{3.10}$$

where $l_{i,k}$ is the k^{th} element of l_i, $k \in \{1, 2\}$. We then extract Q_i by solving the invertible linear system

$$e_i = \begin{pmatrix} l_{i,1} & -l_{i,2} \\ l_{i,2} & l_{i,1} \end{pmatrix} \begin{pmatrix} S_i \cos\theta_i \\ S_i \sin\theta_i \end{pmatrix}. \tag{3.11}$$

The system is invertible as $l_{i,1}^2 + l_{i,2}^2 = \|l_i\|_{\ell^2}^2 > 0$. Equation (3.11) is our *(local) deformation analysis* equation.

3.2.3 Global Shape Deformation: Open Contours

Suppose, for now, that we remove the connection between the first and last contour points in $\mathcal{C}$. The result is an open polygon, denoted by $\mathcal{C}^{\text{open}}$, with the same N_{pts} points as $\mathcal{C}$ but with only $N_{pts} - 1$ directed line segments: $\{e_i\}_{i=1}^{N_{pts}-1}$. An open template polygon, $\mathcal{T}^{\text{open}}$ is defined in a similar way.

Up to a global translation, we can always recover $\mathcal{C}^{\text{open}}$ from $\mathcal{T}^{\text{open}}$ and $\{Q_i\}_{i=1}^{N_{pts}-1}$:

$$v_i = \begin{cases} (0,0) & , \text{ if } \quad i = 1 \\ v_{i-1} + Q_{i-1} l_{i-1} & , \text{ if } \quad i \in \{2, 3, \ldots, N_{pts}\} \end{cases}. \tag{3.12}$$

Equation (3.12) is our *open contour synthesis* equation.

Like Sim(2), the $\text{Sim}(2)^{N_{pts}-1}$ group (namely, the direct product of $N_{pts} - 1$ copies of Sim(2)) is also an Abelian Lie group. Its dimension is $2(N_{pts} - 1)$. Let $\mathcal{Q}^{\text{open}} = (Q_1, Q_2, .., Q_{N_{pts}-1}) \in \text{Sim}(2)^{N_{pts}-1}$. Since we can *identify* $\mathcal{Q}^{open}$ *with a sparse block-*

diagonal matrix,

$$\begin{pmatrix} Q_1 & & & 0 \\ & Q_2 & & \\ & & \ddots & \\ 0 & & & Q_{N_{pts}-1} \end{pmatrix} \in \mathrm{GL}(2N_{pts}-2)\ , \tag{3.13}$$

we may regard $\mathrm{Sim}(2)^{N_{pts}-1}$ as a matrix Lie group.

This means we can apply the tools we saw in Chapter 2; *e.g.*, we have a natural way to compose deformations, we can define a distance between different $\mathcal{C}$'s as the geodesic distance between their corresponding $\mathcal{Q}^{\mathrm{open}}$'s on the deformation manifold (*i.e.*, on $\mathrm{Sim}(2)^{N_{pts}-1}$), and we can apply tools from statistics on manifolds.

This type of pattern analysis (and pattern synthesis – enabled by Eqn. (3.12)) of 2D contours goes back to the early days of Grenander's pattern theory in the 70's, and was later thoroughly explored by Grenander and Miller [53]; see also [52, 55].

3.2.4 Global Shape Deformation: Closed Contours

In the case of open contours, Eqn. (3.12) ensures us that for *any* $\mathcal{Q}^{\mathrm{open}} \in \mathrm{Sim}(2)^{N_{pts}-1}$, we can always build a corresponding open contour $\mathcal{C}^{\mathrm{open}}$. We emphasize the word "any" as this is true even if we never observed this $\mathcal{C}^{\mathrm{open}}$, and even if $\mathcal{Q}^{\mathrm{open}}$ was sampled at random from some arbitrary probability distribution on the $\mathrm{Sim}(2)^{N_{pts}-1}$ manifold.

The contours that are of interest to us, however, are closed ones. Naturally, the corresponding Lie group here is $\mathrm{Sim}(2)^{N_{pts}}$, and if $\mathcal{Q} = (Q_1, Q_2, .., Q_{N_{pts}}) \in \mathrm{Sim}(2)^{N_{pts}}$, then

we identify $\mathcal{Q}$ with a sparse block-diagonal matrix,

$$\begin{pmatrix} Q_1 & & & 0 \\ & Q_2 & & \\ & & \ddots & \\ 0 & & & Q_{N_{pts}} \end{pmatrix} \in \mathrm{GL}(2N_{pts}) \; . \tag{3.14}$$

Again, through this identification, we regard $\mathrm{Sim}(2)^{N_{pts}}$ as a matrix Lie group.

For pattern analysis of closed contours, the discussion carries over transparently from the case of open contours. But what about pattern synthesis? It turns out things do not quite work the way we would like them to. Let us see why.

It is tempting to simply replace Eqn. (3.12) with

$$v_i = \begin{cases} v_{N_{pts}} + Q_{N_{pts}} l_{N_{pts}} & , \text{ if } \quad i = 1 \\ v_{i-1} + Q_{i-1} l_{i-1} & , \text{ if } \quad i \in \{2, 3, \ldots, N_{pts}\} \end{cases} \; . \tag{3.15}$$

This approach, however, does not work as Eqn. (3.15) is inconsistent: there is no guarantee that the equality

$$v_1 - v_{N_{pts}} = Q_{N_{pts}} l_{N_{pts}} \tag{3.16}$$

will hold. Alternatively, defining the synthesized contour by

$$v_i = \begin{cases} (0,0) & i = 1 \\ v_{i-1} + Q_{i-1} l_{i-1} & , \text{ if } \quad i \in \{2, 3, \ldots, N_{pts}\} \end{cases} \; . \tag{3.17}$$

is also not very useful: the resulting contour is unlikely to be the one we would expect as usually Eqn. (3.16) will not be satisfied. In other words, we would get *some* contour $\mathcal{C}$, but once we compute its deformation from $\mathcal{T}$, denoted by $\mathcal{Q}_{\mathcal{C}}^{\text{extracted}}$, *this deformation might be far from* $\mathcal{Q}$. In fact, it is easy to construct $\mathcal{Q}$'s such that their corresponding $\mathcal{Q}_{\mathcal{C}}^{\text{extracted}}$'s would be arbitrary far from them.

The next tempting idea would be to think that if we set N_{pts} to a number high enough, we might as well ignore the $l_{N_{pts}}$, treat the contour as open, and hope that since the resolution is high enough, the contour will be essentially closed. Unfortunately, this proves to be a particularly bad idea when statistical modeling is involved. When synthesizing contours from the deformations extracted from the training data, everything works just fine. However, upon leaning a statistical model from these deformations and sampling a random deformation from the model, the resulting synthesized contour usually has a very significant gap at its ends which makes the resulting shape look unnatural.

A more plausible idea, at least conceptually, would be to restrict ourselves to $M^{\text{closed-contours}}$, defined as the subset of $\text{Sim}(2)^{N_{pts}}$ that consists of exactly those $\mathcal{Q}$'s that, upon the application of Eqn. (3.15), satisfy the constraint from Eqn. (3.16). This is indeed possible, but, as observed in [53], will result in losing the group structure so we will not be able to benefit from the advantages associated with matrix Lie groups.

We here suggest an alternative approach, which is also computationally simpler than enforcing the constraint. We choose to keep working with the whole of $\text{Sim}(2)^{N_{pts}}$ (so we preserve its matrix Lie group structure) but instead of synthesizing a contour using Eqn. (3.15), we do it in a different way that enables us to ensure that $\mathcal{Q}_{\mathcal{C}}^{\text{extracted}}$ will be as close as possible to $\mathcal{Q}$.

To synthesize a contour we solve the following least-squares (LS) problem:

$$\begin{aligned} &\text{minimize} \;\; f(\mathcal{C}) = \|E\mathcal{C} - \mathcal{Q}E\mathcal{T}\|_{\ell^2}^2 = \sum_{i=1}^{N_{pts}} \|e_i - Q_i l_i\|_{\ell^2}^2 \\ &\text{subject to} \;\; \mathcal{C} \in \mathbb{R}^{2N_{pts}} \; . \end{aligned} \tag{3.18}$$

With f so defined, we regard

$$\mathcal{C} = \underset{x \in \mathbb{R}^{2N_{pts}}}{\arg\min}\, f(x) \tag{3.19}$$

as our *closed contour synthesis* equation. As before, we let $\mathcal{Q}_{\mathcal{C}}^{\text{extracted}}$ denote the deformation extracted from this minimizer. Several remarks are in order:

1. The minimizer exists but is unique only up to a global 2D translation of the curve. Consequently, and without loss of generality, we may assume that the first two entries in $\mathcal{C}$ (that is, by definition, (x_1, y_1)) are zero. Consequently the optimization problem becomes a convex Least-Squares (LS) problem where the domain is $\mathbb{R}^{2N_{pts}-2}$ – the space of curves whose first point is the origin.

2. The interpretation of the quantity being minimized is the difference between the line segments we want (for a nominal value of $\mathcal{Q}$) and the line segments we can actually have in the deformed contour.

3. In terms of graphs, the connectivity of these curves – captured by the connectivity matrix E – is fixed, so the minimizer is always a closed contour (although it may have self-intersections): regardless what the value of the minimizer is, we always connect its points in the same order.

4. The difference between $\mathcal{Q}$ and $\mathcal{Q}_{\mathcal{C}}^{\text{extracted}}$ is distributed over all of the Q_i's, not just over $Q_{N_{pts}}$; in practice this is translated into more regular curves.

5. If Q was computed from some contour $\mathcal{C} = [x_1, y_1, x_2, y_2, \ldots, x_{N_{pts}}, y_{N_{pts}}]^T$, then

$$[x_1 - x_1, y_1 - y_1, x_2 - x_1, y_2 - y_1, \ldots, x_{N_{pts}} - x_1, y_{N_{pts}} - y_1]^T$$

   is the unique minimizer (modulo global translation), $\mathcal{Q}_{\mathcal{C}}^{\text{extracted}} = \mathcal{Q}$, and f is nullified.

6. In our experiments, $\mathcal{Q}$ is sampled from a statistical model (to be discussed later), so the minimal value of f might be (and usually is) positive. The good news, however, is that these values are usually very small. For example, $\frac{f(\mathcal{C})}{N_{pts}}$ is usually much smaller than $\frac{\sum_{i=1}^{N_{pts}} \|e_i\|_{\ell 2}^2}{N_{pts}}$, the average square length of a line segment in $\mathcal{C}$. We attribute that to the effectiveness of the statistical model, meaning that the learned distribution is concentrated on regions of the manifold that are not too far from $M^{\text{closed-contours}}$. Also, these regions have some overlap with $M^{\text{closed-contours}}$ as all the $\mathcal{Q}$'s computed from the training contours are, by definition, in $M^{\text{closed-contours}}$.

7. E is singular, sparse and fixed. If N_{pts} is small enough, then it is enough to pre-

compute $E^\dagger$, the Moore-Penrose pseudoinverse of E. The minimizer is then given by $\mathcal{C} = E^\dagger \mathcal{Q} E \mathcal{T}$. Otherwise, a sparse solver can be pre-computed. In our experiments we have used $E^\dagger$. In Chapter 4 we will see an analogous problem where the high dimensionality prohibits the use of $E^\dagger$ and a sparse solver is used instead.

8. The minimization is done over curves, not over deformations. This is not the same as minimizing the Euclidean difference between $\mathcal{Q}_{\mathcal{C}}^{\text{extracted}}$ and $\mathcal{Q}$ subject to both $\mathcal{Q}_{\mathcal{C}}^{\text{extracted}} \in \text{Sim}(2)^{N_{pts}}$ and the constraint from Eqn. (3.16) (for one thing, the minimization problem in Eqn. (3.18) is much easier). It is possible to use tools from optimization over matrix manifolds[5] to solve an analogous (non-convex) problem in terms of minimizing the geodesic distance between $\mathcal{Q}_{\mathcal{C}}^{\text{extracted}}$ and $\mathcal{Q}$ (with the same constraints). This would be a conceptually better approach then our LS approach since it will tie the resulting $\mathcal{C}$ more closely to the objects that are being statistically modeled (*i.e.*, the $\mathcal{Q}$'s); however, the optimization problem would be much harder (and significantly slower) to solve, and the solution procedure may get stuck in a local minimum. In light of this trade-off we resort to our LS approach which is easy and works well in practice.

9. This LS approach for contour synthesis is similar to approaches used successfully for the harder problem of synthesis of 3D triangular meshes from deformations of triangles [6, 16, 142]. In fact, in that aspect, we were inspired by these works. Unlike in these works, however, our motivation is not limited to finding a solution to the problem of creating a shape (a 2D contour in our case, a 3D mesh in those works) from deformations, but also pertains to our desire to *preserve the matrix Lie group structure*. We will return to the topic of triangle deformations in Chapter 4.

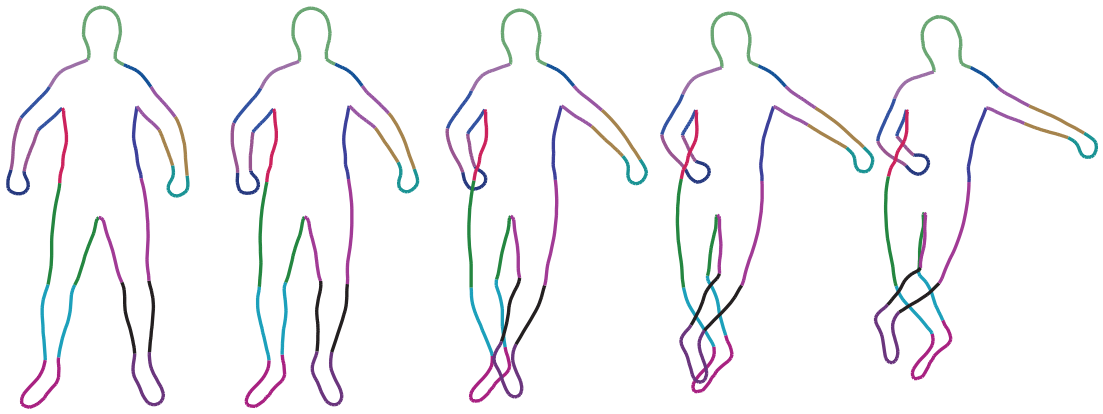


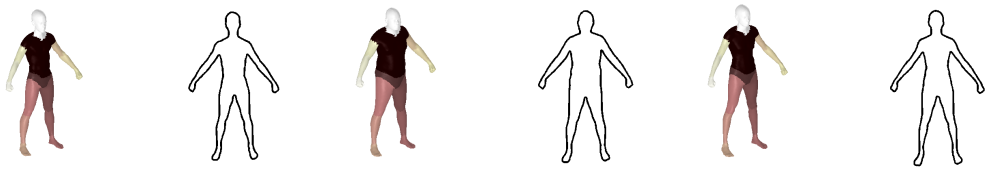
Figure 3.2: The Contour Person representation. Colors code the different body parts. A range of articulations is shown. Note that the representation can support certain types of self-occlusions.

3.3 The Contour Person Model

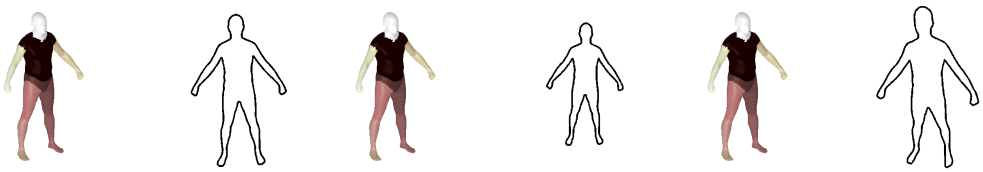
Let us suppose, for now, that we have at our disposal some contour-related shape representation that can capture a detailed articulated 2D shape (*e.g.*, as seen in Fig. 3.2). Based on this representation – to be defined later – we seek to build our statistical model. Building a 2D model of a 3D person presents many challenges. We seek a model that is expressive enough to represent a wide range of human bodies and poses, yet low dimensional enough to be computationally tractable for common computer vision problems. We build on the idea of the SCAPE model [6] and develop a *factored* model. In particular we use:

1. a low-dimensional PCA model to capture shape changes across different people;
2. a low-dimensional PCA model to approximate distortions caused by camera changes;
3. planar rotation and length scaling of body parts to represent articulation;
4. a linear prediction model to capture non-rigid deformations caused by the articulation.

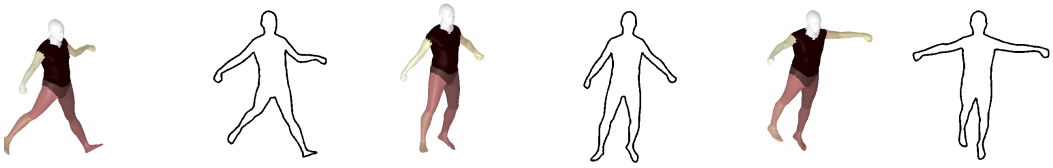
[5] see [1] for a textbook on this topic.



(a) Training contours: three examples of different body shapes



(b) Training contours: three examples of different cameras



(c) Training contours: three examples of different poses

Figure 3.3: Generating training contours. By varying the 3D body-shape (a), the 3D pose (b) or the camera parameters (c), we obtain outlines of different projected 2D shapes. Left in each pair: 3D. Right: 2D.

Figure 3.2 illustrates how varying the pose parameters results in realistic looking non-rigid deformations predicted by the CP model.

3.3.1 Generating Training Contours

The CP model is a deformable template model built from training contour data generated using samples from a 3D SCAPE body model [6], capturing realistic body shape variation and non-rigid pose variation. See Figures 3.3a and 3.3c for typical samples of shape and poses from the 3D model and their corresponding contours.

Since SCAPE itself is also a deformable template model, we consider the projection of SCAPE's 3D template – using some reference camera – to be our 2D template, denoted by $\mathcal{T}$ (see Fig. 3.4).

To understand how we generate training contours for the CP model, we start by de-

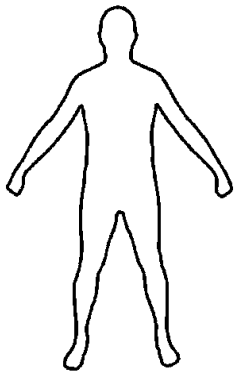


Figure 3.4: The template of the Contour Person model, denoted by $\mathcal{T}$.

scribing a simplified version of the process. Given a 3D model (*e.g.*, SCAPE), the first step is generating an instance of a 3D body. The second step is projecting the resulting 3D mesh to the image plane to obtain a 2D silhouette. The third step is extracting the discretized outline of the silhouette in order to produce a training contour obtained from the outline of the 2D shape. The last two steps are done using standard computer vision algorithms.

The process we just described captures the general idea of generating the training contours; however, to deal with problems such as self-occlusion or misalignment between different exemplars, a more complicated scheme is needed.

3.3.1.1 Dealing with Self-Occlusion and Point-to-Point Correspondence

A crucial point is that the CP representation utilizes 3D information; this is quite different from standard PS representations. This point is illustrated in the way it deals with self occlusions as well as out-of-the-plane rotations, as depicted in Fig. 3.2. In a standard contour model, the ordering of the points would be poorly defined (*cf*. [8]). Since our contours are generated from a 3D mesh, we have known correspondence between contour points and their respective points and body parts on the 3D mesh. This provides the correct connectivity of the contour even when it crosses itself in 2D.

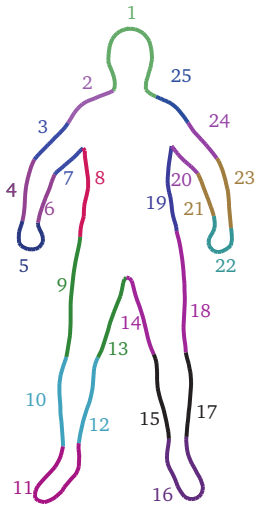


Figure 3.5: Connecting parts in a fixed order.

Instead of projecting the entire 3D body at once, we project the body parts separately. We can do this since the 3D mesh has a known segmentation to body parts. The 16 parts are: head; torso[6]; shoulders; upper arms; lower arms; hands; upper legs; lower legs; feet. From each one of these projected parts we extract its 2D outline using a MATLAB routine. The output of the routine is a polygon whose number of points need not be fixed at this stage. Henceforth we will use the terms "contour" and "polygon" interchangeably; note, however, that these polygons have many more points than the those associated with PS representations. Having all of the outlines at our disposal, we use MATLAB routines of polygon intersection, polygon union and polygon difference to obtain the portions of the outlines we are interested in (these parts are depicted in Fig. 3.5). These partial contours are in turn combined to a form single contour in a fixed ordering according to the enumeration shown in Fig. 3.5. This step results in a single polygon which might have self intersections (as in the rightmost example in Fig. 3.2).

Each partial contour is then re-sampled to have a fixed number of points, equally spaced by the arc-length; this number may vary across body parts (*e.g.*, the head requires more points than a hand), but is *held fixed across different examples*. We denote by N_{pts} the number of points in the entire contour. In our experiments $N_{pts} = 500$.

[6]SCAPE also has a pelvis part, but we treat it as part of the torso.

Importantly, as the known segmentation of the 3D model into parts induces a similar 2D contour segmentation, and since we know for each contour point the 3D point that was projected to it, the issue of misalignment is essentially eliminated: a point can never "slide" outside its body part so no gross alignment errors are present. For example, the mid point of the contour that represents the left side of the torso might slight up or down a little across different examples, but it will never go as as far as the arm or the leg. Consequently, the training contours are in good point-to-point correspondence with each other.

We also apply basic signal processing techniques to ensure smooth transitions between the different parts of the contour[7]. We omit the details, but point out that operations such as Gaussian smoothing are done with respect to the arc-length and not with respect to index difference.

Remark 3.3.1. The approach described here can accommodate certain cases of self-occlusion, typified by the illustration in Fig. 3.2. However, the restriction to a single contour (even with self-intersections) limits us to cases where the ordering of the parts is preserved. Later, when we discuss a later variant of the CP model, we will use a slightly different approach using multiple contours instead of one. That approach will support arbitrary self-occlusions.

3.3.2 Composition of Deformations and a Factored Model

For the remainder of this Chapter, we will use M as a shorter notation for $\mathrm{Sim}(2)^{N_{pts}}$. Recall that, being a matrix Lie group, M is both a group and a manifold. One of the attractive features of matrix Lie groups is the ease of composition. For example, $\mathcal{Q} \in M$ may be the result of composing two deformations:

$$\mathcal{Q} = \mathcal{Q}_1 \mathcal{Q}_2 \ , \ \mathcal{Q}_1, \mathcal{Q}_2 \in M \ . \tag{3.20}$$

[7] We work with two 1D discrete signals: $x[n]$ and $y[n]$, $n \in \{1, 2, \ldots, N_{pts}\}$, where, as is customary in signal processing literature, square brackets indicate discrete indices.

Remark 3.3.2. If only the LHS side of Eqn. (3.20) is known, then the factorization in the RHS is not unique since $\mathcal{Q}_1\mathcal{Q}_2 = \mathcal{Q}_1\mathcal{Q}_3\mathcal{Q}_3^{-1}\mathcal{Q}_2$ for any $\mathcal{Q}_3 \in M$.

In fact, a deformation can be factored into more than just two deformations; in the CP model we factor the contour deformation into several constituent parts: pose, shape, and camera. Thus, we write

$$\mathcal{Q} = \mathcal{Q}_{\text{pose}}\mathcal{Q}_{\text{shape}}\mathcal{Q}_{\text{camera}} \ . \tag{3.21}$$

To reduce dimensionality, we learn a low-dimensional parametric model for each one of these parts (these models will be described later):

$$\mathcal{Q}_{\text{shape}} = \mathcal{Q}_{\text{shape}}(\Theta_{\text{shape}}),\ \mathcal{Q}_{\text{pose}} = \mathcal{Q}_{\text{pose}}(\Theta_{\text{pose}}),\ \mathcal{Q}_{\text{camera}} = \mathcal{Q}_{\text{camera}}(\Theta_{\text{camera}}) \ ,$$

where Θ_{shape}, Θ_{pose}, and Θ_{camera} are the corresponding parameter sets. Earlier, in Section 3.3.1, we explained how we generate training contours. Once the contours have been generated we extract the deformations between $\mathcal{T}$ and them to obtain the training (deformation) data for our models. We now continue to explain this process in more detail as well as to define each part of the model.

3.3.2.1 Variation in Body Shape

To train the shape deformation model $\mathcal{Q}_{\text{shape}}(\Theta_{\text{shape}})$ we take the 3D SCAPE model and generate N realistic 3D bodies shapes in a canonical pose and, following their projections using a canonical camera, apply the process described in Section 3.3.1 to produce training contours; see Fig. 3.3a. In our experiments $N = 1000$.

For each contour we extract its deformation from T using Eqn. (3.11). We achieve dimensionality-reduction by either of the following two techniques.

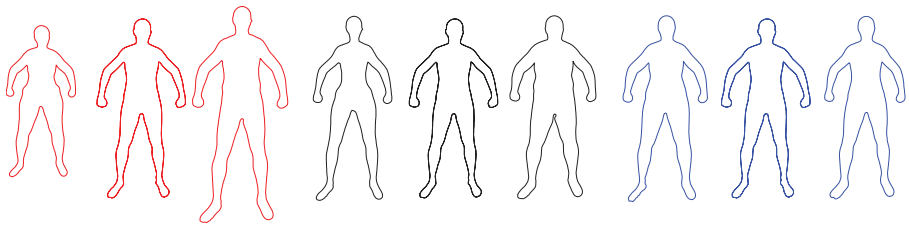


Figure 3.6: Shape variation in a gender-neutral shape model. Red: first PC. Black: second PC. Blue: third PC. In each color, from left to right: -3, 0 and +3 σ from the mean in direction of respective principal component.

Ordinary PCA on Euclidean Deformations. In the first technique, we ignore our knowledge about the manifold structure of M and treat the deformations as elements of a Euclidean space. From each training deformation $\mathcal{Q}$ we create a column vector of the form

$$[S_1 \cos\theta_1, S_1 \sin\theta_1, S_2 \cos\theta_2, S_2 \sin\theta_2, \ldots, S_{N_{pts}} \cos\theta_{N_{pts}}, S_{N_{pts}} \sin\theta_{N_{pts}}]^T \in \mathbb{R}^{2N_{pts}} \ .$$

We then take all these N vectors and (after subtracting the mean) perform ordinary Euclidean PCA. This gives a low-dimensional model of contour deformations caused by body shape variation parametrized by the PCA coefficients Θ_{shape}. The first three principal components (PCs) of a shape model, learned from samples of both genders, can be seen in Fig. 3.6 (similar gender-specific models are created and used when the gender is known). The principal components clearly capture correlated properties of human shape such as variations in height, weight, girth and so on.

Lie-algebraic PGA. The second technique is Lie-algebraic PGA. (see Chapter 2). It can be shown that the Lie algebra of Sim(2) is given by

$$\left\{ \begin{pmatrix} s & -\theta \\ \theta & s \end{pmatrix} : s \in \mathbb{R}, \theta \in \mathbb{R} \right\} \subset \mathfrak{gl}(2) \ , \tag{3.22}$$

that

$$\exp\left(\begin{pmatrix} s & -\theta \\ \theta & s \end{pmatrix} \right) = \begin{pmatrix} S\cos\theta & -S\sin\theta \\ S\sin\theta & S\cos\theta \end{pmatrix} \ (\text{where } S = \exp(s)) \ , \tag{3.23}$$

and that

$$\log\left(\begin{pmatrix} S\cos\theta & -S\sin\theta \\ S\sin\theta & S\cos\theta \end{pmatrix}\right) = \begin{pmatrix} s & -\theta \\ \theta & s \end{pmatrix} \text{ (where } s = \log(S)) \,. \tag{3.24}$$

Thus, PCA in the Lie algebra of M is done on vectors of the form

$$\begin{pmatrix} \log(S_1) - \log(\mu_{S_1}) \\ \theta_1 - \mu_{\theta_1} \\ \log(S_2) - \log(\mu_{S_2}) \\ \theta_2 - \mu_{\theta_2} \\ \vdots \\ \log(S_{N_{pts}}) - \log(\mu_{S_{N_{pts}}}) \\ \theta_{N_{pts}} - \mu_{\theta_{N_{pts}}} \end{pmatrix} \in \mathbb{R}^{2N_{pts}} \,.$$

where μ_{S_i} and μ_{θ_i} are the mean scale and mean angle, respectively, for the deformation of the $i-$th line segment (averaging is done over the N examples, not the N_{pts} line segments).

In practice, the eigenvectors for these two models look very similar. However, the model learned in the Lie algebra required slightly fewer principal components to capture the same percentage of the cumulative variance. Additionally, a random sample from the ordinary PCA model might produce local deformations of negative determinant; *i.e.*, producing a matrix outside of the manifold of similitudes. As the main topic of this chapter is the design of the CP model, and not so much the differences between Euclidean and Manifold representations, we do not pursue this comparison any further at this point. In Chapter 4, however, we will make a more thorough comparison between a Euclidean PCA model and a Lie-Algebraic PCA model, and then we will see that the differences in favor of the latter are even more substantial than here.

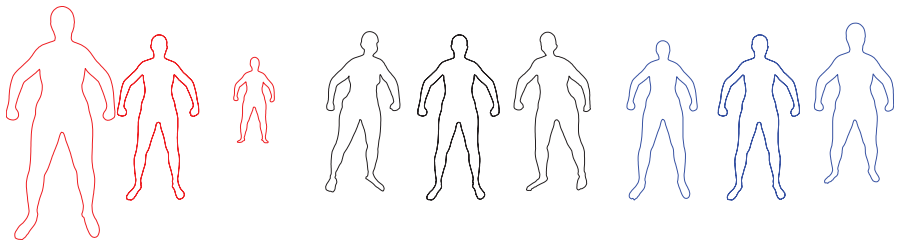


Figure 3.7: Contour-person camera variation. The first three camera principal components for the Female model. Red: first PC. Black: second PC. Blue: third PC. In each color, from left to right: -3, 0 and +3 σ from the mean in the direction of the respective PC.

3.3.2.2 Variation in Camera

A procedure analogous to the above is used to capture contour deformation due to camera variation. Training data consists of contours generated from a single fixed body shape and posture viewed by cameras of different 3D location and tilt angle. Focal length is held fixed as it has a similar affect on the model as person-to-camera distance. See Fig. 3.3b.

The deformations due to camera variation are well captured by PCA, with 6 components accounting for more than 90% of the variance; *i.e.*, $\Theta_{\text{camera}} \in \mathbb{R}^6$. The first three principal components, for the female model, can be seen in Fig. 3.7 and roughly correspond to changes in distance between the camera and the person, rotation of the camera about the person, and foreshortening of the body caused by tilt of the camera. Note that the view-variation is learned on the template person in a canonical pose and then is applied to other people and poses; this is an approximation.

3.3.2.3 Variation in Pose

In the 3D SCAPE model, deformations due to body articulation are modeled by a two-step process. First, a rigid rotation is applied to the entire limb or body part, and then local non-rigid deformations are applied according to a learned linear model. We employ a similar approach.

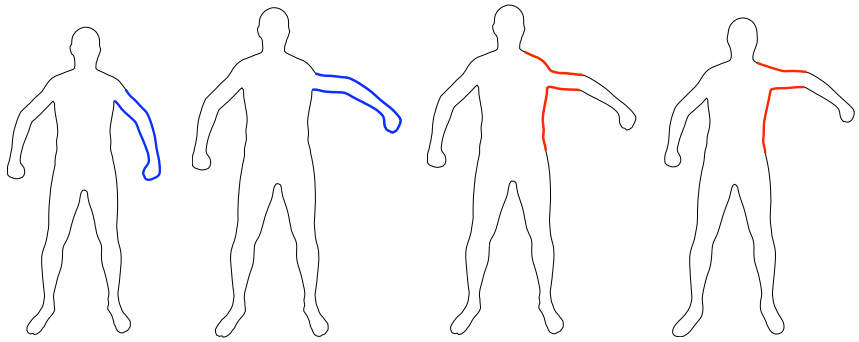
For example, in Fig. 3.8b, a rigid motion of the upper arm does not account for non-rigid deformations of the shoulder. This is corrected by applying a learned non-rigid deformation to the part of the contour that is in the vicinity of the joint (Fig. 3.8d). Specifically, we break the deformation into $\theta_i = \theta^R + \Delta\theta_i$ and $S_i = S^R \times \Delta S_i$, where $Q^R = (\theta^R, s^R)$ is the rigid deformation and $\Delta Q_i = (\Delta\theta_i, \Delta S_i)$ corresponds to non-rigid deformation. Q^R has the same value for all edges, e_i, in the same body part (for example, the left upper arm) while ΔQ_i varies along the contour.

To learn the non-rigid deformation model we generate training contours using the 3D SCAPE model, in random poses while holding the body shape fixed, and projected in the image plane using a fixed camera. See Fig. 3.3c. Note that the 3D SCAPE model already captures the non-rigid deformations of the limbs, so that the generated 2D contour looks natural. The rigid 2D rotation, θ^R, and limb scaling, S^R, of each limb is computed between the template contour and the training contour. The scale is important as it captures foreshortening of the body parts and thus helps model out-of-the-plane movements.

We also compute the deformations, Q_i, between the line segments of the template and training contours using Eqn. (3.11). We then remove the rigid rotation, θ^R, and limb scaling, S^R, from Q_i for all line segments affected by this body part to derive a residual deformation. Note, *e.g.*, that a rigid motion of the upper arm affects the non-rigid deformation of the upper arm as well as those of the lower arm and the shoulder. The residual is the deformation of the contour that is not accounted for by part-rotation and part-scaling.

Given many such ΔQ_i and Q^R (of the same i, but from different training contours) we learn a linear predictor from the rigid transformation parameters to the non-rigid deformations. Such a model is defined by

$$\begin{pmatrix} \Delta\theta_i \\ \Delta S_i \end{pmatrix} = \begin{pmatrix} \alpha_1(i) & \cdots & \alpha_{2n(i)}(i) & \alpha_0(i) \\ \beta_1(i) & \cdots & \beta_{2n(i)}(i) & \beta_0(i) \end{pmatrix} p, \tag{3.25}$$



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

Figure 3.8: Non-rigid deformation due to pose. (a) the template with left arm marked in blue. (b) rigid transformation of the upper arm. (c) same as (b) but with parts which should be non-rigidly deformed due to the rigid motion marked in red. (d) final deformed contour with the non-rigidly deformed parts marked in red.

where $p = \left(\theta_1^R, S_1^R, \ldots, \theta_{n(i)}^R, S_{n(i)}^R, 1\right)^T \in \mathbb{R}^{2n(i)+1}$ is a vector of rigid transformations, $n(i)$ is the number of parts affecting Q_i, and the α's and the β's are parameters to be learned. Once the model is learned, for every choice of Q^R, we compute the associated ΔQ_i's. Then we can compute the full Q_i's, and define $\mathcal{Q}_{\text{pose}}$ in a similar way to $\mathcal{Q}_{\text{shape}}$ and $\mathcal{Q}_{\text{camera}}$. The difference is that Θ_{pose} does not represent PCA coefficients. Instead, it represents the different scales and rotations of each body part.

Figure 3.9, depicts the outlines of contour people sampled at random from the pose.

3.3.2.4 The Full Model

We train each deformation model independently as described above and then compose them. The ease of composition of deformations – which boils down to simple matrix multiplication – is what enables our model to be factored. This is a key advantage over contour representations that use vertices directly. Since M is Abelian, the composition order is immaterial. Given $\Theta \stackrel{\text{def}}{=} (\Theta_{\text{shape}}, \Theta_{\text{pose}}, \Theta_{\text{camera}})$, the full parameter set of the CP model, the overall deformation is given by the *parametric factorized-deformation synthesis* equation:

$$\mathcal{Q}(\Theta) = \mathcal{Q}_{\text{shape}}(\Theta_{\text{shape}})\mathcal{Q}_{\text{pose}}(\Theta_{\text{pose}})\mathcal{Q}_{\text{camera}}(\Theta_{\text{camera}}) \; . \tag{3.26}$$

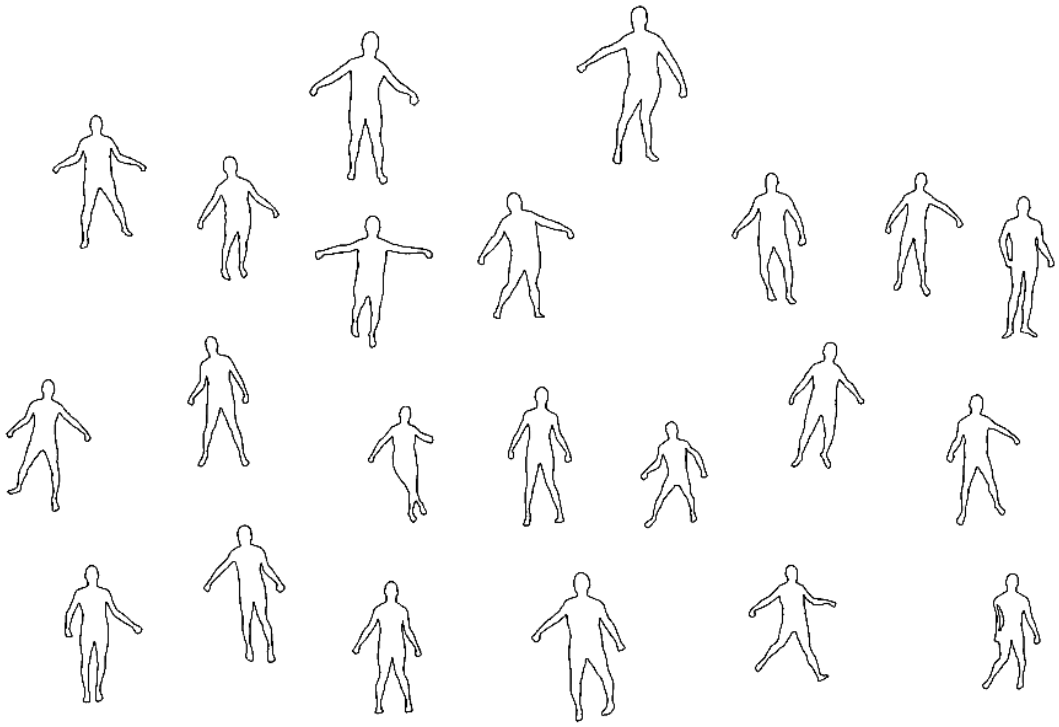


Figure 3.9: Outlines of contour people sampled at random poses.

Figure 3.10: Contour people sampled from the model. Large deviations from the mean body are shown for shape, pose, and camera. Row 1: variations in body shape. Row 2: variations in pose. Row 3: variations in camera view. Row 4: all variations together.

$\mathcal{Q}(\Theta)$ can be substituted into Eqn. (3.19) to produce a new $\mathcal{C}$. Here we use 24 pose parameters (12 joints $\times 2$), 10 shape coefficients and 6 camera coefficients, for a total of 40 parameters.

This factored model is an approximation, but one that works well. Example contours synthesized from the generative model are shown in Fig. 3.10. Note that PCA implies a Gaussian probabilistic model defined by the variance along the principal component directions. This works well for body shape and camera pose where the training samples are roughly normally distributed. The figure shows camera- and shape-variations sampled from this model. The joint angles and limb scaling are sampled uniformly over a predefined range. Note that, because the 2D model is generated from 3D, there are correlations in 2D joint angles and scaling that could be modeled; this is future work.

3.4 Pose Estimation Combined with Segmentation

As an example application of this model, we considered the problem of segmenting images of humans. The CP model provides a strong prior over human body shape that can be used to constrain more general segmentation algorithms such as GrabCut [124]. Specifically we search over the CP parameters that optimally segment the image into two regions (person and non-person) using a cost function that 1) compares image statistics inside the contour with those outside; 2) favors contours that align with image edges; 3) enforces our prior model over shape, pose and camera parameters.

Initialization. We initialize the CP model using the output of a standard PS algorithm [4]. The PS model is lower dimensional than the full CP model and hence provides a more efficient initialization. We simply set the rigid deformation parameters (rotation and scale) in the CP model to be equal to those of the PS model. While the PS model defines a segmentation of the image, it is a crude depiction of the human form. Consequently we refine the segmentation using the CP model.

Region term. The region term of the segmentation objective compares intensity and color histograms inside and outside the body contour. We take the pixel mask $m(\Theta)$ consisting of all pixels of the image plane I_c within the contour and compare the normalized histograms $H_c^{\text{in}}(I,m) = hist(I_c(m))$ and $H_c^{\text{out}}(I,m) = hist(I_c(\bar{m}))$ using the χ^2 histogram distance:

$$d_c(I,m) = 2 - \sum_i \frac{\left(H_c^{\text{in}}(i) - H_c^{\text{out}}(i)\right)^2}{H_c^{\text{in}}(i) + H_c^{\text{out}}(i)}. \tag{3.27}$$

We follow Martin, *et al.* [103] in treating intensity histograms and color histograms as separate features (we use the YCbCr colorspace)

$$E_{\text{St}}(I,m) = \lambda_1 d_Y(I,m) + \lambda_2 d_{Cb}(I,m) + \lambda_3 d_{Cr}(I,m).$$

Edge term. The segmented contour should also follow image edges. We detect image edges using a standard edge detector and apply a thresholded distance transform to define an edge cost map normalized to $[0,1]$. We use the trapezoid rule to evaluate the line integral of the set of all model edges over the edge cost image. This defines an edge cost, $E_{\text{Eg}}(I,\Theta)$, that is included in the objective function.

Prior. We use a loose prior, $E_{\text{Pr}}(\Theta)$, on shape, pose and camera only to prevent values significantly outside what the model is trained on. This prior remains zero until the parameters are three standard deviations from the mean and then rises linearly from there.

Objective. The full cost function is then $E(I,\Theta) = E_{\text{St}}(I,m) + \lambda_4 E_{\text{Eg}}(I,\Theta) + \lambda_5 E_{\text{Pr}}(\Theta)$, which we optimize using a gradient-free direct search simplex method.

(a) (b)

Figure 3.11: An image of a person and the corresponding manually-segmented silhouette.

3.5 Experimental Results

3.5.1 Fitting To Silhouettes

As first sanity-check aimed to test if the CP model can capture human shapes taken from real images (*e.g.*, Fig. 3.11a), we have fitted the model to silhouettes segmented by hand (*e.g.*, Fig. 3.11b). Fitting was done by minimizing, with respect to Θ, a bi-directional Chamfer distance between the manually-segmented silhouette and the silhouette implied by the model. As can be seen by several selected results shown in Fig. 3.12, it turns out that the CP is indeed expressive enough.

3.5.2 Pose Estimation Combined with Segmentation

The CP model realistically captures a large range of real human poses in the space in which it was trained Fig. 3.13). This enables it to find segmentations which, while not perfect, are guaranteed to be plausibly human, unlike more general segmentation methods Fig. 3.14). This is a fairly simplistic segmentation approach which is designed only to

Figure 3.12: Fitting the CP model to manually segmented silhouettes: selected results.

illustrate the CP model; note that the parametric segmentation method here is similar in spirit to PoseCut [12].

Also, note that the model is not clothed and consequently will produce segmentations that tend to ignore clothing. While model fitting could be made robust to clothing [16], for segmenting clothed people it is preferable to explicitly model clothing [57, 58].

Given our simplistic segmentation method, the model can also make mistakes such as those in Figures 3.13a and 3.13b, where the optimization latches on to a strong edge at the hairline and finds that the hair matches the background color statistics better than the foreground statistics; this pushes the shoulders down, causes the head to be smaller than the torso and legs would otherwise indicate, so the camera gets detected as tilted upwards, which in turn causes the shoulders to narrow and the arms to shorten. In Fig. 3.13g the PS initialization is far enough off that a simple direct search optimization method cannot escape the local minimum; note though that only the left arm was poorly initialized and that only the left arm remains poorly localized and segmented. Another failure case is typified by the left hand in Fig. 3.13f. We train this model without varying the angle of the wrist and our training data consists of exclusively closed fists. Consequently the model does a poor job representing open hands and bent wrists.

Figure 3.13: Selected results for pose estimation and segmentation. Left column: PS result. Middle: CP initialization from PS (red) and CP result (green). Right: CP result.

Figure 3.14: Comparison to GrabCut. GrabCut with a manual initialization step and no manual cleanup, compared to fully automatic CP segmentation. In the first example, note the redundant shadow region that Grabcut attached to the woman's leg. In the two other examples, GrabCut fails to capture one of the legs. In contrast, by design, the CP model ensures the output is a legitimate human shape.

3.6 Conclusion

We have defined a new type of shape representation for articulated 2D human body that retains the standard part-based structure of classical PS representations. Over this representation we define a new statistical model that goes beyond previous statistical models of 2D human shape in several significant ways. First, it *factors* 2D body shape into several causes. Deformations from a training template are used to describe changes in shape due to camera view, body shape, and articulated pose. The approach is similar to the 3D SCAPE model in that deformations are combined into a complete generative model. Second, the CP model captures the non-rigid deformations of the body that result from articulation. Like SCAPE, these are learned from training examples. The result is a fairly low-dimensional model that represents realistic human body contours and can be used for vision applications such as pose estimation, person detection and tracking. Since its introduction [43], the CP model has provided the basis for models that either handle 2D deformations of loose clothing [57] or support inference through belief propagation [155]. Ongoing works that use the CP model or its variants include activity recognition and a human-specific model for optical flow. Future work should include application of the CP model – or one of its variants – to tracking.

Chapter 4

Lie Shapes

Quoting Mumford and Desolneux [107]:

> "Grenander [51] has often emphasized that when we want to model some collections of patterns, it is very important to consider the symmetries of the situation – whether there is an underlying group."

Just like *Flatland*'s unlikely narrator, we now make the transition from two dimensions to three. The current chapter is focused on the *representation* of deformable 3D shapes. In particular, we consider surfaces represented as triangulated meshes and develop a new representation for their deformations. Our manifold-based representation, self-coined *Lie shapes*, leads to statistical shape models that are better than those based on existing Euclidean representations of shape deformations. This representation can be easily utilized in existing statistical models (*e.g.*, [6,48,62]), and thus its scope is not limited to a particular statistical model. We illustrate our representation with human body shapes; as depicted in Fig. 4.1 (left), shapes correspond to points on a nonlinear manifold - the manifold of mesh deformations. The formulation, however, is completely general: it applies to other shape classes as well and is not limited to human bodies.

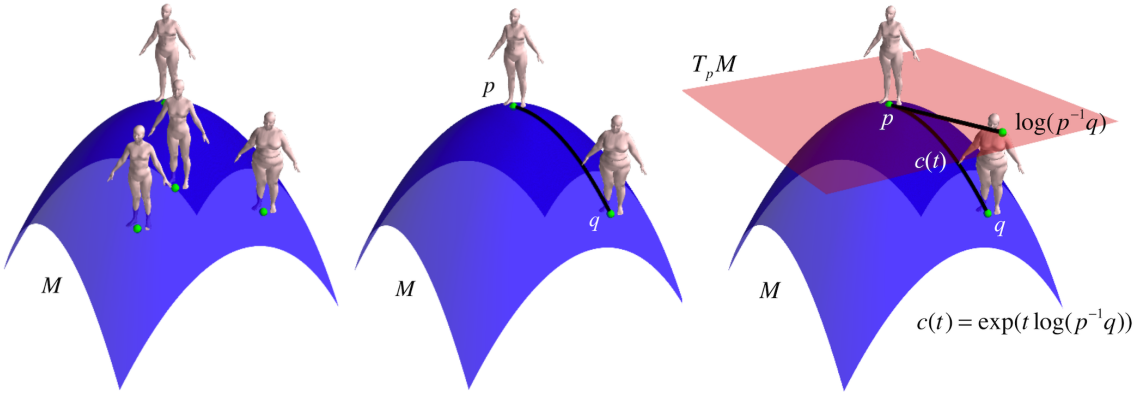


Figure 4.1: The manifold of mesh deformations of the human body – a particular case of Lie shapes. Left: Human shapes as points on M, a nonlinear manifold with a matrix Lie group structure. Every point represents a deformation from a template. Center: Distance between shapes p and q is measured via a geodesic distance; *i.e.*, the length of the shortest path between them along M. Right: The tangent space at p, denoted by T_pM, is a vector space.

One of the advantages of Lie shapes is their associated distance measure. We stress that our primary interest is in facilitating better statistical models. While our representation also supports *shape interpolation*, yielding better results than its Euclidean counterparts, our interest in this application is only secondary[1]; however, shape interpolation is strongly tied to *distances*, the latter being crucial for statistics. Thus, shape interpolation provides us with yet another way to illustrate the benefits of our representation and its distance measure over Euclidean deformation approaches that are currently used in state-of-the-art statistical models. Additionally, it also clearly shows one of the reasons why working directly with vertices is inferior to working with transformations.

Let us lay out the setup. We assume a dataset of registered 3D meshes with the same connectivity of N_{tri} triangles (see Fig. 4.2). Having chosen a particular shape representation that enables to capture deformations between two meshes, the *deformable mesh statistical modeling problem* has two parts:

1. Given a set of training meshes $\{\mathcal{M}_i\}_{i=1}^N$ and a template mesh $\mathcal{T}$, for each $\mathcal{M}_i$, extract the deformation between $\mathcal{T}$ and $\mathcal{M}_i$.

[1] In particular, we make no claims about our method being the optimal approach to shape interpolation for every possible computer graphics application.

2. Learn a statistical model of these deformations.

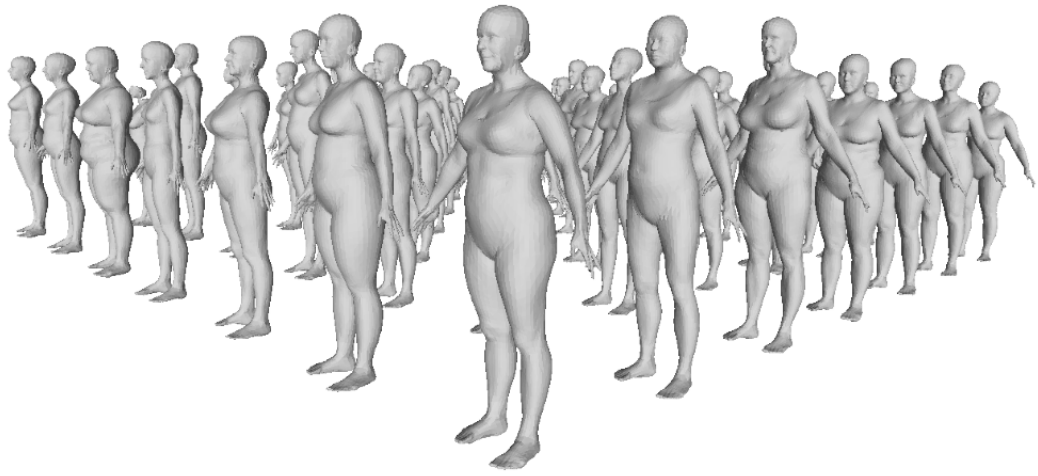
For effective statistical learning, it is crucial to have an appropriate shape representation. The simplest approach is to work directly with the points in $\{\mathcal{M}_i\}_{i=1}^N$ or their displacements from $\mathcal{T}$. While this may work well for rigid objects, it is a poor choice for non-rigid and/or articulated objects such as the human body where the deformations are more complex and can result from a composition of multiple causes. The common approach represents shape in terms of 3-by-3 transformation matrices acting on the triangles of $\mathcal{T}$, and treats these matrices as elements of a nine-dimensional space. Consequently, each matrix has 9 Degrees of Freedom (DoF).

In this chapter we define an appropriate mathematical representation for mesh deformations in terms of a manifold. We call our representation *Lie Shapes* as it is based on an easy-to-understand[2] new *six-dimensional Lie group of triangle deformations* that eliminates redundant DoF. The action of the group is illustrated in Fig. 4.3 and will be explained later. With this group, a deformation between two triangles can be computed exactly, in closed-form, without heuristics.

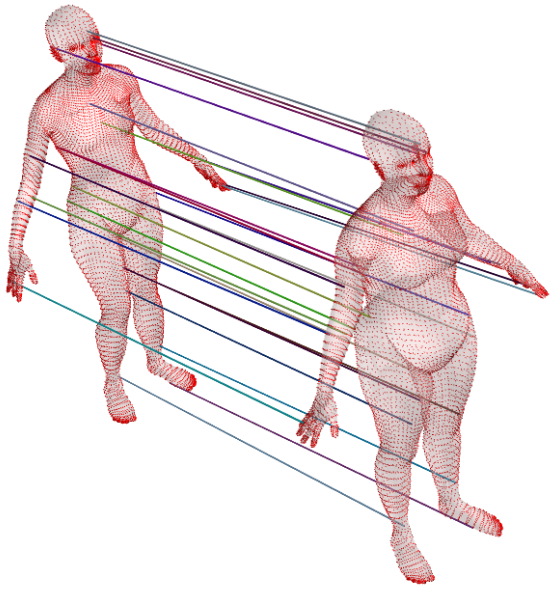
The $6N_{tri}$-dimensional manifold of Lie Shapes, denoted by M, is built from N_{tri} copies of this six-dimensional group. M has a Riemannian structure inducing a *left-invariant* distance between shapes that is computed in closed-form. This metric defines distances between body shapes in a principled way using the length of the shortest path (on the manifold) between them; see Fig. 4.1 (middle). It also allows us to compute *statistics on manifolds* using methods such as those mentioned in Chapter 2. These better capture the statistics of human body shape deformations than do standard Euclidean methods.

We focus here on human shape and show that our formulation results in a better, more parsimonious, and more accurate model of shape deformation than the traditional Euclidean representation. We evaluate performance in several ways: 1) Our group structure

[2]It was tempting to use the word "simple" here, but mathematically this would have been wrong: the class of *simple groups* is a well-defined term, and our group does not belong to it.



(a)



(b)

Figure 4.2: A dataset of human shapes. While for visualization purposes the shapes are displayed in their rendered form (a), they are in fact stored as 3D triangular meshes (b). Note that meshes are registered: they have the same number of points with a known point-to-point correspondence between meshes. Here, to avoid clutter, only a subset of these correspondences are shown.

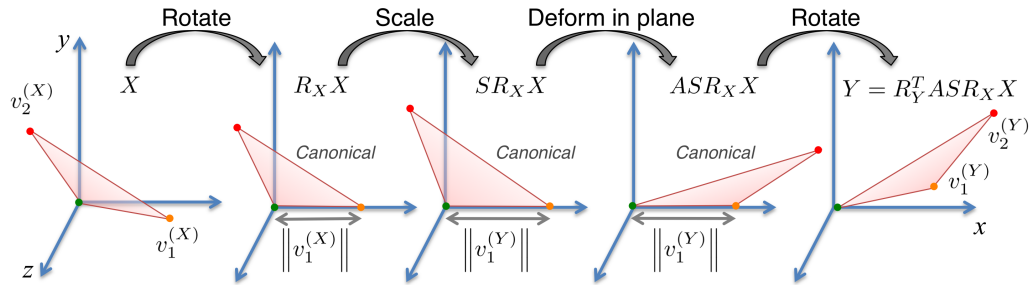


Figure 4.3: Deforming $X = [v_1^{(X)}, v_2^{(X)}]$ to $Y = [v_1^{(Y)}, v_2^{(Y)}]$ through several steps. See text fore more details.

results in meshes that are better behaved and exhibit lower variance across a database of registered body shapes. 2) For a fixed number of low-dimensional shape vectors we find that our model is better able to predict biometric measurements. 3) For a fixed number of shape vectors, our reconstruction of Euclidean shape is even better than the Euclidean model. 4) Finally we show that our representation better captures properties of body shape related to human perception.

To recap, our main contributions covered in this chapter are:

1. A novel nonlinear manifold representation for deformations of triangular meshes. This manifold has the minimal number of DoF required for arbitrary triangle deformations, provides a heuristic-free way to compute deformations, and eliminates non-physical deformations.

2. This representation is consistent in the sense it has a group structure: deformations can be composed or inverted in a meaningful way.

3. A principled way to measure distances and interpolate between shapes using geodesic distances and geodesic paths.

4. We show how statistics on the manifold capture shape variation over a database of aligned triangular meshes.

5. Closed-form formulas, or efficient approximations, are given for all computations.

4.1 Previous Work

Sumner and Popović [142] define shape in terms of affine deformations of triangles from $\mathcal{T}$. A triangle deformation is represented by a 3×3 deformation matrix and a 3D displacement vector. The 9 dimensional space of deformations is under-constrained as deformations outside the plane of the triangle are undefined. They deal with this heuristically by adding a fourth virtual vertex defined by the cross product of two of the triangle edges.

Anguelov *et al*. [6] use these matrices to define the SCAPE model, which factors deformations due to body shape from those due to pose. Bălan [14] builds a SCAPE model from the CAEASR dataset [120] and regularizes the ambiguity in the 3×3 deformations using a spatial smoothness constraint that penalizes difference in deformations between neighboring triangles. Hasler *et al*. [62] learn a multilinear model of affine deformations. Like all the above models, distances between deformations are measured in a Euclidean space with redundant DoF. Hasler *et al*. [63] use an even higher-dimensional representation of deformations. They model deformations with 15 DoF and a nonlinear encoding of triangle deformations that captures dependencies between pose and shape. Again, Euclidean distance still plays a central role.

Note that in fact, triangle deformations lie in a 6 dimensional nonlinear manifold. Unlike previous methods, our deformations live on this manifold, have positive determinant by construction, and thus exclude non-physical deformations such as reflections[3].

Chao *et al*. [19] represent deformations, using an analogy to elasticity, as a rotation plus an affine residual deformation closest (in a Euclidean sense) to an isometry. Their affine deformations have positive determinant but have more than 6 DoF. Additionally, to compute an approximation of their shape distance and the path between shapes, they require an expensive optimization scheme. In contrast, we provide accurate closed-form

[3] One may argue that sometimes we want to reflect a mesh around some axis. This may be the case, but this should be accomplished by a single global reflection affecting the entire mesh. A situation where only certain isolated individual triangles are "flipped" is undesirable.

formulas for paths and distances. Also, in [19] there is no notion of a Lie group or shape deformation statistics.

The idea of employing Lie groups to represent 3D deformations is not new and is widely used in computational anatomy; *e.g.*, see [35–38, 54, 105, 146]. These methods, however, work on volumetric representations rather than triangulated meshes.

Alexa [2] uses a Lie group for triangle deformations but both his motivation (addressing noncommutativity) and solution (defining a new group operation) are quite different than ours. Essentially, his approach operates in the tangent space at the identity of standard matrix Lie groups while ignoring a non-vanishing Lie bracket; this approximation can be justified only if matrices are close to each other. Additionally, the set of matrices comprising his group is chosen ad-hoc, and so, depending on the case, suffers from either excessive DoF or is not expressive enough to accurately capture arbitrary mesh deformations. Of note, graphics applications described in [2] (or [83]) can benefit from our representation.

Note that our work is not directly related to the classical work of Kendall [80], despite the fact we use a manifold representation for shapes. In [80], shapes are represented by a set of landmarks, and their quotient spaces are studied. In contrast, our manifold represents a group of *transformations* acting on 3D triangular surfaces. For a more recent work on geometric modeling based on Kendall's theory, see [83]. Note that since we provide closed-form formulas for geodesic paths, interpolation (or extrapolation) on our manifold is considerably simpler than the algorithms presented in [83].

A possible source of confusion might arise due to the term geodesic distance. Many mesh registration methods try to align meshes so as to preserve geodesic distances on the surface of the mesh. Here when we refer to a geodesic distance, it is the distance between two *shape deformations* living on a $6N_{tri}$ dimensional manifold, and not between *points on the 3D surface of a particular shape.*

To summarize, we introduce a manifold representation for accurate arbitrary non-rigid

deformations using a six-dimensional Lie group acting directly on the 3D triangular surfaces. Our shape representation is naturally suitable for composition of shape deformations. Finally, the ease of computations on our manifold is in sharp contrast to alternative nonlinear representations.

4.2 The Triangle Deformation Paradigm

We are working within an existing paradigm of *expressing differences between two triangular meshes through transformations acting on triangles*. These transformations are called triangle deformations. Within this paradigm, our contribution is suggesting a new way to represent these deformations. The paradigm was first introduced in the computer graphics community [142], where it was also used to build a statistical factored model, called SCAPE, of articulated human shape [6]; see Fig. 1.1. The factorization – to body-shape deformations and pose deformations – is possible as transformations lend themselves to composition. SCAPE, in turn, was imported into computer vision where it is now used for pose and shape estimation [15, 16, 59, 149]. This model also led to several additional variants [58, 62, 63, 72]. Finally, triangle deformations are also useful for automatic segmentation of an articulated mesh into its constituent parts [48].

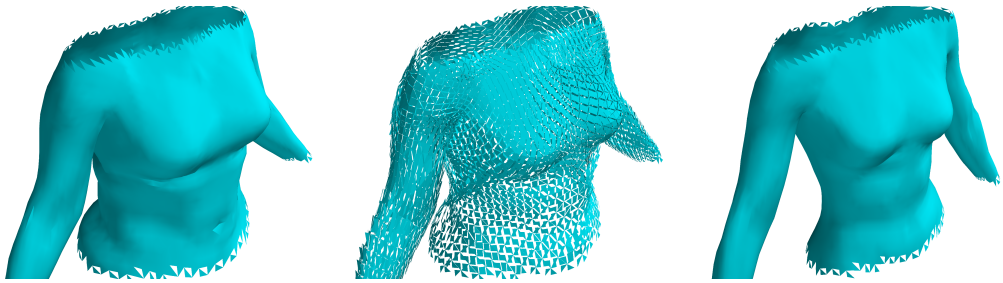
The reminder of the current section is dedicated to explaining this paradigm; our representation will be presented in the next section.

4.2.1 Basic Definitions

Schematically, mesh synthesis is done in two steps (see Fig. 4.4 for an illustration):

1. Local deformations are applied to individual triangles.

2. A second procedure then finds the mesh whose triangle edges are as close as possible



(a) (b) (c)

Figure 4.4: Deforming one mesh to another through deformations of triangles is a two-step process. Step 1: the triangles of a template mesh (a) are deformed (b). The deformations are translation-invariant: for the purpose of visualization, the first vertex of each triangle in (b) is located in the same place as the first vertex of the corresponding triangle in (a). Hence the disconnectedness in (b). Step 2: a second procedure is applied in order to find the mesh (c) whose triangle edges are as close as possible to the edges of the deformed triangles in (b). See text for more details.

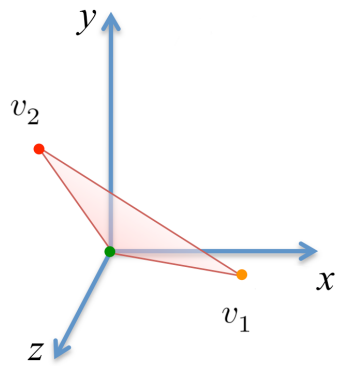


Figure 4.5: When translation is ignored, a triangle is identified with two directed edges.

to the edges of the deformed triangles.

This scheme is not unlike the one from Section 3.2.4. We now proceed to the basic definitions of triangle deformations. We will always assume that triangles are nondegenerate. Let $(v_0, v_1, v_2) \subset \mathbb{R}^3$ be an ordered triplet defining a triangle. Without loss of generality, we assume $v_0 = [0, 0, 0]^T$ so we can *identify a triangle with a matrix consisting of an ordered pair of directed edges*:

$$[v_1 - v_0, v_2 - v_0] = [v_1, v_2] \in \mathbb{R}^{3\times 2} \tag{4.1}$$

(see Fig. 4.5 for an illustration).

Remark 4.2.1 (The ordering of the points in a triangle). Note the word "ordered": we consider $[v_1, v_2]$ and $[v_2, v_1]$ to represent two different elements in the space of triangles. Also note it is in fact impossible for v_1 and v_2 to be equal: the triangle is nondegenerate.

We can compute all of the N_{tri} pairs of directed edges associated with a mesh $\mathcal{M}$ in a single linear operation. To that aim, assuming there are N_{verts} 3D points (also known as vertices) in $\mathcal{M}$, it is most convenient to regard $\mathcal{M}$ as an element of $\mathbb{R}^{N_{verts} \times 3}$. We construct a sparse matrix $E \in \mathbb{R}^{2N_{tri} \times N_{verts}}$, taking values in $\{-1, 0, 1\}$, so that $E\mathcal{M} \in \mathbb{R}^{2N_{tri} \times 3}$ contains the desired result. Finally, let $(E\mathcal{M})_{reshape} \in \mathbb{R}^{3N_{tri} \times 2}$ denote the reshaping of $E\mathcal{M}$ such that the N_{tri} matrices of pairs of directed edges are stacked vertically. A simple numeric example is in order.

Example 4.2.1 (Constructing E for a 3D mesh)**.** Let $N_{verts} = 5$, implying

$$\mathcal{M} \stackrel{\text{def}}{=} \begin{pmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \\ x_4 & y_4 & z_4 \\ x_5 & y_5 & z_5 \end{pmatrix} \in \mathbb{R}^{5 \times 3} \ . \tag{4.2}$$

Let $N_{tri} = 3$. Assume that the mesh connectivity is given by the following ordered triplets:

$$((x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3)) \qquad \text{(points comprising triangle \#1)} \ ; \tag{4.3}$$

$$((x_2, y_2, z_2), (x_1, y_1, z_1), (x_4, y_4, z_4)) \qquad \text{(points comprising triangle \#2)} \ ; \tag{4.4}$$

$$((x_4, y_4, z_4), (x_1, y_1, z_1), (x_5, y_5, z_5)) \qquad \text{(points comprising triangle \#3)} \ . \tag{4.5}$$

Finally let the 3-by-2 matrices $[v_1, v_2]$, $[v_3, v_4]$, and $[v_5, v_6]$ denote the corresponding pairs of directed edges. Then, E has form as in the following equation:

$$\overbrace{\begin{pmatrix} v_1^T \\ v_2^T \\ v_3^T \\ v_4^T \\ v_5^T \\ v_6^T \end{pmatrix}}^{\text{directed edges}} \stackrel{\text{by def.}}{=} \begin{pmatrix} x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \\ x_1 & y_1 & z_1 \\ x_4 & y_4 & z_4 \\ x_1 & y_1 & z_1 \\ x_5 & y_5 & z_5 \end{pmatrix} - \begin{pmatrix} x_1 & y_1 & z_1 \\ x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_2 & y_2 & z_2 \\ x_4 & y_4 & z_4 \\ x_4 & y_4 & z_4 \end{pmatrix} = \overbrace{\begin{pmatrix} -1 & +1 & 0 & 0 & 0 \\ -1 & 0 & +1 & 0 & 0 \\ +1 & -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & +1 & 0 \\ +1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & +1 \end{pmatrix}}^{E \in \mathbb{R}^{6 \times 5}} \overbrace{\begin{pmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \\ x_4 & y_4 & z_4 \\ x_5 & y_5 & z_5 \end{pmatrix}}^{\mathcal{M}, \text{ by def.}} , \tag{4.6}$$

so $E\mathcal{M} \in \mathbb{R}^{6 \times 3}$. Finally,

$$(E\mathcal{M})_{reshape} = \begin{pmatrix} v_1 & v_2 \\ v_3 & v_4 \\ v_5 & v_6 \end{pmatrix} \in \mathbb{R}^{9 \times 2} \ . \tag{4.7}$$

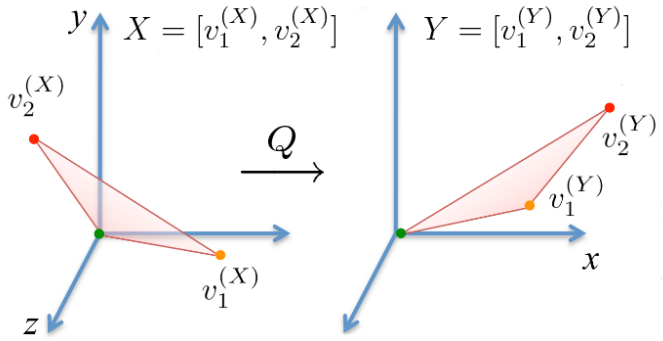


Figure 4.6: Deforming a triangle X to another triangle Y using a deformation matrix Q. The different colors of the points indicate the point-to-point correspondence between the triangles.

The generalization from Example 4.2.1 to other cases is straightforward. For a given set of aligned meshes, the connectivity is fixed across all of the meshes and so E is fixed too. Thus, for example, we write the directed edges in the template $\mathcal{T}$ as $(E\mathcal{T})_{reshape} \in \mathbb{R}^{3N_{tri}\times 2}$.

Definition 4.2.1 (Deformation matrix)**.** Let $X \in \mathbb{R}^{3\times 2}$ and $Y \in \mathbb{R}^{3\times 2}$ be a pair of triangles. If $Q \in \mathbb{R}^{3\times 3}$ satisfies

$$Y = QX \ , \tag{4.8}$$

then it is called a *deformation matrix* and we say it is acting on X and deforming it to Y (see Fig. 4.6 for an illustration).

As a mesh $\mathcal{M}$ is made out of N_{tri} triangles, we have N_{tri} such deformations with respect to the template mesh $\mathcal{T}$ denoted by $\{Q_i\}_{i=1}^{N_{tri}}$. In other words, we have N_{tri} equations of the form $Y_i = Q_i X_i$, where $\{X_i\}_{i=1}^{N_{tri}}$ and $\{Y_i\}_{i=1}^{N_{tri}}$ stand for the sets of triangles associated with $\mathcal{T}$ and $\mathcal{M}$, respectively. For a given i, we think of Q_i as a *local deformation*. The word "local" does not mean that the deformation is small. Rather, it means that Q_i pertains to a particular triangle and not to the entire mesh. We denote the entire set of deformations (for a given mesh) by $\mathcal{Q} = [Q_1, Q_2, \ldots, Q_{N_{tri}}]$, and *identify it with a sparse block-diagonal*

matrix,

$$\mathcal{Q} \leftrightarrow \begin{pmatrix} Q_1 & & & 0 \\ & Q_2 & & \\ & & \ddots & \\ 0 & & & Q_{N_{tri}} \end{pmatrix} ; \tag{4.9}$$

i.e., $\mathcal{Q}$ is a sparse $3N_{tri}$-by-N_{tri} matrix, whose only possible nonzero entries can appear in 3-by-3 blocks along its main diagonal.

We think of $\mathcal{Q}$ as the *mesh deformation* of $\mathcal{M}$ with respect to $\mathcal{T}$. Consequently, assuming $\mathcal{T}$ and $\mathcal{M}$ are fixed and $\mathcal{Q}$ is the corresponding mesh deformation, we have the following equality:

$$(E\mathcal{M})_{reshape} = \mathcal{Q}(E\mathcal{T})_{reshape} \in \mathbb{R}^{3N_{tri}\times 2} . \tag{4.10}$$

The juxtaposition of $\mathcal{Q}$ and $(E\mathcal{T})_{reshape}$ stands for matrix multiplication; it does not indicate that the former is a function of the latter. Let $\mathcal{Y} \stackrel{\text{def}}{=} (E\mathcal{M})_{reshape}$ and $\mathcal{X} \stackrel{\text{def}}{=} (E\mathcal{T})_{reshape}$, denote the triangles of $\mathcal{M}$ and $\mathcal{T}$, respectively. With this notation, Eqn. (4.10) can be rewritten analogously to Eqn. (4.8) as

$$\mathcal{Y} = \mathcal{Q}\mathcal{X} . \tag{4.11}$$

4.2.2 Local Deformation Analysis and Mesh Deformation Analysis

Given $\mathcal{T}$ and $\mathcal{M}$, the *mesh deformation analysis problem* is to find a mesh deformation $\mathcal{Q}$ that satisfies Eqn. (4.10). In its local form, called the *local deformation analysis problem*, the goal is to find Q that satisfies Eqn. (4.8), with X and Y standing for two triangles that belong to $\mathcal{T}$ and $\mathcal{M}$, respectively. Recall that X and Y are both (known) elements of

$\mathbb{R}^{3\times 2}$. Thus, we need to solve the following under-constrained equation:

$$\overbrace{\begin{pmatrix} X_{1,1} & X_{2,1} & X_{3,1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & X_{1,1} & X_{2,1} & X_{3,1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & X_{1,1} & X_{2,1} & X_{3,1} \\ X_{1,2} & X_{2,2} & X_{3,2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & X_{1,2} & X_{2,2} & X_{3,2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & X_{1,2} & X_{2,2} & X_{3,2} \end{pmatrix}}^{6\times 9} \overbrace{\begin{pmatrix} Q_{1,1} \\ Q_{1,2} \\ Q_{1,3} \\ Q_{2,1} \\ Q_{2,2} \\ Q_{2,3} \\ Q_{3,3} \\ Q_{3,2} \\ Q_{3,3} \end{pmatrix}}^{9\times 1} = \overbrace{\begin{pmatrix} Y_{1,1} \\ Y_{2,1} \\ Y_{3,1} \\ Y_{1,2} \\ Y_{2,2} \\ Y_{3,2} \end{pmatrix}}^{6\times 1} . \tag{4.12}$$

Suppose, for now, that given $\mathcal{T}$ and $\mathcal{M}$ we have chosen a method that enables us, for each one of the $\{(X_i, Y_i)\}_{i=1}^{N_{tri}}$ pairs, to find a Q_i that solves the corresponding local deformation analysis problem. Applying such method to the entire $\{Q_i\}_{i=1}^{N_{tri}}$ set enables us to find *a* solution for the mesh deformation analysis problem. Let us denote this solution by $\mathcal{Q}_{\mathcal{M}}^{\text{extracted}}$; *i.e.*, $(E\mathcal{M})_{reshape} = \mathcal{Q}_{\mathcal{M}}^{\text{extracted}}(E\mathcal{T})_{reshape}$. Or, equivalently, $\mathcal{Y} = \mathcal{Q}_{\mathcal{M}}^{\text{extracted}}\mathcal{X}$.

4.2.3 Triangle Synthesis and Mesh Synthesis

Let X be a triangle that belongs to the template $\mathcal{T}$; *i.e.*, X is known. For a nominal value of a local deformation Q, it is easy to *synthesize a new triangle* Y: just set $Y = QX$. Likewise, for a nominal value of a mesh deformation $\mathcal{Q}$, we can synthesize a new set of N_{tri} triangles by setting $\mathcal{Y} = \mathcal{Q}\mathcal{X}$. In an analogy to the considerably simpler problem we saw in Section 3.2.4, this is where the plot thickens: for a nominal value of $\mathcal{Q}$, it is likely that there does not exist a mesh $\mathcal{M}$ whose extracted deformation satisfies $\mathcal{Q}_{\mathcal{M}}^{\text{extracted}} = \mathcal{Q}$. Equivalently, for a nominal value of $\mathcal{Y}$, implied by a nominal value of $\mathcal{Q}$, it is likely that there does not exist a mesh $\mathcal{M}$ satisfying $\mathcal{Y} = E\mathcal{M}$.

A popular solution, and one which we adopt as well, is to solve the following problem:

$$\begin{aligned} &\text{minimize} \;\; f(\mathcal{M}) = \|(E\mathcal{M})_{reshape} - \mathcal{Q}(E\mathcal{T})_{reshape}\|_F^2 \\ &\text{subject to} \;\; \mathcal{M} \in \mathbb{R}^{Nverts\times 3} \; . \end{aligned} \tag{4.13}$$

With f so defined,

$$\mathcal{M} = \underset{x \in \mathbb{R}^{Nverts \times 3}}{\arg\min} \; f(x) \tag{4.14}$$

is called the *mesh synthesis equation*. The optimization problem above can be written more compactly as minimizing $\|\mathcal{Y} - \mathcal{Q}\mathcal{X}\|_F^2$ (subject to the same constraint) but the original notation makes the dependency of the cost function on the mesh $\mathcal{M}$ more explicit.

Most of the following remarks are analogous to the ones we made after Eqn. (3.19).

1. The minimizer exists but is unique only up to a global 3D translation of the mesh. Consequently, and without loss of generality, we may assume that the first row in $\mathcal{M}$ contains only zeros. Consequently the optimization problem becomes a Least-Squares (LS) problem where the domain is $\mathbb{R}^{(N_{verts}-1)\times 3}$ – the space of meshes whose first point is the origin.

2. The interpretation of the quantity being minimized is the difference between the edges we want (for a nominal value of $\mathcal{Q}$) and the edges we can actually have in the deformed mesh.

3. In terms of graphs, the connectivity of these meshes is fixed, so the minimizer is always a valid mesh: regardless of what the value of the minimizer is, we always connect its points in the same order.

4. The difference between $\mathcal{Q}$ and $\mathcal{Q}_{\mathcal{M}}^{\text{extracted}}$ is distributed over all of the triangles.

5. If Q was computed from some mesh $\mathcal{M}$, then f is nullified. Moreover, the minimizer coincides with $\mathcal{M}$ modulo global translation and is unique (again, up to global translation).

6. Usually E is not square nor full-rank. If the meshes are of high-resolution[4] then E is quite large a matrix. In which case, as in our experiments, it is not practical to use its

[4]For example, in our experiments in Section 4.5 $N_{tri} \approx 15,000$, while in one of our experiments in Chapter 5 $N_{tri} = 50,000$.

Moore-Penrose pseudoinverse. Instead, we use a sparse solver that is pre-computed offline.

7. In our experiments, as well as in works by others who employ this paradigm, $\mathcal{Q}$ is sampled from a statistical model (to be discussed later), so the minimal value of f might be (and usually is) positive. The good news, however, is that these values are usually very small. We attribute that to reasons similar to ones mentioned in the 2D case.

8. There are several possible modifications to the optimization problem; *e.g.*, one can penalize differences between local deformations (in $\mathcal{Q}_{\mathcal{M}}^{\text{extracted}}$, not in $\mathcal{Q}$ which is regarded as fixed) that correspond to neighboring triangles. Be that as it may, from now on we will assume we have at our disposal *some* way to synthesize a mesh $\mathcal{M}$ from $\mathcal{Q}$, and that the resulting $\mathcal{Q}_{\mathcal{M}}^{\text{extracted}}$ is not too far from $\mathcal{Q}$ (at least for plausible values of $\mathcal{Q}$).

9. It is possible to restrict the space of triangle deformations in a way that that would ensure the existence of a valid mesh $\mathcal{M}$ satisfying $E\mathcal{M} = \mathcal{Q}\mathcal{T}$. In the context of building statistical models of deformations, however, this would significantly complicate these models. In contrast, the two-step approach described above (first deform the triangles, then find a mesh the can be reasonably related to the resulting triangle deformations) enables researchers, us included, to build simpler statistical models for deformations as no hard deterministic constraints are imposed on the relations between different deformations in the same mesh[5]. Our use of this approach too is motivated by this consideration. However, we have another important motivation for "keeping it simple"; more on that later.

[5] Of course, a model may capture the statistical relations between such deformations, *e.g.*, through either a global covariance or a probabilistic graphical model.

4.2.4 For Local Deformations, Using the Space of 3-by-3 Matrices is Unwise

Mathematically, there are numerous methods to solve the under-constrained system in Eqn. (4.12). For example one can encourage sparsity of Q. Another one is to favor a solution of the least ℓ^2 norm. In the particular context of mesh deformations, here are two existing approaches:

Sumner and Popović tackle the problem using the introduction of three fictitious constraints [142]. The authors heuristically add a "virtual" third edge (or fourth vertex) to each triangle, defined as $v_4 = v_1 \times v_2/\sqrt{|v_1 \times v_2|}$. Then, reinterpreting X and Y as elements of $\mathbb{R}^{3\times 3}$ instead of $\mathbb{R}^{3\times 2}$, they set $Q = YX^{-1}$. X and Y have positive determinant, and so Q is bound to be an invertible matrix of positive determinant; this is a plus. Their motivation to scale the cross-product by the reciprocal of the square root of its length is so that length of v_4 will be proportional to the length of the triangle edges. Note this decision is arbitrary and that different choices for the length would lead to different solutions. Moreover, this arbitrary decision determines the Euclidean distance between different deformations, which in turns affects statistical modeling.

Bălan *et al*. deal with the ambiguity by solving for all $\{Q_i\}_{i=1}^{N_{tri}}$ at once using a smoothness prior that penalizes (Euclidean) differences between deformations of neighboring triangles [16]. In their method, an hyper-parameter for the smoothing prior needs to be determined and there are no guarantees that the resulting deformations will not have zero or negative determinants. This is not just a theoretical issue: using Bălan's code on the same data used in [16], up to 10% (the exact percentage depends on the value the hyper-parameter) of the local deformations end up having negative determinant. Additionally, due to the tradeoff between the data fidelity term and the smoothness term, the resulting minimizer $\mathcal{Q}_{\mathcal{M}}^{\text{extracted}}$ causes the deformed edges $\mathcal{Q}_{\mathcal{M}}^{\text{extracted}}\mathcal{X}$ to (usually) deviate from the observed edges $\mathcal{Y}$; the amount of deviation depends again on the hyper-parameter. This means that with this method, an observed mesh $\mathcal{M}$ cannot be accurately recovered from its

extracted deformation $\mathcal{Q}_{\mathcal{M}}^{\text{extracted}}$. This is worrisome: while it is desirable to allow imperfect fit of a *statistical model* to the training data (to avoid over-fitting), the fact that *training meshes* cannot be accurately reproduced from *training deformations* should be seen as a shortcoming of their method.

What all of the aforementioned four methods, as well as many other representations of local deformations, have in common is that they treat deformation matrices as *Euclidean local deformations*.

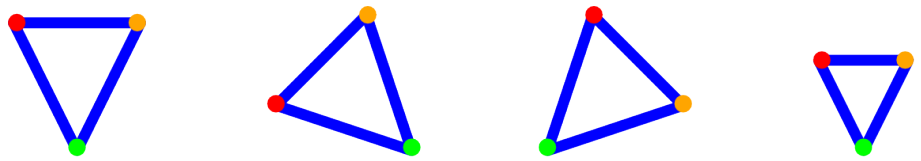
Definition 4.2.2 (Euclidean Local Deformations)**.** Any 3-by-3 matrix Q, when regarded as an element of the Euclidean space $\mathbb{R}^{3\times3}$, is called a *Euclidean Local Deformation*.

Note that in Definition 4.2.2, the existence of two (nondegenerate) triangles X and Y satisfying Eqn. (4.8) (*i.e.*, $Y = QX$) is not required. In fact, even if X is a known triangle, the existence of Y satisfying that equation is not required. For a trivial example, note that the $O_{3\times3}$ is a Euclidean deformation. Additionally, a Euclidean deformation might have a determinant which is negative or even zero (*e.g.*, consider the two trivial examples: $Q = -I_{3\times3}$ and $Q = O_{3\times3}$). Finally, note that the Euclidean space $\mathbb{R}^{3\times3}$ is *a linear space with 9 DoF*.

Remark 4.2.2 (Euclidean local deformations *vs*. locally-Euclidean spaces). Definition 4.2.2 has nothing to do with locally Euclidean spaces (see Definition 2.4.1, page 52).

Besides the fact that there are only 6 DoF in Eqn. (4.12), treating local deformations as Euclidean has many disadvantages. In particular, the Euclidean-space assumption is propagated to the way *triangle deformations* are being synthesized (*e.g.*, through linear combinations), and consequently, to the way *mesh deformations* are being synthesized[6]. For example, recall Fig. 1.4d (page 13): the mesh depicted there is synthesized from a mesh deformation which in turn is synthesized by a linear combination: $(\mathcal{Q}_1 + \mathcal{Q}_2)/2$. As we noted then, the displeasing resulting shape looked unnaturally "squashed". An example

[6]The emphasis on the terms "local deformations" and "mesh deformation" is to contrast synthesis of triangle deformations or mesh deformations with synthesis of triangles or meshes.



(a) X (b) Q_1X (c) Q_2X (d) $\frac{Q_1+Q_2}{2}X$

Figure 4.7: The Euclidean averaging of local deformations. Here, $\theta = \pi/4$ so $c = \cos(\pi/4) = \sin(\pi/4) \approx 0.7 < 1$. Note the shrinking effect caused by applying the average deformation. The different colors of the points indicate the point-to-point correspondence between the triangles. See text for more details.

at the individual-triangle level will help to understand where this effect comes from.

Example 4.2.2 (Averaging Euclidean local deformations)**.** Let $\theta \in [0, 2\pi/2]$, $c = \cos\theta$, and $s = \sin\theta$. Set the two local deformations Q_1 and Q_2 as follows:

$$Q_1 = \begin{pmatrix} c & -s & 0 \\ s & c & 0 \\ 0 & 0 & 1 \end{pmatrix}, Q_2 = \begin{pmatrix} c & s & 0 \\ -s & c & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{4.15}$$

If a triangle $X \in \mathbb{R}^{3\times 2}$ lies completely in the xy-plane (*i.e.*, the third row is $[0, 0]$), then Q_1 (or Q_2) rotates X counterclockwise (respectively, clockwise) about the z-axis by the angle θ. These deformations, Q_1 and Q_2, are just the opposite of each other. Thus, in terms of their average deformation effect, one would like them to cancel each other. In other words, we would like their average to be the identity matrix, $I_{3\times 3}$. Simple arithmetic, however, shows us that

$$\frac{Q_1 + Q_2}{2} = \begin{pmatrix} c & 0 & 0 \\ 0 & c & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{4.16}$$

The result is a shrinking matrix: as θ goes from 0 to $\pi/2$, the value of c gets smaller and smaller till it vanishes. See Fig. 4.7 for an illustration.

Analogously to our early discussion in Section 1.2.2, the failure of Euclidean averaging in Example 4.2.2 suggests that there is a problem with not only the linear structure of

$\mathbb{R}^{3\times 2}$ but also with the distances associated with it[7].

The issue of redundant DoF is also worrisome. For example, a model that tries to capture as much variance as possible should not waste resources on variance inside *equivalent classes* of Euclidean local deformations. Namely, for a given triplet (X, Y, Q) satisfying $Y = QX$, the model should ignore the variability inside

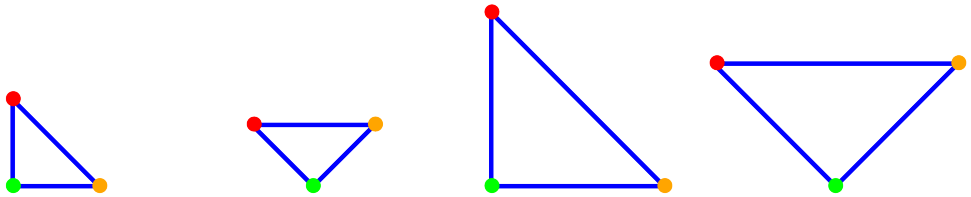
$$[Q] \stackrel{\text{def}}{=} [\widetilde{Q} : Y = \widetilde{Q}X] \subset \mathbb{R}^{3\times 3} \,. \tag{4.17}$$

The aforementioned heuristic suggested by Sumner and Popović in [142] does provide a way to establish the 3 redundant DoF during *deformation analysis*. Additionally, as discussed above, the resulting Q's are bound to be elements $\mathrm{GL}^+(3)$ (see Definition 2.3.6, page 34). However, when these Q's are regarded as Euclidean local deformations (as done, *e.g.*, in [6, 14, 142]) there are still 9 DoF, the space is still linear, and there is no guarantee that a synthesized Euclidean local deformation will have a positive (or even nonzero) determinant.

In addition to the problems mentioned above – related to linearity, distances, DoF, and determinants – there is another subtle problem: standard distances associated with $\mathbb{R}^{3\times 3}$ lack left-invariance.

Example 4.2.3 (Left-invariance fails for Euclidean local deformations)**.** This example, related to triangle deformations, is a modification of an example related to image deformations from Learned-Miller and Chefd'hotel [104]. Suppose Q_1, Q_2, and Q_3, have the effects shown in Fig. 4.8, where for the purpose of illustration we again assume all triangles are completely contained in the xy-plane. Without worrying too much about the exact values of Q_1 and Q_2, let us assume that Q_3 is a uniform scale matrix (see Definition 2.3.7, page

[7] Note that in this example, if we think of the midpoint between two deformations in terms of minimizing the sum of distances (or squared distances) from them, then it does not matter whether we use the ℓ^2 distance or any other of the distances appearing in Eqns. (1.5)-(1.7).



(a) Q_1X (b) Q_2X (c) Q_3Q_1X (d) Q_3Q_2X

Figure 4.8: Left-invariance fails for Euclidean deformations: with X as in Fig. 4.7a, any standard distance on Euclidean deformations fails to satisfy the equality $\text{dist}(Q_1, Q_2) = \text{dist}(Q_3Q_1, Q_2Q_2)$. This is unfortunate: the transformation between the triangles in (a) and (b) (*i.e.*, a rotation by 45 degrees) is exactly the same as the one between (c) and (d). The different colors of the points indicate the point-to-point correspondence between the triangles. See text for more details.

34) where the scale $S > 1$. It follows that

$$\begin{aligned} d(Q_3Q_1, Q_3Q_2)_F &= \|Q_3Q_1 - Q_3Q_2\|_F \quad \text{(by def.)} \\ &= \|SQ_1 - SQ_2\|_F \\ &= S\,\|Q_1 - Q_2\|_F \quad \text{(by positive homogeneity of the norm)} \\ &= Sd(Q_1, Q_2)_F \quad \text{(by def.)} \\ &\neq d(Q_1, Q_2)_F\ . \end{aligned} \tag{4.18}$$

Remark 4.2.3 (Left-invariance *vs*. scale-invariance). Note that left-invariance should not be confused with scale-invariance. A scale-invariance distance is one that satisfies $\text{dist}(Q, SQ)$ for any deformation matrix Q and any positive real number S. Also, while scale-invariance implies $\text{dist}(Q_1, Q_2) = \text{dist}(SQ_1, SQ_2)$, we stress that in Example 4.2.3 we happened to pick Q_3 to be a scaling matrix solely for the ease of illustration.

The fact that linear structure and Euclidean distances are inappropriate for deformations hurt statistical models that are built on them; *e.g.*, as Euclidean distance is not suitable for measuring differences between deformations, it follows that statistical analysis with standard PCA is compromised.

Of course, all of the problems mentioned in this section are trivially propagated from

Euclidean local deformations to *Euclidean mesh deformations*; *i.e.*, the space of sparse $3N_{tri}$-by-N_{tri} matrices, whose only possible nonzero entries can appear in 3-by-3 blocks along its main diagonal, when regarded as elements of the Euclidean space $\mathbb{R}^{3N_{tri}\times 3N_{tri}}$.

Existing statistical models of mesh deformations are built on Euclidean deformations [6,14,48,62,72]. To large extent, this is true even for Hasler *et al.* [63], who use a nonlinear over-parameterization with 15 DoF but still make use of Euclidean distances and the linear structure of $\mathbb{R}^{3\times 3}$. In particular, these parametric models (with [48] excluded) use linear-subspace techniques such as PCA to reduce the dimensionality. Besides the fact that the linear space structure is inappropriate for mesh deformation (and hence, so is the linear structure of these subspaces), *these subspaces are learned with respect to problematic distance measures.*

Back to the drawing board.

4.3 A Novel Representation of Triangle Deformations

Our first observation is that if local deformations are treated as elements of the matrix Lie group $\mathrm{GL}^+(3)$ (see Definition 2.3.6, page 34), many of the aforementioned problems will be avoided. Having said that, the issue of redundant DoF still stands: the nine-dimensional manifold $\mathrm{GL}^+(3)$ is too large a space. Additionally, as we shall see, there are several concerning computational issues associated with picking $\mathrm{GL}^+(3)$.

Fortunately, there is even better a matrix Lie group.

We now show how local deformations are fully described by a six-dimensional space. Instead of working in either the nine-dimensional linear space $\mathbb{R}^{3\times 3}$ or the (smaller) nine-dimensional nonlinear space $\mathrm{GL}^+(3)$, we explicitly work in an appropriate nonlinear six-dimensional space. Moreover, the space is a matrix Lie group. The idea is based on the observation that any triangle X can be deformed to any other triangle Y by a combination

of uniform scaling, a particular in-plane deformation, and a 3D rotation. See Fig. 4.3 for an illustration.

We will show that a group structure is important for the statistical analysis of shape deformations and to enable the principled combination of these deformations. We argue for a new type of deformation to appropriately model 3D shape in triangulated meshes. These deformations do not suffer from the problems of Euclidean deformations, have a group structure, and give rise to a meaningful distance. This distance is also left-invariant. Moreover, this group lends itself to easy computations of the matrix exponential logarithm, geodesic distances, geodesic paths and manifold-valued statistics.

4.3.1 The Three Basic Components

The first and simplest component of our representation, scaling, is defined by the group of scales (see Definition 2.3.7, page 34). As done throughout this work, we use G_S to denote US(1).

A second component models a particular type of in-plane deformation. Its definition is best understood through the notion of canonical triangles.

Definition 4.3.1 (Canonical triangle)**.** A triangle $[v_1, v_2]$ is said to be *canonical* (or in a *canonical position*) if

$$[v_1, v_2] = \begin{bmatrix} x_1 & x_2 \\ 0 & y_2 \\ 0 & 0 \end{bmatrix} \; ; \; x_1 > 0, \; y_2 > 0, x_2 \in \mathbb{R} \; . \tag{4.19}$$

In other words, v_1 lies on the positive x-axis, and v_2 is in the upper open half of the xy-plane (see the canonical triangles in Fig. 4.3).

We now define our in-plane deformations, acting on canonical triangles.

Definition 4.3.2. $G_A \stackrel{\text{def}}{=} \left\{A \in \mathrm{GL}(2) : A[1,0]^T = [1,0]^T, \det A > 0\right\}$. Equivalently,

$$G_A \stackrel{\text{def}}{=} \left\{ A = \begin{bmatrix} 1 & U & 0 \\ 0 & V & 0 \\ 0 & 0 & 1 \end{bmatrix} : U \in \mathbb{R}, V > 0 \right\}. \tag{4.20}$$

Proposition 4.3.1. *G_A is a subgroup of* GL(3).

Proof. The identity matrix, $I_{3\times 3}$, is in G_A. To prove closure under composition, let $A, B \in G_A$. Note that $AB[1,0,0]^T = A[1,0,0]^T = [1,0,0]^T$ and $\det AB = \det A \det B > 0$. Consequently, $AB \in G_A$. To prove closure under inversion, let $A \in G_A$. First, note that $\det A^{-1} = 1/\det A > 0$ since $\det A > 0$. Second,

$$A^{-1} = \begin{bmatrix} 1 & -U/V & 0 \\ 0 & 1/V & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.21}$$

and thus $A^{-1} \in G_A$. □

By a slight abuse of notation, we can also regard G_A as a subgroup of GL(2), using the bijection

$$\begin{bmatrix} 1 & U \\ 0 & V \end{bmatrix} \leftrightarrow \begin{bmatrix} 1 & U & 0 \\ 0 & V & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.22}$$

Remark 4.3.1. This remark is a digression and can be skipped. There is also an obvious correspondence between G_A (when regarded as as a subgroup or GL(2)) and the identity component of the two-dimensional affine group:

$$\begin{bmatrix} 1 & U \\ 0 & V \end{bmatrix} \leftrightarrow \begin{bmatrix} V & U \\ 0 & 1 \end{bmatrix}. \tag{4.23}$$

In fact, these groups are isomorphic. Consequently, many readily-available results of the

affine group – a well studied object – apply transparently to G_A. For example, this shows immediately that G_A is not unimodular.

Note that G_A is neither Abelian (see Definition 2.3.3, page 33) nor it is compact[8]. What we care most about G_A is the geometrical implication of its action on canonical triangles: if $A \in G_A$ and X is canonical, then AX is canonical too. Note that the first edge (column) of X equals the first edge of AX (see Fig. 4.3); *i.e.*, $[1, 0, 0]^T$ is a right eigenvector of A with an eigenvalue 1. We also have the following result.

Proposition 4.3.2. *If X and Y are two canonical triangles, then there exists a unique $(A, S) \in G_S \times G_A$ such that $Y = ASX$.*

Proof. Let $X = [v_1^{(X)}, v_2^{(X)}]$ and $Y = [v_1^{(Y)}, v_2^{(Y)}]$. Set $S = ||v_1^{(Y)}||/||v_1^{(X)}||$. Without loss of generality assume $S = 1$. Let $v_2^{(X)} = [x_2^{(X)}, y_2^{(X)}]^T$ and $v_2^{(Y)} = [x_2^{(Y)}, y_2^{(Y)}]^T$. Now solve for the unknowns U and V (there exists a unique solution: the y's are positive):

$$\overbrace{\begin{bmatrix} 1 & U \\ 0 & V \end{bmatrix}}^{A \in G_A} \begin{bmatrix} x_2^{(X)} \\ y_2^{(X)} \end{bmatrix} = \begin{bmatrix} x_2^{(Y)} \\ y_2^{(Y)} \end{bmatrix} \Rightarrow V = \frac{y_2^{(Y)}}{y_2^{(X)}},\ U = \frac{x_2^{(Y)} - x_2^{(X)}}{y_2^{(X)}}\ . \tag{4.24}$$

□

This is illustrated in Fig. 4.3.

Remark 4.3.2. In terms of group theory, the existence aspect of Proposition 4.3.2 tells us that $G_A \times G_S$ acts *transitively* on the background space of canonical triangles. Namely, we can always move from one canonical triangle to another. As an aside remark, since the action is also continuous, this means that this background space is homogeneous.

Of course, not all triangles are canonical, hence we have the third and final component: SO(3), the rotation group (see Definition 2.3.11, page 35).

[8]The (non)compactness is with respect to the topology G_A inherits from $\mathbb{R}^{3\times3}$ as its subspace.

Fact 4.3.1. If X is a triangle, then we can always find a rotation matrix R_X such that $R_X X$ is canonical. R_X depends on X.

Proof. Let $X = [v_1, v_2] \in \mathbb{R}^{3\times 2}$. First, use any standard technique to find $R_1 \in \mathrm{SO}(3)$ such that $R_1 v_1 = \|v_1\|\,[1,0,0]^T$. Let $[x,y,z]^T$ denote the entries of $R_1 v_2$. Then, solve for the unknowns c and s in

$$\begin{bmatrix} c & -s \\ s & c \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix} = \begin{bmatrix} +\sqrt{y^2+z^2} \\ 0 \end{bmatrix} \tag{4.25}$$

and set

$$R_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c & -s \\ 0 & s & c \end{bmatrix} R_1$$

(note that $c^2 + s^2 = 1$). □

At last, the entire story in Fig. 4.3 is complete: if X and Y are two arbitrary triangles, then the fact above and Proposition 4.3.2 imply we can always find rotation matrices R_X and R_Y such that $R_X X$ and $R_Y Y$ are canonical, and a unique $(A, S) \in G_A \times G_S$ such that $R_Y Y = A S R_X X$. Equivalently, $Y = R_Y^T A S R_X X$. Setting $R \overset{\text{def}}{=} R_Y^T R_X$ (so $R \in \mathrm{SO}(3)$), yields our new *triangle deformation equation*:

$$\boxed{Y = R R_X^T A S R_X X}\,. \tag{4.26}$$

Consider X as belonging to the template $\mathcal{T}$. Thus, X and R_X are fixed. As Y varies, the triplet (R, A, S) has 6 DoF: 3 for R, 2 for A and 1 for S. By construction,

$$\det(R R_X^T A S R_X) = V S^3 > 0\,. \tag{4.27}$$

We have thus found an invertible local deformation

$$Q = R R_X^T A S R_X \tag{4.28}$$

deforming one triangle to another and this matrix has only 6 DoF. Note that unlike the traditional Euclidean approach, with 9 DoF, we do not resort to any heuristic to deal with excessive DoF. Furthermore, the fact that the determinant is strictly positive eliminates deformations that have no physical meaning such as reflections.

Remark 4.3.3. Note that instead of the scalar S we could have used a 3×3 matrix:

$$\begin{pmatrix} S & 0 & 0 \\ 0 & S & 0 \\ 0 & 0 & \rho \end{pmatrix}, \rho \in \mathbb{R}^+ . \tag{4.29}$$

This would have changed nothing (except the exact value of the positive determinant of Q) as, for canonical triangles, what happens outside the plane is immaterial; *i.e.*, unlike the method in [142], we make no arbitrary decision about what happens in the direction which is normal to the plane of a triangle. If ρ is fixed across all triangles, then it has also no effect on the geodesic distance (to be defined later) between different Q's. In turn, this means that a statistical model (of these Q's) is not affected. If for some reason, however, one is interested in weighting deformations according to the size of either X or Y (S only captures the relative scale change between them), then one can set ρ to be a function of these quantities. This will have no effect on the resulting action of Q, but it will change the distance between different Q's as well as the statistical model. Finally, in our experiments we set $\rho = 1$, for the following practical reason: mathematically, the z-coordinate of $R_X X$ is zero. Due to numerical errors caused by machine-precision, however, the actual value might deviate from zero. If S happens to be large, this error will be amplified. By setting ρ to 1 (or even to 0), we avoid amplifying this numerical error.

4.3.2 The Matrix Lie Groups

4.3.2.1 The Matrix Lie Group of Triangle Deformations

There still remains one problem. The set

$$\left\{RR_X^T ASR_X : R \in \mathrm{SO}(3),\ A \in G_A,\ S \in G_S\right\} \tag{4.30}$$

does not form a matrix Lie subgroup of GL(3) (although it is a proper subset of GL(3)); *e.g.*, in obvious notation, if $R_1R_X^T A_1S_1R_X$ and $R_2R_X^T A_2S_2R_X$ are two elements in this set, it may not be possible to find a third element, $R_3R_X^T A_3S_3R_X$ satisfying

$$R_1R_X^T A_1S_1R_XR_2R_X^T A_2S_2R_X = R_3R_X^T A_3S_3R_X\ .$$

Fortunately, this is easily fixed by defining a new group:

Definition 4.3.3 (The triangle deformation group)**.** The *triangle deformation group*, denoted by G_T, is the set of triplets

$$\{(R, A, S) : R \in \mathrm{SO}(3), A \in G_A, S \in G_S\} \tag{4.31}$$

together with the composition map, $G_T \times G_T \to G_T$,

$$((R_1, A_1, S_1), (R_2, A_2, S_3)) \mapsto (R_1R_2, A_1A_2, S_1S_2)\ . \tag{4.32}$$

The group G_T is a direct product of SO(3), G_A, and G_S and so it too is a matrix Lie group; see Section 2.3.3. It is not Abelian (because of both SO(3) and G_A) nor compact (because of both G_A and G_S), and has has a natural identification with a six-dimensional matrix Lie group – which is a matrix Lie subgroup of the GL(6) whose dimension is 36 –

using the bijection:

$$(R, A, S) \leftrightarrow \begin{bmatrix} R & 0_{3\times 2} & 0_{3\times 1} \\ 0_{2\times 3} & A & 0_{2\times 1} \\ 0_{1\times 3} & 0_{1\times 2} & S \end{bmatrix} \in \mathbb{R}^{6\times 6} \tag{4.33}$$

(where we here regard A as a 2-by-2 matrix).

4.3.2.2 The Matrix Lie Group of Mesh Deformations

So far we have discussed the deformation of a single triangle. For a mesh of N_{tri} triangles, we use a direct product to represent the Lie group of triangular mesh deformations: $M \stackrel{\text{def}}{=} G_T^{N_{tri}}$, rendering M a matrix Lie group of dimension $6N_{tri}$. Recall that a group is usually denoted by G and a manifold is usually denoted by M. A matrix Lie group is both, so either notation may be used. Here we choose M to emphasize the nonlinearity manifold structure of mesh deformations.

We use the term M-valued to describe elements in that manifold. A point $p \in M$ corresponds to a mesh deformation $\mathcal{Q}$ as in Eqn. (4.9). The only difference from that equation is that now the $\{Q_i\}_{i=1}^{N_{tri}}$ that comprise $\mathcal{Q}$ have form as in Eqn. (4.28). The structure of M gives us group closure (ensuring a consistent representation) as well as a meaningful measure of distance between shapes.

As was discussed in Chapter 2, when working with matrix Lie groups (or any Lie groups for that matter), their Lie algebras (see Definition 2.3.14, page 41) play an indispensable tool.

4.3.3 The Lie Algebras

4.3.3.1 The Lie Algebra of Triangle Deformations

We now describe $\mathfrak{g}_T \stackrel{\text{def}}{=} \exp^{-1}(G_T)$, the Lie algebra, of G_T. As G_T is a direct product of three matrix Lie groups, $\mathfrak{g}_T$ is defined analogously using the direct product of their three Lie algebras.

Definition 4.3.4 (The Lie algebra of triangle deformations)**.** The *Lie algebra of triangle deformations* is given by

$$\mathfrak{g}_T = \mathrm{so}(3) \times \mathfrak{g}_A \times \mathfrak{g}_S \ . \tag{4.34}$$

The Lie algebras so(3) and $\mathfrak{g}_S$ were defined earlier (see Definition 2.3.19, page 42, and Definition 2.3.17, page 42), while $\mathfrak{g}_A \stackrel{\text{def}}{=} \exp^{-1}(G_A)$ is to be described soon. For $\mathfrak{gl}(3)$ (the Lie algebra of $\mathrm{GL}^+(3)$), the computation of the matrix exponential (see Definition 2.3.13, page 40) usually involves an infinite sum. Likewise for the matrix logarithm. In several cases, however, it is possible to derive closed-form formulas; e.g., for $\exp : \mathrm{so}(3) \to \mathrm{SO}(3)$ and $\log : \mathrm{SO}(3) \to \mathrm{so}(3)$, computations are given by the well known Rodrigues' formula [110]. While only the first map is surjective, the pair does form a local bijection around $0_{3\times 3}$ in so(3) the $I_{3\times 3}$ in SO(3).

For $\mathfrak{g}_A$ and G_A, we have the following result, where for convenience we regard G_A as a subgroup of GL(2).

Proposition 4.3.3. $\mathfrak{g}_A \stackrel{def}{=} \exp^{-1}(G_A)$*, is given by*

$$\mathfrak{g}_A = \left\{ A \in \mathfrak{gl}(2) : g = \begin{bmatrix} 0 & u \\ 0 & v \end{bmatrix} \right\} . \tag{4.35}$$

The maps $\exp : \mathfrak{g}_A \to G_A$ *and* $\log : G_A \to \mathfrak{g}_A$ *are given in closed-form; together, they form a bijection.*

Proof. First, assume $v = 0$. This implies A^2 is the zero matrix and thus so is A^k for every integer $k \geq 2$. By Definition 2.3.13, page 40,

$$\exp(A) = \exp\left(\begin{bmatrix} 0 & u \\ 0 & 0 \end{bmatrix}\right) = I + \begin{bmatrix} 0 & u \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & u \\ 0 & 1 \end{bmatrix} \stackrel{\text{def}}{=} \begin{bmatrix} 1 & U \\ 0 & V \end{bmatrix} \in G_A \,. \tag{4.36}$$

If $v \neq 0$, induction shows that

$$A^n = \begin{bmatrix} 0 & u \\ 0 & v \end{bmatrix}^n = \begin{bmatrix} 0 & uv^{n-1} \\ 0 & v^n \end{bmatrix} \,. \tag{4.37}$$

Consequently,

$$\exp(A) = \exp\left(\begin{bmatrix} 0 & u \\ 0 & v \end{bmatrix}\right) = \begin{bmatrix} 1 & \frac{u}{v}\left(e^v - 1\right) \\ 0 & e^v \end{bmatrix} \stackrel{\text{def}}{=} \begin{bmatrix} 1 & U \\ 0 & V \end{bmatrix} \in G_A \,, \tag{4.38}$$

where in the second equality we have used Definition 2.3.13 and simple calculations of limits. The two cases taken together imply a bijection $(u, v) \mapsto (U, V)$, $\mathbb{R}^2 \to \mathbb{R} \times \mathbb{R}^+$, and thus $\exp : \mathfrak{g}_A \to: G_A$ is bijective too. Finally, For computing the log, set $v = \log(V)$. If $V = 1$, set $u = U$. Otherwise, set $u = Uv/(V - 1)$. □

Proposition 4.3.3 is important for not only ensuring bijectivity but also providing simple and exact closed-form formulas. For example, generic expm and logm functions in Matlab or SciPy use the Padé approximation [69], are slower, and are not easily vectorized. Since such computations are needed frequently and apply to all N_{tri} triangles, this makes their use impractical here[9].

[9] Of course the computations can be parallelized as they apply to each triangle independently; however, approximation errors often lead to losing the group structure as the result falls outside the manifold (or Lie algebra).

Analogously to Eqn. (4.33), we identify $\mathfrak{g}_T$ with a Lie subalgebra of $\mathfrak{gl}(6)$:

$$\left(\begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_z & 0 \end{bmatrix}, \begin{bmatrix} 0 & u \\ 0 & v \end{bmatrix}, s \right) \leftrightarrow \begin{bmatrix} 0 & -\omega_z & \omega_y & 0 & 0 & 0 \\ \omega_z & 0 & -\omega_x & 0 & 0 & 0 \\ -\omega_y & \omega_z & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & u & 0 \\ 0 & 0 & 0 & 0 & v & 0 \\ 0 & 0 & 0 & 0 & 0 & s \end{bmatrix} \leftrightarrow \begin{bmatrix} \omega_x \\ \omega_w \\ \omega_z \\ u \\ v \\ s] \end{bmatrix} \tag{4.39}$$

where here the second "$\leftrightarrow$" indicates an identification of an element of $\mathfrak{g}_T$ with a six-dimensional vector. Finally, the map $\exp : \mathfrak{g}_T \to G_T$ is defined by the product map:

$$(\exp : \mathrm{so}(3) \to \mathrm{SO}(3),\ \exp : \mathfrak{g}_A \to G_A,\ \exp : \mathfrak{g}_S \to G_S)\ . \tag{4.40}$$

4.3.3.2 The Lie Algebra of Mesh Deformations

Lie algebra of M is given by $\mathfrak{m} \stackrel{\text{def}}{=} \mathfrak{g}_T^{N_{tri}}$ while the product map $\exp : \mathfrak{m} \to M$ is given by an N_{tri}-tuple of $\exp : \mathfrak{g}_T \to G_T$ maps. The corresponding logarithm is defined in a similar way. Let $p, q \in M$. We use the Lie algebra, interpreted as $T_I M$, as an intermediate step for moving from M to another tangent space $T_p M$; see Section 2.3.7.2. For example, here are the steps for mapping q to $T_p M$:

$$p^{-1}q \in M \qquad \text{(group closure)}\ ; \tag{4.41}$$

$$\log(p^{-1}q) \in T_I M \qquad \text{(by the definition of the Lie algebra)}\ . \tag{4.42}$$

Since we identify (see Section 2.3.7.1) $T_I M$ with $T_p M$ we interpret the last result, $\log(p^{-1}q)$, as an element of $T_p M$. See Fig. 4.1 for an illustration. Conversely, if $x \in T_I M$ is interpreted

as an element of T_pM, then we map it to M as follows:

$$\exp(x) \in M \qquad \text{(by def. of the Lie algebra)} ; \tag{4.43}$$

$$p \exp(x) \in M \qquad \text{(group closure)} . \tag{4.44}$$

Note that:

1. The logarithm map here is the *Lie group* logarithm map, which, for a matrix Lie group, coincides with the matrix logarithm. We here use the approach that identifies all tangent spaces of M with T_IM; see Section 2.3.7.1. Since M is also a Riemannian manifold, we could also have used the *Riemannian* logarithm map in order to move from M to T_pM, when the latter is not identified as a copy of T_IM. The same remark applies to the exponential map.

2. In this chapter, we stick to the Lie-algebraic approach for such transitions (as well as for PGA) as it is easier to understand (and explain!) and keeps the reader's focus on the Lie group structure rather than on a particular Riemannian metric that M may be endowed with. In Chapter 5, we will introduce a new tool for statistics on manifolds and we will use the manifold presented here as an example manifold to which the tool applies. When we do that, we will adopt a Riemannian approach as it will have several advantages in that context.

4.3.4 Geodesic Distances

To measure distances between mesh deformations (*i.e.*, points in M), we use the distance from Eqn. (2.66); *i.e.*, if p and q are in M, then their distance is

$$d(p, q) = \left\| \log(p^{-1}q) \right\|_F . \tag{4.45}$$

Specifically, the block diagonal structure in Eqn. (4.33) enables easy computation: If $p_i = \{g_{i,j}\}_{j=1}^{N_{tri}} \in M$, $i \in \{1,2\}$ and $g_{i,j} \stackrel{\text{def}}{=} (R_i, G_i, S_i)_j \in G_T$, then

$$d(p_1,p_2)^2 = \sum_{j=1}^{N_{tri}} d_{G_T}(g_{1,j},g_{2,j})^2 \tag{4.46}$$

where, for $(g_1,g_2) \in G_T \times G_T$ (dropping the j notation),

$$d_{G_T}(g_1,g_2)^2 \stackrel{\text{def}}{=} \left\|\log(R_1^T R_2)\right\|_F^2 + \left\|\log(A_1^{-1}A_2)\right\|_F^2 + \left|\log(S_2/S_1)\right|^2 \ . \tag{4.47}$$

Remark 4.3.4. The first term is a geodesic distance on SO(3), while the third is a geodesic distance on G_S. The second term is not a geodesic distance on G_A, for elements of $\mathfrak{g}_A$ are usually not normal matrices; see Section 2.3.8.4. It is, however, a true metric (in the sense of a metric space) and a good proxy to a geodesic distance. This nuance applies transparently to d_{G_T}. By a slight abuse of terminology, we will still refer to both as geodesic distances. The same remark applies to the geodesic curves we will discuss momentarily.

Finally, note that d is left-invariant: $d(p_3p_1, p_3p_2) = d(p_1,p_2)$ for every p_1,p_2,p_3 in M. This often-desired property (*e.g.*, see Example 4.2.3) is especially attractive for models that factor deformations. For example, one may want the distance between the shapes of two people standing in one pose to be the same as the distance between the shapes of the same two people standing in a second pose.

4.3.5 Geodesic Paths

As in Eqn. (2.61), a geodesic path interpolating between $p_1,p_2 \in M$ is given in closed-form by

$$p(t) = p_1 \exp(t \log(p_1^{-1}p_2)) \ , \tag{4.48}$$

where $t \in [0,1]$, $p(0) = p_1$, $p(1) = p_2$. Taking $t < 0$ or $t > 1$ is extrapolation.

By now it should be clear how we generated the deformation from which we synthesized the mesh in Fig. 1.4e: we simply set $t = 0.5$ in the geodesic path equation. Compare this result with its Euclidean counterpart Fig. 1.4d. Examples for longer sequences can be seen in Figs. 4.9, 4.10, and 4.11. Note that the process illustrated in these figures are performed completely locally at the level of individual triangles; *i.e.*, no global information about kinematic tree structure or mesh segmentation are used. A movie containing additional results is available online at:

http://www.dam.brown.edu/people/freifeld/shapes/shape_interpolation.avi

For an explanation of the contents of this movie, please see the following file:

http://www.dam.brown.edu/people/freifeld/shapes/shape_interpolation_ReadMe.txt

In terms of interpolation, there are two limitations worth noting. Before explaining them, note that the Euclidean methods we saw earlier suffer from these limitations too.

1. The first is related rotational ambiguity. Given only two triangles and no additional information, the rotation component between them is not fully determined as there is ambiguity about the direction; *i.e.*, a θ-angle clockwise rotation about some axis produces the same triangle as a $(2\pi - \theta)$-counterclockwise rotation. We always pick that rotation that corresponds to an angle smaller than π. For applications for statistical modeling of human shape, we report this causes no problem, as body shape deformations are usually not greater than π and we work with pose-aligned data. For interpolation between arbitrary poses, however, the geodesic path might end up in the wrong direction. A possible solution, and one we tested successfully on human shape data in various poses, is to use information about the kinematic tree and mesh segmentation: once the global rotation of the body parts are factored out, the residual deformations do not contain rotations of angles greater then π. Another possible solution we have yet to explore, is to determine the direction by the deformations between neighboring triangles[10].

[10]The idea has been suggested to us by Yaron Lipman.

2. The second limitation is the possible self-intersection of articulated bodies. Again, this happens since the process is local (and thus, in particular, happens with the Euclidean approaches as well). One possible solution is to penalize self-intersection as done, *e.g.*, in [58].

We also note that due to the locality, our representation of mesh deformation does not preserve the 3D volume of the shape, in contrast to, *e.g.*, the interpolation method in [83]. However, we do not regard this as a limitation: when interpolating between shapes of, say, a thin person and a fat person of similar height, volume-preserving is not a desired property.

To conclude this section, we reiterate that our interest in interpolation is not for the sake of interpolation itself: rather, it provides us a way to visualize how much the Euclidean approach is based on the wrong type of distance, a fact that compromises the resulting statistical models. Additionally, the most naive approach to shape representation – whose disadvantages are mentioned in several places in this work – is to work directly on the vertices of the meshes. The average shape, in terms of vertices, of the two 3D meshes in Fig. 1.4b and Fig. 1.4c is practically indistinguishable (and thus is not shown) from the one produced from the average of Euclidean deformations shown in Fig. 1.4d (*i.e.*, just as bad). This is hardly a surprise as both approaches share the linear structure assumption.

4.4 The Statistics of Lie Shapes

In this chapter we focus on the comparison of a Lie-algebraic PGA model (see Section 2.6.1) and Euclidean PCA models. In Chapter 5 we will also use Riemannian PGA for the manifold introduced in this chapter, in order to demonstrate a new tool for statistics on manifolds.

Given a training set of shapes, human bodies in our case, $\{\mathcal{M}\}_{i=1}^{N}$, we extract $\{\mathcal{Q}_i\}_{i=1}^{N}$,

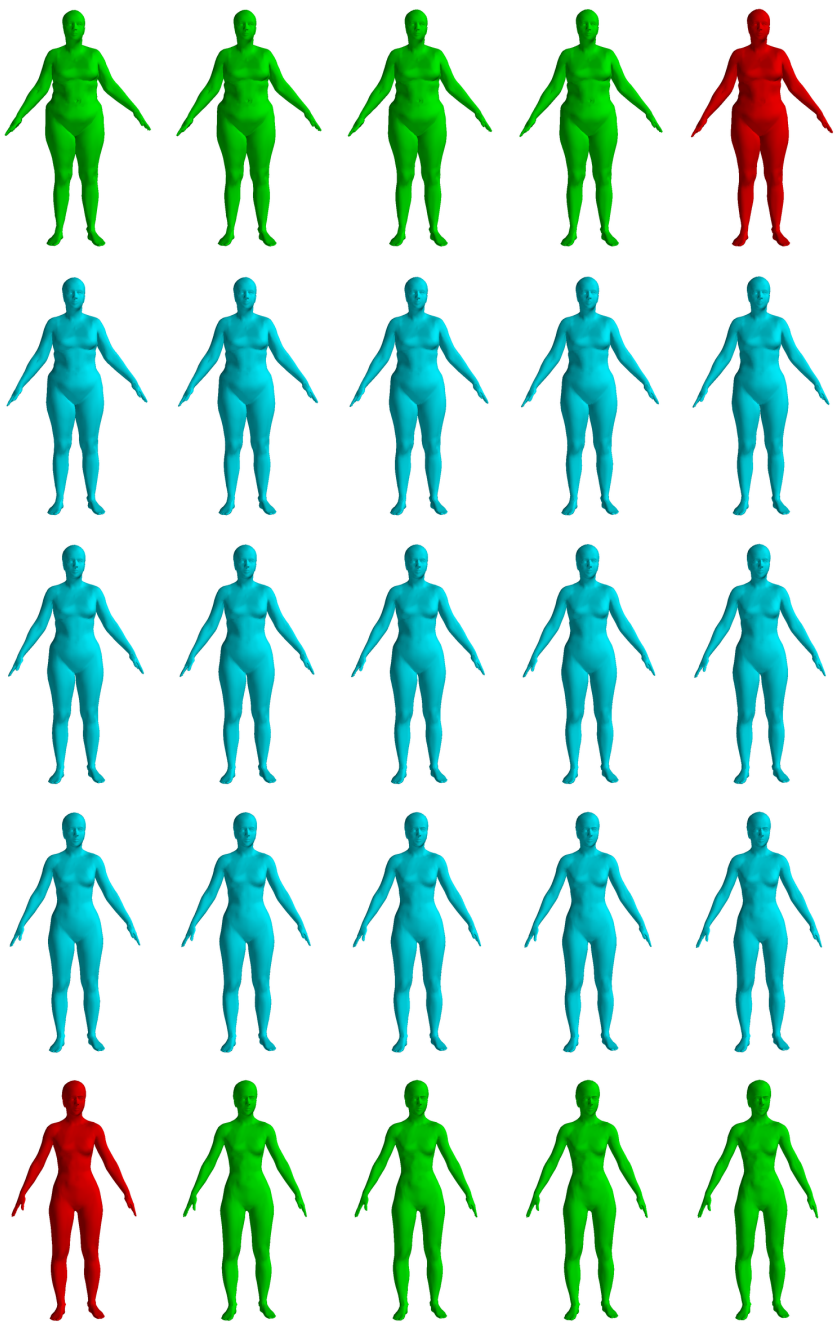


Figure 4.9: Geodesic interpolation of shape deformations. The sequence starts at the top-left corner and continues in a raster scan, from left to right, row by row. The sequence is generated from only two meshes: the red ones. These are two (aligned) laser scans of real people. Light blue stands for interpolation along a geodesic path between their deformations (with respect to a template not shown here). Green stands for extrapolation.

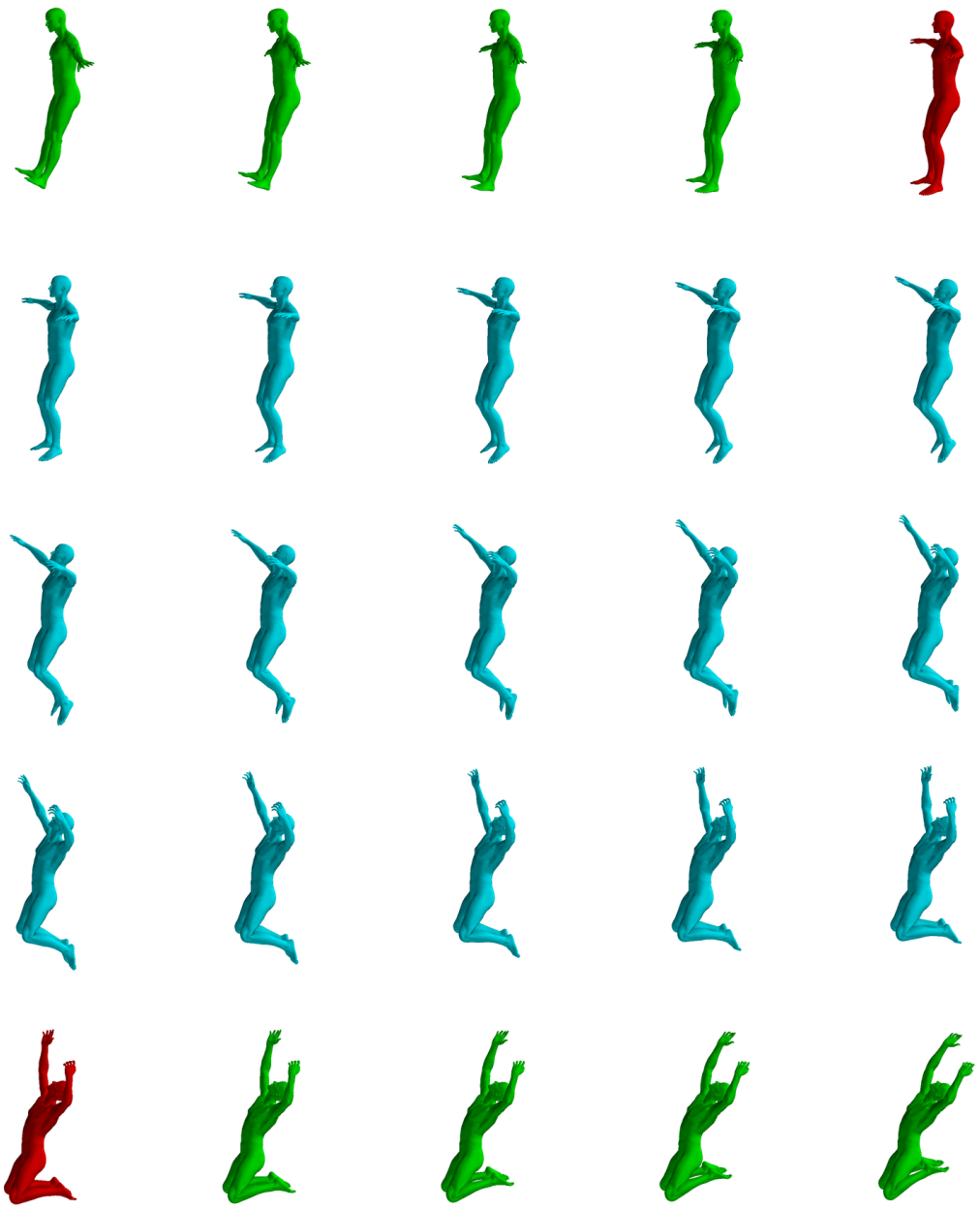


Figure 4.10: This sequence is analogous to Fig. 4.9. The only difference is that here the two red shapes that are used to generate the sequence are taken from the Technion Tosca dataset: http://tosca.cs.technion.ac.il/. Note that no global information about the kinematic tree structure or mesh segmentation is used.

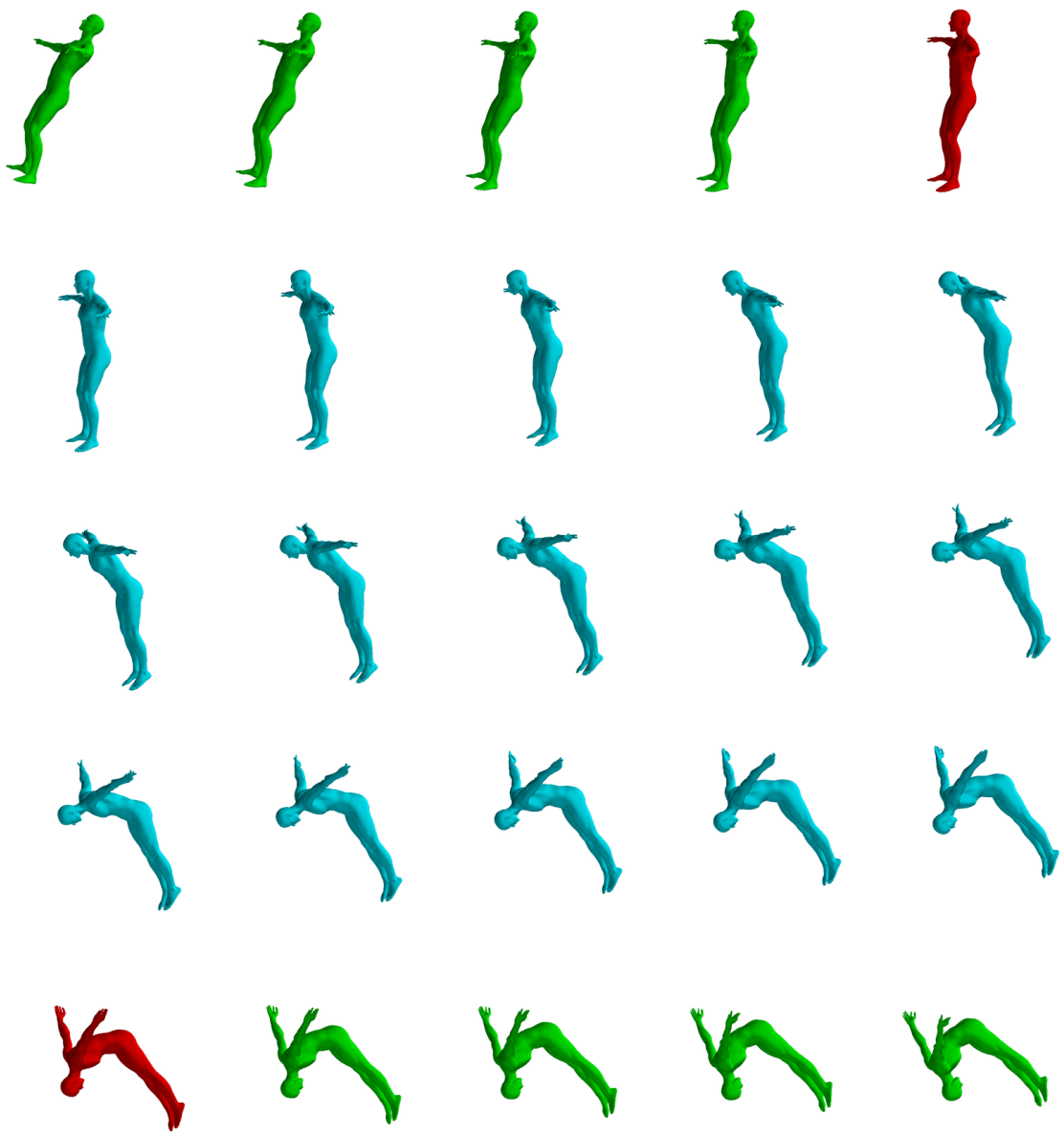


Figure 4.11: Another sequence generated in a similar way to the one from Fig. 4.10.

their mesh deformations from $\mathcal{T}$, using our (R, A, S) approach. Note once this is done, we may regard this deformation-dataset as either M-valued or Euclidean data. The latter just means that although we know that each one of the local Q's has form $Q = RR_X^T ASR_X X$, we can still regard this Q as a nine-dimensional Euclidean deformation; unlike in other Euclidean approaches, this Q was at least extracted in a heuristic-free way. To create a notation distinction, we denote the deformations, when regarded as M-valued, by $\{p_i\}_{i=1}^N \subset M$. In contrast, when regarded as Euclidean data, we keep the $\{\mathcal{Q}_i\}_{i=1}^N$ notation. Of course, the manifold approach is preferred, but this will be useful later when we compare with the standard Euclidean approach. Finally, and again for the purpose of comparison, we compute the Euclidean deformations, using the method from [14], and denote them by $\left\{\mathcal{Q}_i^{\text{Euclidean}}\right\}_{i=1}^N$.

We compute two Euclidean PCA models: one for $\{\mathcal{Q}_i\}_{i=1}^N$ and one for $\left\{\mathcal{Q}_i^{\text{Euclidean}}\right\}_{i=1}^N$. For $\{p_i\}_{i=1}^N$, we first compute the intrinsic mean using the algorithm from [113]. In our experiments convergence is typically reached with few iterations. Note this algorithm can can be applied to each one of the N_{tri} local triangles in parallel. This iterative algorithm is the only non-closed-form computation required in this chapter. Note, however, that there also exists a simple closed-form approximation: $\mu \approx \exp(1/N \sum_i \log p_i)$ [37].

Once the intrinsic mean (or more precisely, the Karcher mean), μ, is computed, we compute the Lie-algebraic PGA. Finally, for each one of these three models, we compute the corresponding coefficients, as defined by projecting the data on the resulting subspaces.

4.5 Experiments

4.5.1 Data

4.5.1.1 Mesh Data

We model a dataset of 986 body scans of adult women standing in a similar pose [120]. Because here we are interested in body shape, we remove the head and hands for the analysis but they can trivially be included (*e.g.*, in Chapter 5 we will work with full bodies). A template mesh with 16218 triangles is aligned to each of the scans and we then work with these aligned meshes.

4.5.1.2 Deformation Data

Given aligned triangles we compute the deformations (both M-valued and Euclidean) to produce three types of training deformation data (where $N = 986$):

1. $\{p_i\}_{i=1}^N$ (M-valued deformations);
2. $\{\mathcal{Q}_i\}_{i=1}^N$ (Euclidean Deformations);
3. $\left\{\mathcal{Q}^{\text{Euclidean}}\right\}_{i=1}^N$ (Euclidean Deformations).

These datasets were produced as was described in Section 4.4. We seek a parsimonious statistical representation of body shape variation where parsimony has two components: it implies that the model has low variance and low reconstruction error.

4.5.2 Variance Comparison

A direct comparison between the Euclidean variance of the manifold data and the $\mathcal{Q}^{\text{Euclidean}}$ data is not possible as these spaces are different. Instead, to fairly compare our representation with the existing Euclidean representation, we compare the variance of the $\mathcal{Q}$'s and the variance of $\mathcal{Q}^{\text{Euclidean}}$'s. Note these variances are measured the same *Euclidean* space: $(\mathbb{R}^{3\times 3})^{N_{tri}}$. We find that the total variance for the $\mathcal{Q}^{\text{Euclidean}}$'s is 1.68 times the total variance of the new deformations $\mathcal{Q}$'s. We attribute this to the fact that our method does not admit non-physical deformations, as well as ignoring variance in equivalent classes (see Eqn. (4.17) and its associated discussion regarding equivalence classes of deformations). This suggests that, even if one avoids a manifold approach, one is still better off computing M-valued deformations, encoding them as $Q = RR_X^T ASR_X$, and then working with these in a Euclidean space.

4.5.3 PCA, Lie-Algebraic PGA, and Reconstruction of Triangle Edges

Table 4.1: Mean edge RMS for mesh reconstruction using a subspace

#PC's	5	10	15	20	25	30	35	40	45	50	100
Euclidean method. Ave. RMS [mm]	2.71	2.53	2.43	2.34	2.28	2.23	2.19	2.15	2.11	2.08	1.91
Our method. Ave. RMS [mm]	2.57	2.43	2.32	2.26	2.21	2.17	2.12	2.09	2.06	2.03	1.88

In addition to low variance, a good representation must model the data. In particular, we seek a low-dimensional approximation of body shape variation that captures as much of the variance as possible. Consequently we evaluate our ability to reconstruct the data meshes using the Euclidean and our Manifold approaches. We compute the interior mean and PGA and show the first few eigenvectors in Fig. 4.12 We also compute Euclidean PCA for the Euclidean deformations.

We reconstruct all the meshes using the same number of components computed with PCA and Lie-Algebraic PGA. For each mesh $\mathcal{M}_i$ we compute the root mean squared (RMS)

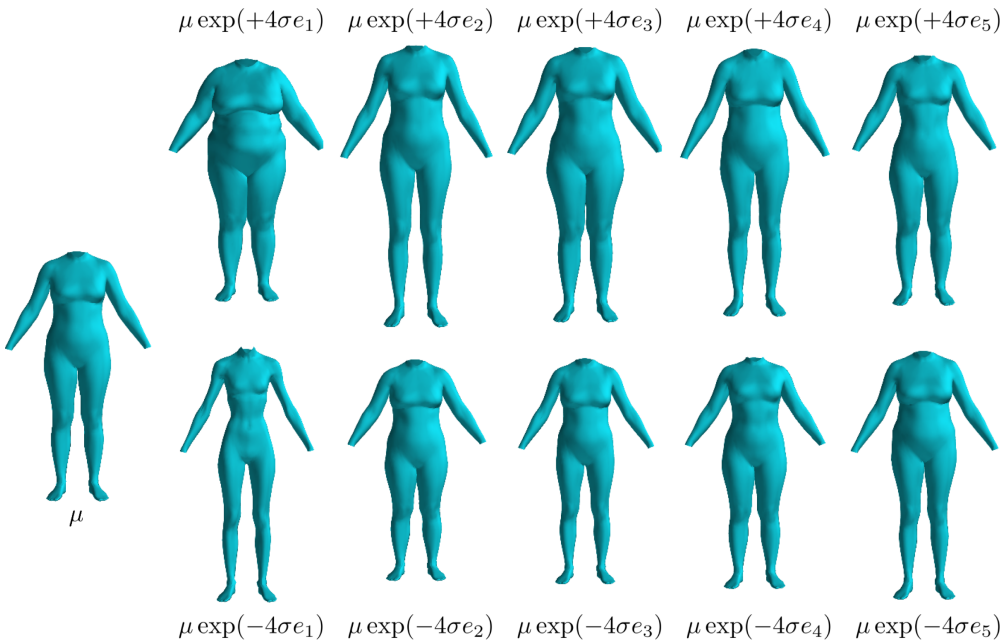


Figure 4.12: Lie-algebraic PGA on human shape data. The single shape in the leftmost column is μ. The other columns present, from left to right, the first five eigenvectors by showing $\mu \exp(+4\sigma e_i)$ (top) and $\mu \exp(-4\sigma e_i)$ (bottom), where e_i is the i^{th} eigenvector, $i \in \{1, 2, 3, 4, 5\}$.

reconstruction error across all edges in M_i and then average the result over all $\{\mathcal{M}_i\}$:

$$\text{Error} \overset{\text{def}}{=} \frac{1}{\#\text{edges}} \sum_{\text{edges}} \underbrace{\sqrt{\frac{1}{N} \sum_{\text{examples}} \|\text{true edge} - \text{recon. edge}\|_{\ell^2}^2}}_{\text{RMSE per edge}} \tag{4.49}$$

The results in Table 4.1 show a lower error for every number of basis vectors used. Note this is despite the fact that RMS is a Euclidean error. The fact that our representation is doing better in terms of reconstruction is perhaps more interesting than it may seem at first sight. As mentioned in Section 2.5.3.1, PCA can be generalized to manifolds in two ways, defining two optimization problems over the space of geodesic subspaces: variance maximization (PGA) and reconstruction-error minimization (GPCA). Unlike in the Euclidean case, the optimizers of these two problems need not coincide. We use PGA, and yet, the we still do better than the Euclidean method in terms of reconstruction.

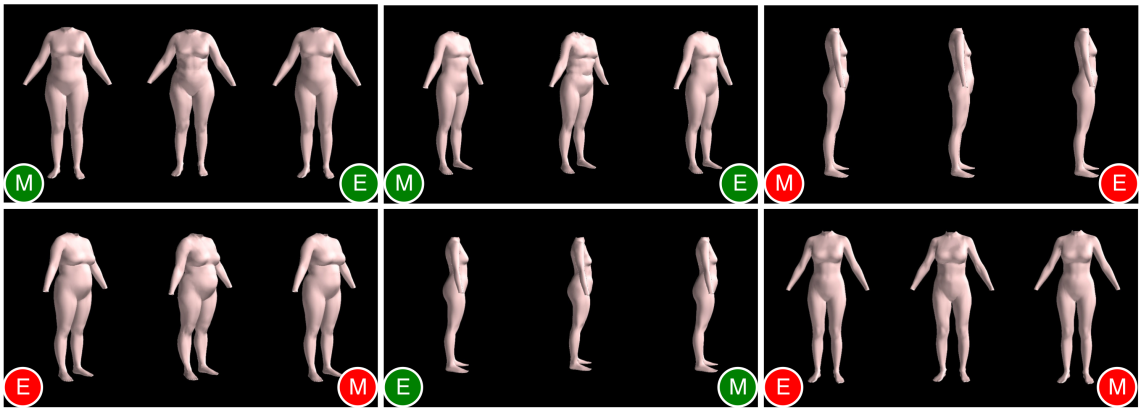


Figure 4.13: Perceptual task. Examples images shown in the shape perception experiment (workers did not see the "M" and "E" notations). The true mesh is always in the center. On either side, at random, is either the manifold representation (M) or the Euclidean representation (E) generated with 20 coefficients. Workers had to indicate which body looked more like the center. Green indicates examples that were "easy" (*i.e.* the manifold was selected as better more than 90% of the time) and red indicates "hard" examples where workers were roughly split in their decisions.

4.5.4 Human Shape Perception

We also evaluate the perceived quality of the reconstructions using a two-alternative forced choice perceptual experiment. This evaluates how well each representation captures features of body shape that are important to human perception. Mechanical Turk workers were shown images like those in Fig. 4.13 drawn from a set of 300. The center shape always showed the original mesh (head and hands were removed to focus people on body shape). On either side, in the same pose, we showed the PCA and Lie-algebraic PGA reconstructions. The left/right location was randomly varied and the reconstructed bodies were shown in a random order in one of 3 viewing directions: profile, frontal, and oblique. Each comparison was presented 10 times and each worker could try as many of the 300 examples as they wished. Of the 3000 answers, 7 were disqualified for technical reasons. M-valued reconstruction was preferred in 1670 out of 2993 answers (55.8%). For each of the 300 test images, the portion (usually out of 10) of the workers who preferred the M-valued reconstruction to the Euclidean was computed. A tie was reached in 20 % of the cases (59/300) while in 80%, a majority was achieved. When a majority was achieved, the manifold approach was selected as bet-

ter significantly more often: $\Pr(M \text{ won}|\text{majority was achieved}) \approx 55/80 = 0.69$ while $\Pr(\text{Euclidean won}|\text{ majority was achieved}) \approx 25/80 = 0.31$.

4.5.5 Predicting Biometric Measurements

An important quantitative measure of body mesh quality is the accuracy with which body shape can be used to compute anthropometric measurements. We compute a simple linear regression from subspace coefficients to body measurements as in [149]. We compare three subspaces: PCA on the $\mathcal{Q}^{\text{Euclidean}}$'s, PCA on the $\mathcal{Q}$'s, and Lie-algebraic PGA on the M-valued deformations.

The dataset was split into two halves for training and test sets. Summary results are shown in Fig. 4.14, while individual results are shown in Fig. 4.15. There is a clear advantage to the manifold approach, especially for a small number of coefficients: with this representation, the same amount of accuracy can be predicted from less shape coefficients.

4.6 Conclusion

When we want to infer a 3D shape from either partial and noisy 3D measurements (*e.g.*, laser scans in Anguelov *et al.* [6] or range data as in Weiss *et al.* [149]) or from image measurements (*e.g.*, Bălan *et al.* [16] or Guan *et al.* [59]), we often resort to doing *analysis-by-synthesis* using a statistical model. This necessitates a choice of a representation that *supports synthesis* of new shapes, as well as composition of different sources of deformations. With the availability of devices like Microsoft's Kinect, there is increasing interest in modeling the 3D shape of non-rigid and articulated objects – particularly, human body shape.

To model object shape variation we need an appropriate consistent representation that leads to effective statistical modeling. We propose a new Lie group for representing shape

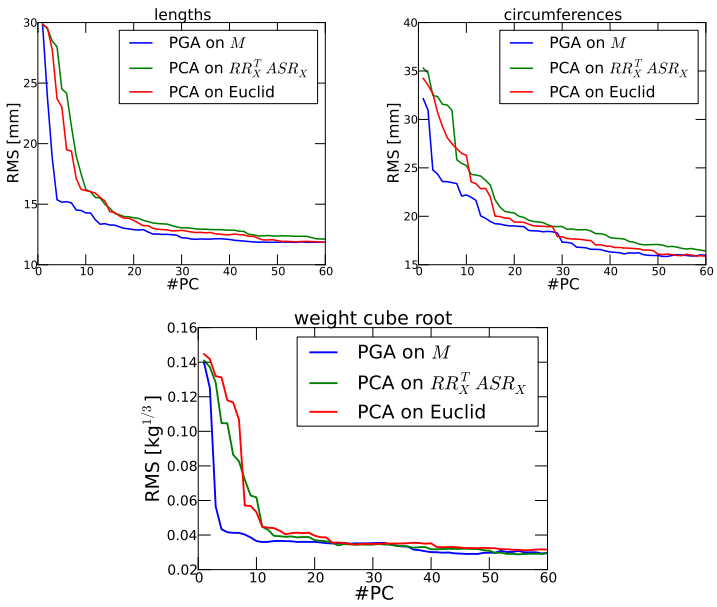


Figure 4.14: Linear prediction of body measurements from shape coefficients (summary). RMS error as a function of the number of coefficients. Left: Average results for length measurements ('spine to elbow', 'shoulder breadth', 'stature', 'knee height', 'spine to shoulder', 'arm'). Center: Average results for circumference measurements ('chest', 'thigh', 'ankle', 'under bust'). Right: Cube root of weight.

as a deformation from a template mesh. The approach has many nice properties. Unlike previous Euclidean methods, representing triangle deformations with our nonlinear manifold gives a heuristic-free exact solution for the required 6 DoF.

The distance between shapes is properly defined as a geodesic distance on the manifold of mesh deformations. Statistics of shape variation are represented using Principal Geodesic Analysis. These benefits come with little additional computational overhead since the main equations are efficiently computed in closed-form. In addition to theoretical benefits, we have shown that the Lie representation of shape deformation consistently outperforms the Euclidean approach.

In both existing representations and ours, the distances and statistical models depend on the arbitrary choice of the template $\mathcal{T}$. This is unsatisfying as the distance between, say, the shapes of two people should be independent of any third person. It is easy to redefine our distance, making it right-invariant. This would eliminate the aforementioned

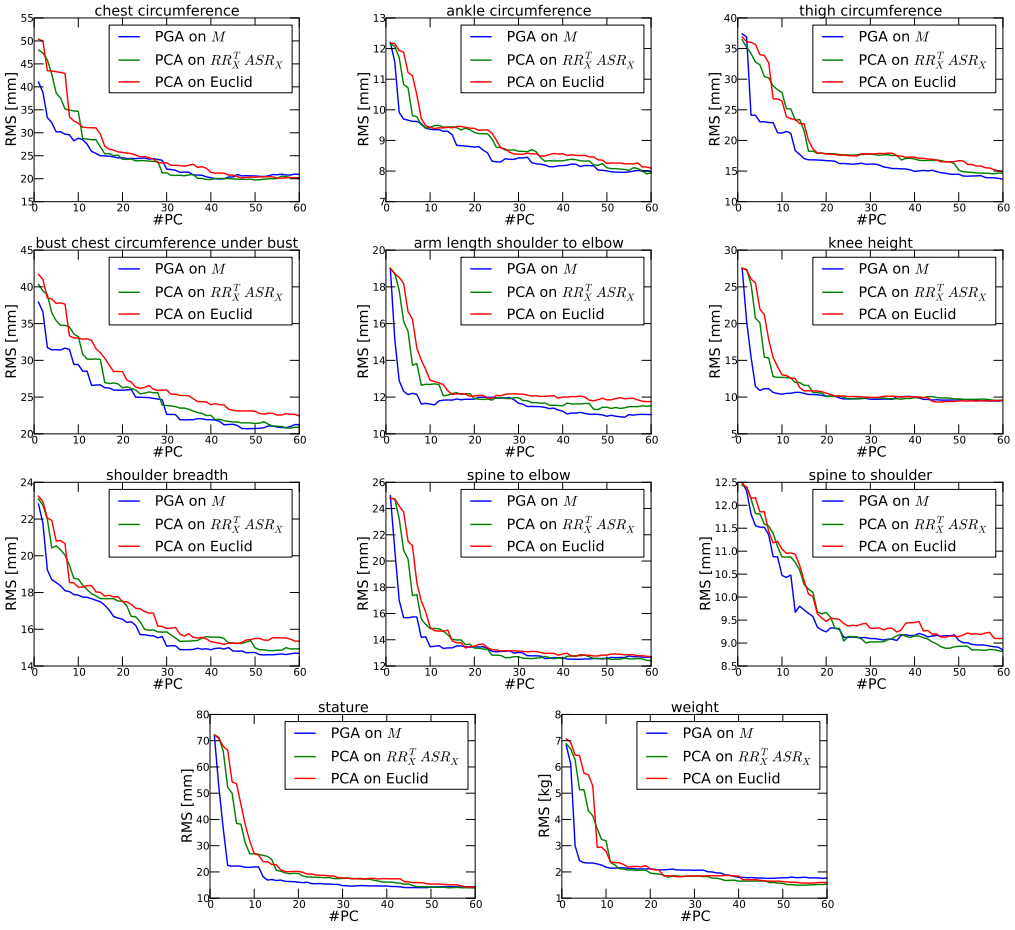


Figure 4.15: Linear prediction of body measurements from shape coefficients; RMS error as a function of the number of coefficients.

dependencies on $\mathcal{T}$. This, however, would come at the cost of losing left-invariance. As there is a tradeoff, the decision should be determined on an application-specific basis.

In our representation, we treat SO(3) as a matrix Lie group. This is consistent with the way we represent the scaling and planar deformations, and enables us to perform statistical analysis in a similar setting for all these three components. There are, of course, other representations for SO(3) (*e.g.*, quaternions, which, among other things, provide the basis for the SLERP method for interpolating rotations [132]), which might be preferred in some applications. As long as one can measure distances, interpolate, and do statistics on these alternate manifolds of rotations, our (R, A, S) decomposition is still applicable by a simple drop-and-replacement of representation of the SO(3) component.

Of note, statistical models based on representations of triangle deformations – our representation included – assume meshes have been aligned during preprocessing. The alignment problem – also known as the correspondence problem – is a fundamental problem in shape analysis that has drawn the interest of many researchers (*e.g.*, see [13, 143]). Typically, mesh alignment is done pairwise, as this is simpler than to solve the harder problem of *corpus mesh alignment*; *i.e.*, aligning multiple meshes simultaneously. A successful corpus alignment of the training mesh data improves the resulting statistical model while conversely, a good statistical model helps in mesh alignment of new meshes. In a recent work by Hirshberg *et al.* [72], the authors show that these two problems can be written as a single cost function. Thus, an alternate optimization between model learning and corpus alignment ensures both type of errors are reduced, and the eventual statistical model is superior to the initial one. Our representation can be naturally used in the setting suggested by Hirshberg *et al.*, improving the statistical model even further.

While focusing on human bodies, we emphasize the generality of the approach and envision Lie Shapes providing an improved foundation for shape representation, analysis, and synthesis.

Chapter 5

Riemannian Covariance Transport

We address the problem of learning statistical models on a known nonlinear Riemannian manifold, denoted by M, where the data of interest may be scarce but where we can leverage statistics learned on some other part of the manifold. More concretely, given two sets of M-valued data, one large (denoted by $\mathcal{D}_{\mathrm{L}}$) and one small (denoted by $\mathcal{D}_{\mathrm{S}}$), we show how the sample covariance of the larger dataset can be used to improve the estimation of the covariance of the smaller one. This echoes, and in fact generalizes, ideas from the literature of transfer learning.

Unfortunately, existing transfer learning methods for leveraging knowledge from one class to another are limited to $\mathbb{R}^n$-valued data. Here we develop a new framework for *covariance transport* of manifold-valued data that takes statistics learned on one part of the manifold to another part using any parallel transport that respects the Riemannian metric of the manifold[1].

We prove that covariance matrix learned in a part of the manifold where data are plentiful can be transported in a principled way to a part of the manifold where data are scarce. This enables us to augment or regularize the covariance of the few using the

[1]By this we mean any *metric* parallel transport – a notion to be explained later on.

covariance of the many.

Our method is equally applicable to both full covariance matrices and low-dimensional Riemannian PGA models (see Section 2.5.3) that are often used for dimensionality reduction. In the latter case, we stress that we use Riemannian PGA (as opposed to Lie-algebraic PGA) even if M happens to be a Lie group. In other words, our approach is purely Riemannian. In addition to making the method more broadly applicable (i.e., not restricted to Lie groups) as well as more tied to distances, this simplifies the use of a metric parallel transport.

We here show the approach applies well to 3D human body shape deformations. Note, however, that in [42] we show that the approach is also successful for several computer vision problem involving image descriptors for face modeling.

To summarize, our main contribution here is a framework for estimating covariance matrices of small-sample classes on nonlinear Riemannian manifolds. The framework, based on leveraging data from related classes, uses a principled way that takes the geometry of the manifold into account, and thus overcomes a limitation of conventional techniques, the latter being limited to Euclidean spaces. We also prove that covariance transport preserves statistical information. Our method is applicable to all (geodesically-complete) Riemannian manifolds, including those that are not Lie groups, whenever a metric parallel transport is analytically or numerically computable. This class includes, but is not limited to, spherical data, several popular image features, and 3D human body shape deformations. We here use the latter to illustrate the effectiveness of the approach. All computational details are provided.

5.1 Previous Work

Statistics on manifolds has a long history, starting with Ronald A. Fisher's work on spherical data [33]. Recently, various authors have generalized statistical tools from Euclidean spaces to nonlinear manifolds. A full literature survey of statistics on manifolds is beyond our scope; however, for a partial list – biased toward computer vision, medical imaging, and shape analysis – of recent works in this field, see [10, 25, 34–39, 49, 56, 66, 67, 70, 73, 99, 102, 104, 113–115, 135, 137, 139–141, 146, 150].

We go beyond previous work to provide the first generalization, from a Euclidean setting to a Riemannian one, of a transfer learning approach that addresses data scarcity by leveraging data from another class.

Using data from one class to improve parameter estimation for a model of another related class is a type of transfer learning and is sometimes called *domain adaptation* [26]. Given our two data sets, $\mathcal{D}_{\mathrm{S}}$ and $\mathcal{D}_{\mathrm{L}}$, the most straightforward idea is to learn a model from $\mathcal{D}_{\mathrm{L}}$ and use it as a prior for learning the model of $\mathcal{D}_{\mathrm{S}}$ [92]. A related approach learns models of the two datasets independently and then combines them using *shrinkage estimation* [129]. Daumé III and Marcu [26] propose to model both $\mathcal{D}_{\mathrm{S}}$ and $\mathcal{D}_{\mathrm{L}}$ as two-component mixture distributions, where one component is shared by the two distributions, such that both datasets contribute to the estimation of the shared component. While these ideas have been successful in Euclidean spaces, they are currently not applicable to data on nonlinear manifolds.

Manifold alignment [60] is a branch of machine learning where transfer learning is done through manifold learning techniques and is based on estimating a latent low-dimensional manifold shared by two $\mathbb{R}^D$-valued (or an $\mathbb{R}^{D_1}$- and an $\mathbb{R}^{D_2}$-valued) datasets. Despite the terms "manifold" and "transfer learning", our approach tackles a very different problem. Rather than $\mathbb{R}^D$, the space here is a known manifold, perhaps high dimensional, and we show how to overcome the known nonlinearity to achieve transfer learning of *manifold-*

valued data between two classes that live on two different parts of the manifold. See also our discussion in Section 2.5.1 regarding the differences between manifold learning and manifold-valued statistics.

Riemannian PGA [38] is a popular technique for dimensionality reduction for manifold-valued data. While dimensionality reduction is not our focus, we prove mathematically that our approach is applicable to PGA models just as it is to full covariances. This is also supported by our experiments.

As we will see, our method – for transfer learning of manifold-valued data – exploits a notion called *parallel transport*. This concept, providing a principled way to move vectors between different tangent spaces, will be defined in Section 5.4. We are not the first to employ parallel transport in statistical problems in computer vision. We now proceed to mention several examples of such works, but please note that this is a partial list[2]. Recently, parallel transport of *data* has been employed in image registration [96, 97]. Our work differs in both the objects being transported (in our case: covariance matrices) and the application (in our case: transfer learning between classes). Hauberg *et al*. [66] have used parallel transport of the uncertainty covariance in a Kalman Filter for tracking applications. That work did not address transfer learning and our application of covariance transport is different and more general. Moreover, while the authors of [66] showed that *some* covariance can be generated from the parallel transport of another, we prove a stronger result: *the new covariance is not just an arbitrary covariance but has a concrete statistical meaning tied to the transport of the data that generated the old one.*

Importantly, we stress that while used before, parallel transport was never used for transfer learning. The only recent exception is Wei *et al*. [147]. Before explaining the main difference between that work and ours, we point out a well-known fact from differential geometry: parallel transport has many types.

[2]In particular, applications of parallel transport in non-statistical problems in computer vision are beyond our scope.

We observe that for transporting correlations or variance, however, it is crucial that the parallel transport *preserves inner-products*; see Section 5.5. Wei *et al*. [147] suggest sharing information across a Lie group of image deformations using the parallel transport implied by a spatial image transformation. In particular, they transport linear bases, each one of which spans a subspace of the Lie algebra and is also associated with a different mean deformation (and thus each basis is associated with a different region of the manifold). Unfortunately, their choice of parallel transport has the unwanted side-effects that neither the inner-product between two vectors nor the length of a vector is preserved. Thus, while Wei *et al*. are able to transport a basis across the manifold, the basis is distorted, and, more importantly, they are unable to transport a covariance matrix or preserve variance, hence statistical information is lost. Our approach does not suffer from this shortcoming. Also, while their approach is Lie-algebraic, ours is purely Riemannian and hence, besides being applicable to a much broader class of manifolds, utilizes true geodesic distances rather than their approximations; see Section 2.6.

5.2 The Euclidean Setting

Before moving to the manifold setting, let us briefly describe the simpler scenario where the space is Euclidean. Consider two sets of $\mathbb{R}^D$-valued data, denoted by $\mathcal{D}_\mathrm{L}$ and $\mathcal{D}_\mathrm{S}$. The first dataset consists of N_L points, while the second consists of N_S points. It is assumed that $N_\mathrm{L} > N_\mathrm{S}$. As shorthands, and as we will do throughout this chapter, we use L to denote 'large' and S to denote 'small'. Suppose $\mathcal{D}_\mathrm{L}$ and $\mathcal{D}_\mathrm{S}$ represent two sets of *i.i.d.* samples from two D-dimensional Gaussians:

$$\mathcal{D}_\mathrm{L} = \{x_1, \ldots, x_{N_\mathrm{L}}\} \subset \mathbb{R}^D \,, \qquad x_i \overset{i.i.d.}{\sim} \mathcal{N}(\mu_\mathrm{L}, \Sigma_\mathrm{L}) \,; \tag{5.1}$$

$$\mathcal{D}_\mathrm{S} = \{y_1, \ldots, y_{N_\mathrm{S}}\} \subset \mathbb{R}^D \,, \qquad y_i \overset{i.i.d.}{\sim} \mathcal{N}(\mu_\mathrm{S}, \Sigma_\mathrm{S}) \,, \tag{5.2}$$

where μ_{L} and μ_{S} are points in $\mathbb{R}^D$, while Σ_{L} and Σ_{S} are $D \times D$ covariance matrices. Note that the two covariance matrices can also be seen as bilinear forms on $\mathbb{R}^D$:

$$\Sigma_{\mathrm{L}} : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R} \ : (a, b) \mapsto a^T \Sigma_{\mathrm{L}}^{-1} b \ ; \tag{5.3}$$

$$\Sigma_{\mathrm{S}} : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R} \ : (a, b) \mapsto a^T \Sigma_{\mathrm{S}}^{-1} b \ . \tag{5.4}$$

In particular, note that the domain of both these bilinear forms is the same: $\mathbb{R}^D \times \mathbb{R}^D$. We will return to this point in the next section.

Let $\widehat{\mu_{\mathrm{L}}}$ and $\widehat{\mu_{\mathrm{S}}}$ denote the corresponding sample means:

$$\widehat{\mu_{\mathrm{L}}} \stackrel{\text{def}}{=} \frac{1}{N_{\mathrm{L}}} \sum_{i=1}^{N_{\mathrm{L}}} x_i \ ; \tag{5.5}$$

$$\widehat{\mu_{\mathrm{S}}} \stackrel{\text{def}}{=} \frac{1}{N_{\mathrm{S}}} \sum_{i=1}^{N_{\mathrm{S}}} y_i \ . \tag{5.6}$$

Likewise, let $\widehat{\Sigma_{\mathrm{L}}}$ and $\widehat{\Sigma_{\mathrm{S}}}$ denote the corresponding sample covariances:

$$\widehat{\Sigma_{\mathrm{L}}} \stackrel{\text{def}}{=} \frac{1}{N_{\mathrm{L}} - 1} \sum_{i=1}^{N_{\mathrm{L}}} (x_i - \widehat{\mu_{\mathrm{L}}})(x_i - \widehat{\mu_{\mathrm{L}}})^T \ ; \tag{5.7}$$

$$\widehat{\Sigma_{\mathrm{S}}} \stackrel{\text{def}}{=} \frac{1}{N_{\mathrm{S}} - 1} \sum_{i=1}^{N_{\mathrm{S}}} (y_i - \widehat{\mu_{\mathrm{S}}})(y_i - \widehat{\mu_{\mathrm{S}}})^T \ . \tag{5.8}$$

If N_{S} is too small, then $\widehat{\Sigma_{\mathrm{S}}}$ is usually a poor estimate of Σ_{S}. Suppose, however, that we have reasons to believe that Σ_{L} and Σ_{S} are not unrelated[3]. In which case, one may try to improve $\widehat{\Sigma_{\mathrm{S}}}$ by also using $\widehat{\Sigma_{\mathrm{L}}}$. There are plenty of methods to combine two (or more) covariance matrices in order to produce a new one[4]. The particular method of choice is not so important for our discussion. What is important, however, is to understand that there is a particular property of $\mathbb{R}^D$ that is being used here transparently. The property we allude to is that once both datasets are centered with respect to their (sample)

[3]For example, we may suspect that their geodesic distance with respect to the manifold of D-by-D SPD matrices is small, or we may believe that their inverse matrices can be well modeled as samples from the same (possibly unknown) Wishart distribution.

[4]For example, we can use a convex combination of $\widehat{\Sigma_{\mathrm{L}}}$ and $\widehat{\Sigma_{\mathrm{S}}}$.

means, both covariance matrices, $\widehat{\Sigma_{\mathrm{L}}}$ and $\widehat{\Sigma_{\mathrm{S}}}$, "live" on the same space; *i.e.*, both are covariance matrices of zero-mean D-dimensional Gaussians. In other words, one implicitly uses ordinary translation as the very first part of the process. Geometrically speaking, a $\widehat{\mu_{\mathrm{L}}}$-centered Gaussian that is associated with $\widehat{\Sigma_{\mathrm{L}}}$ is translated from $\widehat{\mu_{\mathrm{L}}}$ to $\widehat{\mu_{\mathrm{S}}}$. Another way to think of it is that had we added the constant (but random) offset $\widehat{\mu_{\mathrm{S}}} - \widehat{\mu_{\mathrm{L}}}$ to each one of the elements of $\mathcal{D}_{\mathrm{L}}$, then this would have not changed the value of $\widehat{\Sigma_{\mathrm{L}}}$. Of course, in practice there is no need to do this translation since it has no effect on the resulting estimates; however, this line of thinking prepares us for the more complicated scenario that is coming up next.

5.3 The Manifold Setting

In the setup we are interested in, however, the space is not Euclidean. To be more explicit, let the data lie on a geodesically-complete Riemannian manifold (see Definition 2.4.10, page 64). As usual, we use M to denote the manifold. Our goal is to move a statistical model learned in one region of M to another. While in $\mathbb{R}^D$ this can be achieved by translation, such an approach fails for nonlinear manifolds as a translation of a tangent vector rarely produces another tangent vector.

On manifolds, statistical models are often expressed in tangent spaces. For example, recall how *a sample covariance is expressed in a tangent space* (see Eqn. (2.83), page 69) or how PGA is defined (see Section 2.5.3.2, page 69). Consequently, in order for us to be able to apply a similar idea to the one we have just seen in Section 5.2, we need to be able to move a covariance matrix between between tangent spaces.

We again consider two datasets, $\mathcal{D}_{\mathrm{L}}$ and $\mathcal{D}_{\mathrm{S}}$, but henceforth we will assume these

datasets consist of points in M:

$$\mathcal{D}_{\mathrm{L}} = \{p_1, \ldots, p_{N_{\mathrm{L}}}\} \subset M \; ; \tag{5.9}$$

$$\mathcal{D}_{\mathrm{S}} = \{q_1, \ldots, q_{N_{\mathrm{S}}}\} \subset M \; . \tag{5.10}$$

Let μ_{L} and μ_{S} represent two, possibly unknown, points in M. It is assumed that the two datasets satisfy the following statistical rules:

$$\left\{\mathrm{Log}_{\mu_{\mathrm{L}}}(p_1), \ldots, \mathrm{Log}_{\mu_{\mathrm{L}}}(p_{N_{\mathrm{L}}})\right\} \subset T_{\mu_{\mathrm{L}}}M \quad , \quad \mathrm{Log}_{\mu_{\mathrm{L}}}(p_i) \overset{i.i.d.}{\sim} \mathcal{N}(0, \Sigma_{\mathrm{L}}) \; ; \tag{5.11}$$

$$\left\{\mathrm{Log}_{\mu_{\mathrm{S}}}(q_1), \ldots, \mathrm{Log}_{\mu_{\mathrm{S}}}(q_{N_{\mathrm{S}}})\right\} \subset T_{\mu_{\mathrm{S}}}M \quad , \quad \mathrm{Log}_{\mu_{\mathrm{S}}}(q_i) \overset{i.i.d.}{\sim} \mathcal{N}(0, \Sigma_{\mathrm{S}}) \; . \tag{5.12}$$

In other words, when $\mathcal{D}_{\mathrm{L}}$ is expressed (as tangent vectors) in the tangent space (to M) at μ_{L}, it can be seen as a set of *i.i.d.* samples from a zero-mean Gaussian with covariance Σ_{L}. Likewise, when $\mathcal{D}_{\mathrm{S}}$ is expressed in the tangent space at μ_{S} it can be seen as a set of *i.i.d.* samples from a zero-mean Gaussian with covariance Σ_{S}. This generalizes the Euclidean case.

The two covariance matrices can also be seen as bilinear forms:

$$\Sigma_{\mathrm{L}} : T_{\mu_{\mathrm{L}}}M \times T_{\mu_{\mathrm{L}}}M \to \mathbb{R} \; : (a, b) \mapsto a^T \Sigma_{\mathrm{L}}^{-1} b \; ; \tag{5.13}$$

$$\Sigma_{\mathrm{S}} : T_{\mu_{\mathrm{S}}}M \times T_{\mu_{\mathrm{S}}}M \to \mathbb{R} \; : (a, b) \mapsto a^T \Sigma_{\mathrm{S}}^{-1} b \; . \tag{5.14}$$

Note that unlike in the Euclidean case (Eqns. (5.3) and (5.4)), here the domains are different: Σ_{L} defines a bilinear form on $T_{\mu_{\mathrm{L}}}M$, while Σ_{S} defines a bilinear form on $T_{\mu_{\mathrm{S}}}M$. This means that as mathematical objects, the two covariance matrices do not "live" on the same space. Thus, standard Euclidean methods for combining covariance matrices to produce a new one do not apply here. Earlier, in Section 5.2, we noted that these methods implicitly rely on ordinary Euclidean translation. This is exactly what breaks down here. If x is in $T_{\mu_{\mathrm{L}}}M$, then $x + (\mu_{\mathrm{S}} - \mu_{\mathrm{L}})$ is usually not in $T_{\mu_{\mathrm{S}}}M$. In fact, if there is no ambient space present, this linear combination is not even defined. Consequently, a covariance cannot be simply translated between tangent spaces as if M were linear; see Fig. 5.1.

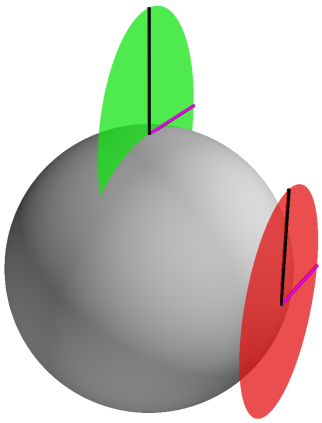


Figure 5.1: On a manifold, the ordinary translation of a covariance matrix (expressed in tangent space) usually fails. Here we see that when the red covariance is translated by the Euclidean difference of two points of tangency, the result (green) is no longer in any tangent space to the manifold.

To remedy this problem, we need to replace ordinary translation with a method that moves vectors from one tangent space to another. However, this alone is not enough: not every map between tangent spaces suffices for our needs. We want a method that can be used to move not only vectors but also a covariance matrix. Moreover, the new covariance must preserve the statistical information of the old one, for otherwise the new covariance will be of limited use to us.

Fortunately, these goals can be accomplished by using a *(metric) parallel transport*.

5.4 Parallel Transport

5.4.1 Parallel Transport on the Sphere

As a warm-up, we start our discussion with a relatively easy example: the unit sphere. Let $M = S^{n-1}$ where

$$S^{n-1} \stackrel{\text{def}}{=} \{p : p \in \mathbb{R}^n,\ \|p\|_{\ell^2} = 1\}\ , \tag{5.15}$$

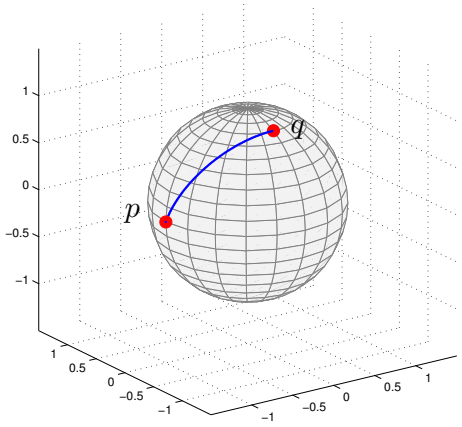


Figure 5.2: Two points on the sphere connected by a geodesic.

and let the Riemannian metric on M be defined as in Example 2.2.3, page 30. Let p and q be points in M, and let x_0 be a tangent vector in T_pM. As a first step, we would like to map x_0 from T_pM to T_qM. Of course, there are many possible way to do it, but here will focus our attention on a particular one. Set $x = \mathrm{Log}_p(q)$, $u = x/\|x\|$ and $m = \|x\|$. Note that the formula for computing $\mathrm{Log}_p(q)$ is given by Eqn. (2.19), page 31. Let $c(t) : [0,1] \to M$ be the geodesic curve between p and q (see Eqn. (2.20), page 31); see Fig. 5.2. Note that $c(0) = p$ and so $T_{c(0)}M = T_pM$. Likewise, $c(1) = q$ and so $T_{c(1)}M = T_qM$. Now consider the following collection of maps,

$$\left\{\Gamma_{c(0)\to c(t)}(\cdot) : T_{c(0)}M \to T_{c(t)}M\right\}_{t\in[0,1]} \tag{5.16}$$

where, for a given t,

$$\Gamma_{c(0)\to c(t)}(x_0) = \left(-p\sin(mt)u^T + u\cos(mt)u^T + (I_{n\times n} - uu^T)\right)x_0\ . \tag{5.17}$$

In particular, for $t = 1$, we get that $\Gamma_{c(0)\to c(1)}(x_0)$ is indeed in T_qM. The results of applying these maps to x_0, for several different values of t, can be shown in Fig. 5.3. The maps defined by Eqn. (5.17) are known as the (Levi-Civita) parallel transport of a tangent vector along a geodesic curve on the sphere (with the particular Riemannian metric mentioned above) [1]. Several remarks are in order, some of which may be better understood later on when we discuss the more general case:

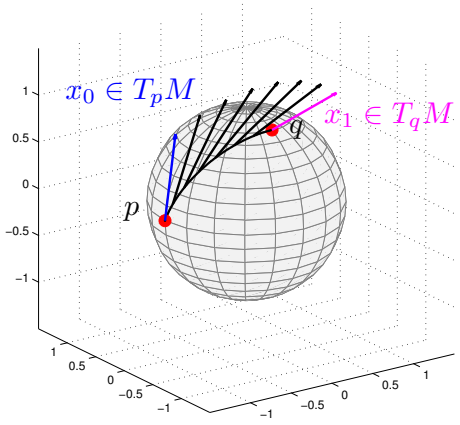


Figure 5.3: Parallel transport of x_0 along the geodesic between p and q.

1. $\Gamma_{c(0)\to c(0)}(x_0) = x_0$.

2. Let u, s and t be in $[0,1]$. If $\Gamma_{c(u)\to c(t)}$ and $\Gamma_{c(s)\to c(u)}$ are defined in a similar way to $\Gamma_{c(0)\to c(t)}$, then $\Gamma_{c(u)\to c(t)} \circ \Gamma_{c(s)\to c(u)} = \Gamma_{c(s)\to c(t)}$.

3. If x_0 is fixed and t varies, then $\Gamma_{c(0)\to c(t)}(x_0) : [0,1] \to \mathbb{R}^n$ is a smooth function of t.

4. If x and y are in T_pM, then, for every t, their inner-product is preserved:

$$\langle x, y \rangle_{c(0)} = \left\langle \Gamma_{c(0)\to c(t)}(x), \Gamma_{c(0)\to c(t)}(y) \right\rangle_{c(t)} . \tag{5.18}$$

   Consequently, norms of tangent vectors and angles between tangent vectors are preserved; see Fig. 5.4.

5. $\Gamma_{c(0)\to c(t)} : T_{c(0)}M \to T_{c(t)}M$ is a bijective linear map.

5.4.2 The General Case

We now proceed to the more general case. Let M be a geodesically-complete Riemannian manifold. To see how a covariance can be moved across M, consider first tangent vectors. While simple translation will not do, we can transport vectors from one tangent space to another using *parallel transport*, to be defined as follows[5].

[5] Another way to define it is trough the notion of a *connection* - a term we avoid elaborating on. See [27].

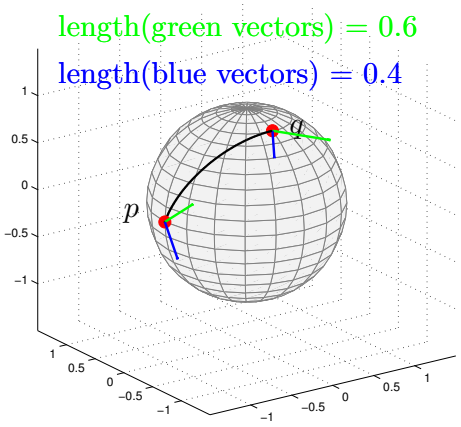


Figure 5.4: The Levi-Civita parallel transport preserves angles and norms.

Definition 5.4.1 (Parallel transport)**.** Suppose that for every smooth curve $c : [0,1] \to M$ we have a collection, depending on c, of maps, $\left\{\Gamma_{c(s)\to c(t)} : T_{c(s)}M \to T_{c(t)}M | s,t \in [0,1]\right\}_c$ such that

1. $\Gamma_{c(s)\to c(s)}$ is the identity map on $T_{c(s)}M$.
2. $\Gamma_{c(u)\to c(t)} \circ \Gamma_{c(s)\to c(u)} = \Gamma_{c(s)\to c(t)}$.
3. The dependency of $\Gamma_{c(s)\to c(t)}$ on s and t is smooth.

In which case, if $x \in T_{c(s)}M$, we call $\Gamma_{c(s)\to c(t)}(x) \in T_{c(t)}M$ the *parallel transport* of x.

When c, s, and t are understood from the context, we use Γ instead of $\Gamma_{c(s)\to c(t)}$ for brevity.

5.4.3 Metric Parallel Transport

As there are many ways to satisfy the conditions from Definition 5.4.1, *there exist many different types of parallel transport.* In our context, some types are more useful than others.

Definition 5.4.2 (Metric parallel transport)**.** A *metric parallel transport* is one that pre-

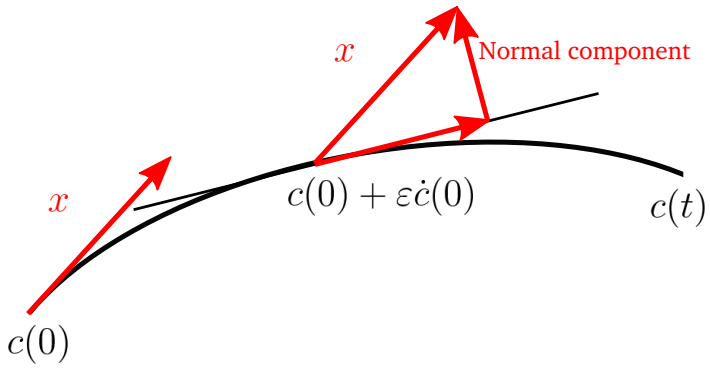


Figure 5.5: Levi-Civita parallel transport: an infinitesimal construction. This figure has been adapted from a similar figure in [28].

serves inner-products; *i.e.*,

$$\langle x, y \rangle_p = \langle \Gamma(x), \Gamma(y) \rangle_q \tag{5.19}$$

for all points p, q on c, and every $x, y \in T_pM$.

Consequently, orthogonality and distance between vectors are preserved by a metric parallel transport. As in the more general case, there exist many types of metric parallel transport.

5.4.4 The Levi-Civita Parallel Transport

Our framework is general and be can be used with any metric parallel transport. For concreteness, simplicity, and ease of computation, in our experiments we use the one associated with the Levi-Civita (LC) connection. While often defined via a connection it may also be constructed as follows [28]: a tangent vector can be transported along a curve by infinitesimally translating the vector and removing the normal component of the translated vector. See Fig. 5.5 for an illustration.

This construction leads to an ordinary differential equation (ODE) whose solution coincides with the LC parallel transport. For some manifolds, closed-form solutions for this ODE can be derived. This includes the orthogonal group, rotation group [28] and the

unit sphere in $\mathbb{R}^n$ [1]. In our experiments, we use the manifold from Chapter 4; that is, a manifold whose building blocks are SO(3), G_A, and G_S. For G_S and SO(3), closed-form solutions are available. For manifolds for which no closed-form formula is available, such as G_A in our experiments, we use a technique known as Schild's ladder.

5.4.4.1 Closed-Form Solutions for G_S and SO(3)

G_S: For $G_S \cong \mathbb{R}^+$ the LC parallel transport of x is simply x. Here is a simple proof.

Proof. Let $M = \mathbb{R}^+$. First, observe that $\langle x, y \rangle_p = xy$ for all $p \in M$. Let $\Gamma(x)$ denote the parallel transport of x along the geodesic from p to q. Since the LC parallel transport preserves inner-products, $xy = \Gamma(x)\Gamma(y)$. In particular, this is true for $x = y = 1$. Thus, $\Gamma(1) = 1$ (it cannot be the case that $\Gamma(1) = -1$ since if we take $p = q$ this would mean a huge change to a vector - although we did not even move from the starting point). Now, take $y = 1$ and $x \neq 1$. This implies $1x = 1\Gamma(x)$ and so $\Gamma(x) = x$ for all x. □

SO(3): Let $R_0, R_1 \in \text{SO}(3)$. The LC parallel transport of $x_0 \in T_{R_0}\text{SO}(3)$ to $T_{R_1}\text{SO}(3)$ along the geodesic between R_0 and R_1 is given by

$$R_0 \exp(A/2) B_0 \exp(A/2) \; . \tag{5.20}$$

where $B_0 \stackrel{\text{def}}{=} R_0^T x_0 \in \text{so}(3)$ and $A \stackrel{\text{def}}{=} \log(R_0^T R1) \in \text{so}(3)$. See [28] for a proof. In fact, this result holds for SO(n) in general, and not just for SO(3) [28].

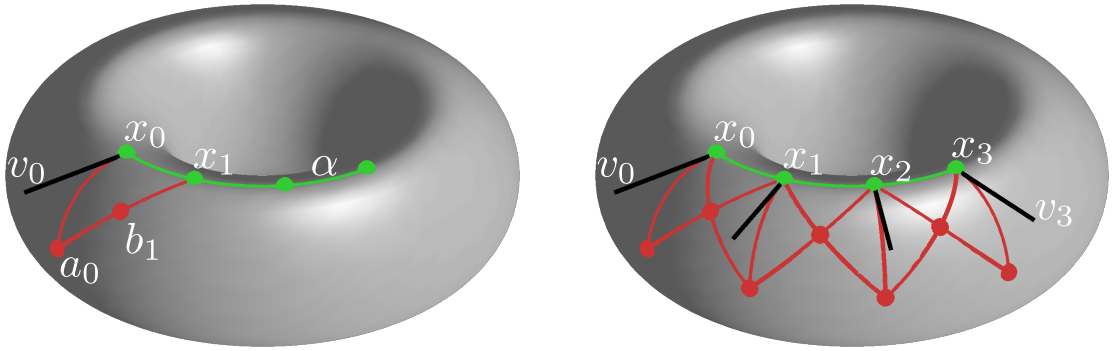


Figure 5.6: An illustration of *Schild's Ladder* for approximating the parallel transport (for $K = 3$). See text for details.

5.4.4.2 Schild's ladder

When there is no closed-form solution, we use *Schild's ladder* [106], a strikingly simple numerical technique to compute an arbitrarily accurate approximation[6] of the LC parallel transport using only the Exp/Log maps[7]. We are not the first to use Schild's ladder in computer vision applications (although we are the first to use it in the context of transfer learning). For example, the technique has been recently used in modeling longitudinal medical data [96, 115] and in tracking [66].

Schild's ladder is a numerical scheme that enables computation of the LC parallel transport [106]. We wish to parallel transport a tangent vector v_0 from x_0 to x_K on M along the geodesic curve α that joins them. Schild's ladder places points along α and approximately parallel transports v_0 to these by forming generalized parallelograms[8] on M (see Fig. 5.6): Let $\{x_1, \ldots, x_{K-1}\}$ denote points along α. Start by computing $a_0 = \text{Exp}_{x_0}(v_0)$ and the midpoint b_1 of the geodesic segment joining x_1 and a_0. Follow the geodesic from x_0 through b_1 for twice its length to the point a_1. This scheme is repeated for all sampled points along the geodesic from x_0 to x_K. The final parallel transport of v_0

[6] This is not true for every parallel transport – but it does hold at least for parallel transport associated with a *symmetric* connection [82] such as LC, the only connection which is both metric and symmetric.

[7] Note that if Exp/Log maps are unavailable analytically, then Exp maps can be computed by integrating an initial value problem and geodesics/Log maps can be computed by solving a boundary value problem [111].

[8] This generalization is known as the Levi-Civita parallelogramoid.

is then given by $\text{Log}_{x_K}(a_K)$.

Remark 5.4.1. The infinitesimal construction of the LC parallel transport described above, while elegant and simple to understand, relies on the presence of an ambient space. When an ambient space is not present, the traditional connection-based construction should be used instead. We stress, however, that this nuance merely refers to how the LC parallel transport is constructed as a mathematical object: for actually computing the parallel transport, Schild's ladder is applicable regardless of which construction is used, and does not rely on the presence of an ambient space.

5.5 Covariance Transport

Recent work [96,115] has explored the implications of different choices of parallel transport of vectors for several computer vision applications. While the optimal choice depends on the application, note that in our new application, we are interested in transporting *covariance matrices*, not just vectors. For this purpose, it is imperative to restrict the choice to metric parallel transports. While this restriction is intuitive since the notion of a covariance is strongly tied to inner products and distances[9], our following *Covariance Transport Theorem* makes this precise; a metric parallel transport enables the transport of a covariance matrix through the transport of its eigenvectors while still preserving statistical information.

5.5.1 The Covariance Transport Theorem

Theorem 5.5.1 (Covariance Transport)**.** *Let p and q be points in a D-dimensional (geodesically-complete) Riemannian manifold M, and let $\{x_i\}_{i=1}^N \subset T_pM$ denote the data. Let VS^2V^T be the eigen-decomposition of $XX^T \in \mathbb{R}^{D\times D}$ where $X \overset{def}{=} [x_1, \ldots, x_N]$ and let VSU^T be the Singular Value Decomposition (SVD) of $X \in \mathbb{R}^{D\times N}$ with $\{S_{i,i}\}_{i=1}^D$, the diagonal entries of S, sorted in a non-increasing order: $S_{1,1} \geqslant S_{2,2} \geqslant \ldots \geqslant S_{D,D}$. Let $[v_1, \ldots, v_N]$ denote the*

[9] Consider the $\mathbb{R}^n$ definitions of variance, correlations, and angles.

columns of V and let $\tilde{x} \stackrel{def}{=} \Gamma(x) \in T_qM$ denote the transport of $x \in T_pM$ according to a metric parallel transport along a smooth curve $c : [0,1] \to M$. Let $k < D$. Then:

1. *$\tilde{V}SU^T$ is the SVD of $\tilde{X}$, where $\tilde{X} \stackrel{def}{=} [\tilde{x}_1, \ldots, \tilde{x}_N]$ is the transported data, and $\tilde{V} \stackrel{def}{=} [\tilde{v}_1, \ldots, \tilde{v}_N]$ is the transported left singular vectors. Similarly, $\tilde{V}S^2\tilde{V}^T$ is the eigen-decomposition of $\tilde{X}^T\tilde{X} \in \mathbb{R}^{D \times D}$.*

2. *The k-dimensional PCA model of $\{\widetilde{x}_i\}_{i=1}^N \subset T_qM$ is given by*

$$\textit{(eigenvectors)} \qquad [\tilde{v}_1, \ldots, \tilde{v}_k] \subset T_qM \text{ and} \tag{5.21}$$

$$\textit{(eigenvalues)} \qquad \left[\frac{S_{1,1}}{\sqrt{N-1}}, \frac{S_{2,2}}{\sqrt{N-1}}, \ldots, \frac{S_{k,k}}{\sqrt{N-1}}\right] \in \mathbb{R}^k \ . \tag{5.22}$$

   *In other words, this model is equivalent to computing a k-dimensional PCA model of $\{x_i\}_{i=1}^N$ (*i.e.*, in T_pM) given by*

$$\textit{(eigenvectors)} \qquad [v_1, \ldots, v_k] \subset T_pM \text{ and} \tag{5.23}$$

$$\textit{(eigenvalues)} \qquad \left[\frac{S_{1,1}}{\sqrt{N-1}}, \frac{S_{2,2}}{\sqrt{N-1}}, \ldots, \frac{S_{k,k}}{\sqrt{N-1}}\right] \in \mathbb{R}^k \ , \tag{5.24}$$

   and then transporting the eigenvectors while keeping the standard deviations unchanged.

Remark 5.5.1. Note that while X and $\tilde{X}$ have different left-singular vectors (V and $\tilde{V}$, respectively), they share the same singular values (S) and same right-singular vectors (U).

Before the proof, we need some preliminaries:

Claim 1. *Parallel transport between tangent spaces is surjective.*

Proof. By part *ii)* of Definition 5.4.1, $\Gamma_{c(t)\to c(s)} \circ \Gamma_{c(s)\to c(t)} = \Gamma_{c(s)\to c(s)}$. By part *i)* $\Gamma_{c(s)\to c(s)}$ is the identity map on $T_{c(s)}M$. Thus $\Gamma_{c(s)\to c(t)}$ is a bijection between $T_{c(s)}M$ and a subset of $T_{c(t)}M$. We need to show that this subset is in fact the whole of $T_{c(t)}M$. Similar reasoning shows that $\Gamma_{c(t)\to c(s)}$ is a bijection between $T_{c(t)}M$ and a subset of $T_{c(s)}M$. We now use a

proof by contradiction. Suppose there exists y in $T_{c(t)}M$ such that $y \notin \Gamma_{c(s)\to c(t)}(T_{c(s)}M)$. We know that $\Gamma_{c(t)\to c(s)}(y) \in T_{c(s)}M$ and thus $z \stackrel{\text{def}}{=} \Gamma_{c(s)\to c(t)}(\Gamma_{c(t)\to c(s)}(y))$ is in the image of $\Gamma_{c(s)\to c(t)}$. By assumption, $z \neq y$. However, this is in contradiction to the fact that $\Gamma_{c(t)\to c(s)} \circ \Gamma_{c(s)\to c(t)}$ is the identity map on $T_{c(t)}M$. □

The following theorem is well known from functional analysis.

Theorem (Mazur-Ulam)**.** *Every surjective isometry between two normed linear spaces is affine.*

The term *isometry* means a distance-preserving map.

Claim 2. *Every surjective inner-product-preserving map between two inner-product spaces is linear.*

Proof. An inner-product-preserving map between two inner-product spaces is an isometry. If the map is also surjective then by the Mazur-Ulam Theorem it is affine. Let $f : U \to V$ be a surjective inner-product-preserving map between two inner-product spaces U and V. Thus, f is affine: $f : x \mapsto Ax + b$ where A is linear map $A : U \to V$ and $b \in V$. We need to show that $b = 0_V$, the zero element of V. If $x, y \in U$ then,

$$\langle x, y\rangle_U = \langle Ax + b, Ay + b\rangle_V = \langle Ax, Ay\rangle_V + \langle Ax, b\rangle_V + \langle b, Ay\rangle_V + \|b\|_V^2 \ . \tag{5.25}$$

Taking $x = y = 0_U$, we get that

$$\begin{aligned} 0 &= \|0_U\|_U^2 \\ &= \|A0_U\|_V^2 + \langle A0_U, b\rangle_V + \langle b, A0_U\rangle_V + \|b\|_V^2 \\ &= \|0_V\|_V^2 + \langle 0_V, b\rangle_V + \langle b, 0_V\rangle_V + \|b\|_V^2 \\ &= 0 + 0 + 0 + \|b\|_V^2 \ , \end{aligned} \tag{5.26}$$

implying that $b = 0_V$. □

The following corollary follows immediately.

Corollary 5.5.2. *Every metric parallel transport is linear.*

We now prove the theorem.

Proof. Recall that VS^2V^T is the eigen-decomposition of $XX^T \in \mathbb{R}^{D\times D}$ and VSU^T is the SVD of $X \in \mathbb{R}^{D\times N}$. Let $A \overset{\text{def}}{=} V^TX = SU^T$, $A \in \mathbb{R}^{D\times N}$. Denote the columns of A by $A = [a_1, \ldots, a_N] = [Su_1^T, \ldots, Su_1^T]$ where $[u_1, \ldots, u_1] \overset{\text{def}}{=} U$. Thus, $X = VV^TX = VA \in (T_pM)^N \cong \mathbb{R}^{D\times N}$ and so $x_j = Va_j = \sum_{k=1}^D v_kA_{k,j} \in T_pM \cong \mathbb{R}^D$. Applying a metric parallel transport, we get (for $\tilde{x}_j \in T_qM \cong \mathbb{R}^D$),

$$\tilde{x}_j = \overbrace{\sum_{k=1}^{D} v_kA_{k,j}} \overset{\text{linearity}}{=} \sum_{k=1}^{D} \tilde{v}_kA_{k,j} = \tilde{V}a_j \ . \tag{5.27}$$

In other words, $\tilde{X} = \tilde{V}A = \tilde{V}SU^T$. Since V is orthogonal and inner products are preserved, we know that $\tilde{V}$ is orthogonal too. And since S is diagonal and U is orthogonal, it follows that $\tilde{V}SU$ is indeed the SVD of $\tilde{X}$; namely, the left singular vectors of $\tilde{X}$ are exactly the transported left singular vectors of X, while the singular values and right singular vectors are unchanged. As an aside remark, note that $A \overset{\text{def}}{=} V^TX = \tilde{V}^T\tilde{X}$ since inner products are preserved. That $\tilde{V}S^2\tilde{V}^T$ is the eigen-decomposition of $\tilde{X}\tilde{X}^T \in \mathbb{R}^{D\times D}$ follows from $\tilde{X}\tilde{X}^T = \tilde{V}SUU^TS\tilde{V}^T = \tilde{V}S^2\tilde{V}^T$, concluding the proof of part *i*).

Finally, by the usual connections between SVD and PCA, part *ii*) is a direct consequence of part *i*). □

5.5.2 Implications of the Theorem

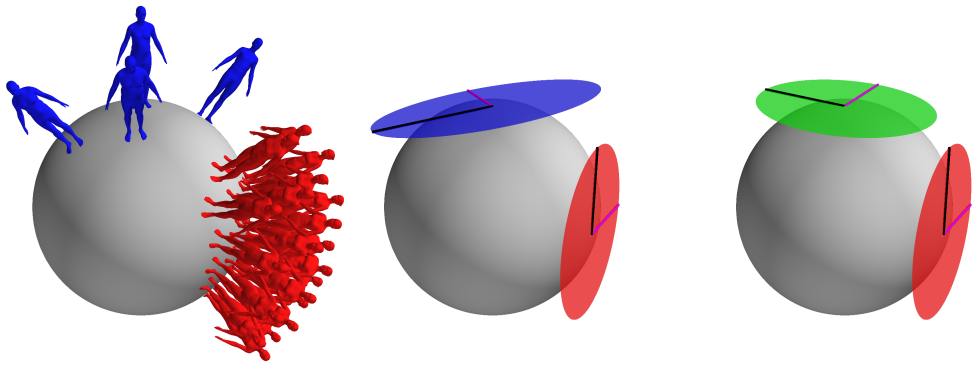
In Hauberg *et al.* [66], the authors worked with LC parallel transport and show that from parallel-transported eigenvectors of a covariance matrix in T_pM, *some* covariance matrix can be built in T_qM. Theorem 5.5.1 is much stronger. First, it shows that not

only can a covariance be transported to yield a second covariance, but that the result is indeed the *covariance of the transported data*. Second, it shows that computing a k-dimensional PCA model at T_pM, followed by transporting its implied covariance to T_qM, is the same as transporting the data from T_pM to T_qM and computing the k-dimensional PCA model at T_qM. Thus, while our framework is not limited to low-dimensional models, when dimensionality reduction is needed, we do not have to transport the entire dataset; it is enough to transport the model.

Note that the setup is purely Riemannian. Thus, in the low-dimensional case, in case we want to transport a PGA model using the method described in the theorem, even if the manifold happens to be a Lie group this model is Riemannian and not Lie-algebraic. This is not a limitation: while a Lie-algebraic PGA model is (slightly) simpler to work with in general, in the case of a metric parallel transport it is in fact easier to stay in the Riemannian setting. Moreover, the Riemannian model is more strongly tied to distances than the Lie-algebraic is.

Henceforth, in order to minimize notational clutter, we will drop the "$\widehat{\ }$" notation which was previously used to create a distinction between a parameter (*e.g.* μ) and its estimator (*e.g.*, $\widehat{\mu}$). Returning to our statistical learning problem, we first compute μ_{L} and μ_{S}, the sample intrinsic means of $\mathcal{D}_{\mathrm{L}}$ and $\mathcal{D}_{\mathrm{S}}$ respectively. For the running example from Section 1.3 (which is depicted here again in Fig. 5.7a), which involved many shapes of women but only few shapes of men, these are shown in Fig. 5.8a (μ_{L}) and Fig. 5.8b (μ_{S}).

Let Σ_{L} denote the covariance of $\mathcal{D}_{\mathrm{L}}$, computed at $T_{\mu_{\mathrm{L}}}M$, and Σ_{S} denote the covariance of $\mathcal{D}_{\mathrm{S}}$, computed at $T_{\mu_{\mathrm{S}}}M$. We now compute Σ_{L} and Σ_{S} (Fig. 5.7b). Next, we parallel transport Σ_{L} from $T_{\mu_{\mathrm{L}}}M$ to $T_{\mu_{\mathrm{S}}}M$ along the geodesic between μ_{L} and μ_{S}. We denote the result by Σ_Γ (Fig. 5.7c). In practice, we are typically interested in high-dimensional data so the covariance cannot be stored, let alone computed directly; *e.g.* for body shape our covariance matrix is 258600×258600. Thus, typically we only compute dominant eigenvectors of the covariance as is common in $\mathbb{R}^n$. We compute $V_{\mathrm{L}} \subset T_{\mu_{\mathrm{L}}}M$ and $V_{\mathrm{S}} \subset T_{\mu_{\mathrm{S}}}M$



(a) $\mathcal{D}_{\mathrm{S}}$ and $\mathcal{D}_{\mathrm{L}}$ (b) Σ_{S} and Σ_{L} (c) Σ_{Γ} and Σ_{L}

Figure 5.7: Covariance Transport. (a) $\mathcal{D}_{\mathrm{L}}$ (red) and $\mathcal{D}_{\mathrm{S}}$ (blue). (b) Σ_{L} (red) and Σ_{S} (blue). (c) Σ_{L} (red) and Σ_{Γ} (green). See text for details.

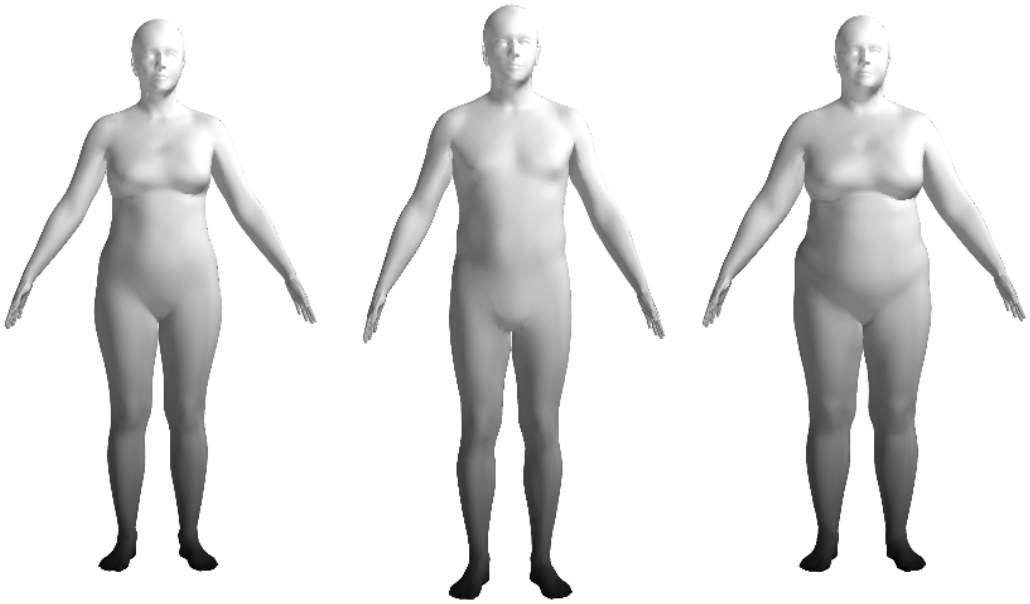
which denote the first K_{L} eigenvectors of Σ_{L} (where $K_{\mathrm{L}} \leq N_{\mathrm{L}}$) and first K_{S} eigenvectors of Σ_{S} respectively (we take $K_{\mathrm{S}} = N_{\mathrm{S}}$).

Similarly, $\sigma_{\mathrm{L}}^2 \in \mathbb{R}^{K_{\mathrm{L}}}$ and $\sigma_{\mathrm{S}}^2 \in \mathbb{R}^{K_{\mathrm{S}}}$ denote the eigenvalues. We now parallel transport V_{L} to $T_{\mu_{\mathrm{S}}}M$, and denote the result by $V_{\Gamma} \subset T_{\mu_{\mathrm{S}}}M$. Despite its adaptation to the curvature of the manifold, V_{Γ} is still orthogonal for the transport is metric. Moreover, by Theorem 5.5.1, through V_{Γ} and σ_{L} we are able to *transport the statistical variation* captured in V_{L} and σ_{L} to the region of M where $\mathcal{D}_{\mathrm{S}}$ resides (Fig. 5.7c). Relying on a principled geometric approach, this simple novel idea leads to remarkable statistical results on real-world data as we will see in Section 5.6. By respecting the underlying geometry, it turns out that V_{Γ} often performs quite well in modeling unseen examples of the small-sample class; *e.g.*, shape variation of women, once transported to the mean shape of only few men, generalizes gracefully.

Of course, we can do even better: we can combine V_{Γ} with V_{S}.

5.5.3 Subspace Fusion and Regularization

Our goal is to use the transported model in order to improve the model learned from the small-sample class. Note that, as V_{Γ} and V_{S} are subspaces of the *same* linear space



(a) (b) (c)

Figure 5.8: Mean body shapes. Intrinsic means computed using 1000 female shapes (a), 50 male shapes (b), and 50 shapes of women with high BMI (c). See text.

$(T_{\mu_S}M)$, we are now back in the Euclidean realm. We here suggest several methods to combine the models. These methods are based on similar ideas from the literature on domain adaptation in Euclidean spaces.

Of course, if the dimension is small enough to work with full covariances, we do not have to use lower-dimensional subspaces and can instead use the full covariance matrices. For example, we can combine models by a convex combination:

$$\Sigma_\lambda \stackrel{\text{def}}{=} \lambda\Sigma_\Gamma + (1-\lambda)\Sigma_S \ , \ 0 \leq \lambda \leq 1 \ . \tag{5.28}$$

In Euclidean spaces, this technique is known as *shrinkage estimation* [129].

Otherwise, let $V_F \subset T_{\mu_S}M$ denote the result of orthonormalization applied to $[V_\Gamma, V_S]$. V_F contains $K_L + K_S$ vectors, and we think of it as the *fusion* of the two models. The fused model is able to generalize better beyond $\mathcal{D}_S$ because it also contains the transported variation from $\mathcal{D}_L$. To enable a direct comparison with V_L or V_Γ, we can also restrict the resulting model to have the same dimensionality by using only K_L vectors. To do so we

introduce the notion of CT *regularization*. Let $X = [x_1, .., x_{N_S}] \subset T_{\mu_S}M$ represent $\mathcal{D}_S$, expressed in $T_{\mu_S}M$. Define $X_\lambda \subset T_{\mu_S}M$ by

$$X_\lambda \stackrel{\text{def}}{=} [\lambda V_\Gamma \sigma_L, (1-\lambda)X] \ , \ 0 \leq \lambda \leq 1 \ , \tag{5.29}$$

and let V_λ represent the K_L-dimensional PCA subspace of X_λ. This is simply weighted PCA, where we treat vectors in V_Γ as examples weighted by their associated standard deviations. In both equations, the larger λ is, the stronger is the influence of $\mathcal{D}_L$. This influence can be thought of as regularization. The value of λ may be chosen by cross-validation.

For summary of the algorithms described above, see Algorithms 1, 2, and 3.

Input: $\mathcal{D}_L = \{p_1, \ldots, p_{N_L}\} \subset M$ and $\mathcal{D}_S = \{q_1, \ldots, q_{N_S}\} \subset M$

Output: Σ_λ

begin

- μ_L, $\mu_S \longleftarrow$ compute the Karcher means for $\mathcal{D}_L$ and $\mathcal{D}_S$ (using, *e.g.*, the algorithm from [113])
- $X_L \longleftarrow$ map $\mathcal{D}_L$ from M to $T_{\mu_L}M$ using Log_{μ_L}
- $X_S \longleftarrow$ map $\mathcal{D}_S$ from M to $T_{\mu_S}M$ using Log_{μ_S}
- $\Sigma_L \longleftarrow$ compute the covariance of X_L
- $\Sigma_S \longleftarrow$ compute the covariance of X_S
- V_L, $\sigma_L \longleftarrow$ compute the eigen-decomposition of Σ_L
- V_S, $\sigma_S \longleftarrow$ compute the eigen-decomposition of Σ_S
- $V_\Gamma \longleftarrow$ apply a metric parallel transport (from $T_{\mu_L}M$ to $T_{\mu_S}M$) to V_L
- $\Sigma_\Gamma \longleftarrow V_\Gamma \, \text{diag}(\sigma_L) V_\Gamma^T$
- $\Sigma_\lambda \longleftarrow \lambda\Sigma_\Gamma + (1-\lambda)\Sigma_S \ , \ 0 \leq \lambda \leq 1$ (λ may be chosen by cross-validation)

end

Algorithm 1: Covariance Transport: the full covariance case

Input: $\mathcal{D}_{\mathrm{L}} = \{p_1, \ldots, p_{N_{\mathrm{L}}}\} \subset M$ and $\mathcal{D}_{\mathrm{S}} = \{q_1, \ldots, q_{N_{\mathrm{S}}}\} \subset M$

Output: V_{F}

begin

- μ_{L}, $\mu_{\mathrm{S}} \longleftarrow$ compute the Karcher means for $\mathcal{D}_{\mathrm{L}}$ and $\mathcal{D}_{\mathrm{S}}$ (using, *e.g.*, the algorithm from [113])
- $X_{\mathrm{L}} \longleftarrow$ map $\mathcal{D}_{\mathrm{L}}$ from M to $T_{\mu_{\mathrm{L}}}M$ using $\mathrm{Log}_{\mu_{\mathrm{L}}}$
- $X_{\mathrm{S}} \longleftarrow$ map $\mathcal{D}_{\mathrm{S}}$ from M to $T_{\mu_{\mathrm{S}}}M$ using $\mathrm{Log}_{\mu_{\mathrm{S}}}$
- V_{L}, $\sigma_{\mathrm{L}} \longleftarrow$ compute the first K_{L} eigen-vectors/eigen-values pairs of the covariance of X_{L} (no need to compute the full covariance)
- V_{S}, $\sigma_{\mathrm{S}} \longleftarrow$ compute the first K_{S} eigen-vectors/eigen-values pairs of the covariance of X_{S} (ditto)
- $V_\Gamma \longleftarrow$ apply a metric parallel transport (from $T_{\mu_{\mathrm{L}}}M$ to $T_{\mu_{\mathrm{S}}}M$) to V_{L}
- $V_{\mathrm{F}} \longleftarrow$ orthogonalize the $K_{\mathrm{L}} + K_{\mathrm{S}}$ vectors in $[V_\Gamma, V_{\mathrm{S}}]$

end

Algorithm 2: Covariance Transport for PGA models: fusion

Input: $\mathcal{D}_{\mathrm{L}} = \{p_1, \ldots, p_{N_{\mathrm{L}}}\} \subset M$ and $\mathcal{D}_{\mathrm{S}} = \{q_1, \ldots, q_{N_{\mathrm{S}}}\} \subset M$

Output: V_λ

begin

- μ_{L}, $\mu_{\mathrm{S}} \longleftarrow$ compute the Karcher means for $\mathcal{D}_{\mathrm{L}}$ and $\mathcal{D}_{\mathrm{S}}$ (using, *e.g.*, the algorithm from [113])
- $X_{\mathrm{L}} \longleftarrow$ map $\mathcal{D}_{\mathrm{L}}$ from M to $T_{\mu_{\mathrm{L}}}M$ using $\mathrm{Log}_{\mu_{\mathrm{L}}}$
- $X_{\mathrm{S}} \longleftarrow$ map $\mathcal{D}_{\mathrm{S}}$ from M to $T_{\mu_{\mathrm{S}}}M$ using $\mathrm{Log}_{\mu_{\mathrm{S}}}$
- V_{L}, $\sigma_{\mathrm{L}} \longleftarrow$ compute the first K_{L} eigen-vectors/eigen-values pairs of the covariance of X_{L} (no need to compute the full covariance)
- $V_\Gamma \longleftarrow$ apply a metric parallel transport (from $T_{\mu_{\mathrm{L}}}M$ to $T_{\mu_{\mathrm{S}}}M$) to V_{L}
- $X_\lambda \longleftarrow [\lambda V_\Gamma \sigma_{\mathrm{L}}, (1-\lambda) X_{\mathrm{S}}]$, $0 \leq \lambda \leq 1$, (λ may be chosen by cross-validation)
- $V_\lambda \longleftarrow$ K_{L}-dimensional PCA subspace of X_λ

end

Algorithm 3: Covariance Transport for PGA models: regularization

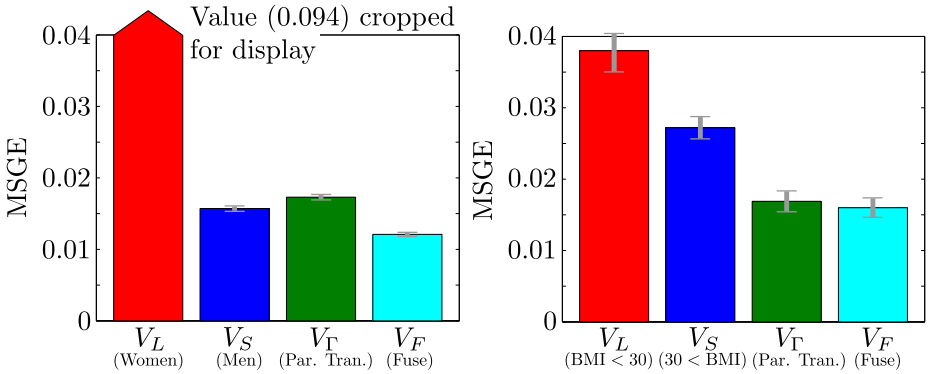


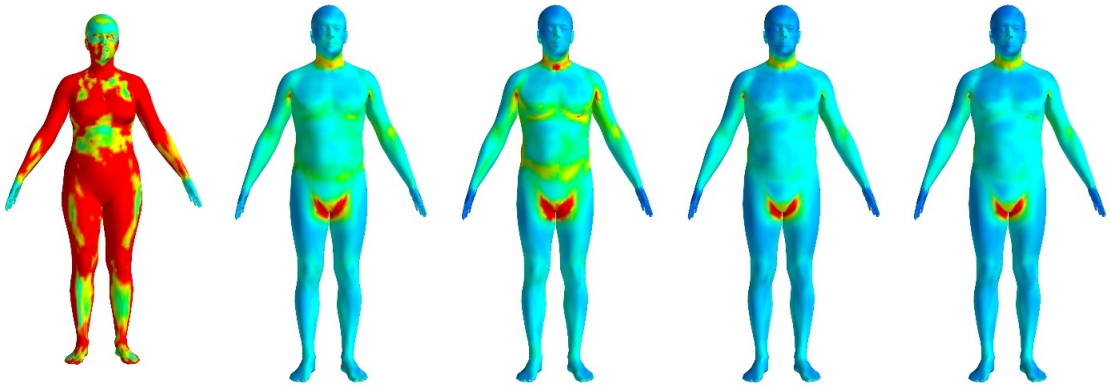
Figure 5.9: Summary for body shape experiments. *Left:* Gender. *Right:* BMI. In each subfigure, the four bars represent the overall reconstruction error for four models: V_{L}, V_{S}, V_{Γ}, and V_{F}. For a given model, the height of the column represents the reconstruction error (measured in terms of squared geodesic distances on G_T – the Lie group of triangle deformations – averaged over both the entire test dataset as well as all of the N_{tri} triangles in the mesh. See text for an interpretation of these results.

5.6 Results

While CT is applicable on numerous manifolds, we here focus on human shape modeling. For additional experiments with image descriptors, involving the manifold of SPD matrices, see [42]. For our experiments here, we use the manifold of mesh deformations we introduced in Chapter 4. Recall that at the core of this manifold, there is a six-dimensional manifold, called G_T, which in turn is built on three other smaller ones: SO(3), G_A, and G_S. Note that in Chapter 4 dimensionality-reduction was achieved by Lie-algebraic PGA. In contrast, here we do Riemanninan PGA. Additionally, in this chapter we also use another dataset of meshes – with better alignment and better pose normalization – that was not available to us when we performed the experiments in that chapter.

5.6.1 From Venus to Mars

We consider a challenging scenario to illustrate the surprising power of CT. Here $\mathcal{D}_{\mathrm{L}}$ and $\mathcal{D}_{\mathrm{S}}$ consist of shapes of $N_{\mathrm{L}} = 1000$ women and $N_{\mathrm{S}} = 50$ men respectively. First we use an iterative algorithm [113] to compute the respective intrinsic means, μ_{L} (Fig. 5.8a) and



(a) V_{L} (b) V_{S} (c) V_{Γ} (d) V_{F} (e) V_{λ}

Figure 5.10: Model mean error: Genders. Blue and red indicate small and large errors respectively. The heat maps are overlaid over the points of tangency associated with the models: μ_{L} for (a), and μ_{S} for (b-e). See text for details.

μ_{S} (Fig. 5.8b). We then compute the Riemannian PGA subspaces with $K_{\mathrm{L}} = 200$ and $K_{\mathrm{S}} = 50$. Thus, V_{L} is a 200-dimensional subspace of $T_{\mu_{\mathrm{L}}}M$, while V_{S} is a a 50-dimensional subspace of $T_{\mu_{\mathrm{S}}}M$. Next, we use the LC parallel transport to move V_{L}, from $T_{\mu_{\mathrm{L}}}M$ to $T_{\mu_{\mathrm{S}}}M$, to produce V_{Γ}, a 200-dimensional subspace of $T_{\mu_{\mathrm{S}}}M$. An animated illustration is available online at:

http://www.dam.brown.edu/people/freifeld/shapes/CovTransport.mpg

For an explanation of the contents of this movie, please see the following file:

http://www.dam.brown.edu/people/freifeld/shapes/CovTransport_ReadMe.txt

Finally, we create two additional subspaces of $T_{\mu_{\mathrm{S}}}M$. The first is the fused model, V_{F}, a 250-dimensional subspace. The second is the regularized model V_{λ}, a 200-dimensional subspace. Both these models were obtained using the procedures described in Section 5.5.3.

We evaluate performance on 1000 test male shapes. As explained in Chapter 4, for each test mesh, denoted by $\mathcal{M}_i$, we extract its associated mesh deformation, denoted by p_i. Our goal here is to evaluate the reconstruction of the p_i's using the different models. The process has similarity to our reconstruction experiments from Chapter 4. However, as there we use Lie-algebraic PGA models while here we used Riemannian PGA models,

there are slight differences in the projection and reconstruction procedures so it is worth going over the details. Let V denote any one of the 5 models of interest: V_{L}, V_{S}, V_{Γ}, V_{F}, and V_{λ}. Let μ denote the point of tangency associated with the model. For V_{S}, this point is μ_{L}, while for the other 4 models it is μ_{S}. To project p_i on V we compute the subspace coefficients $V^T \mathrm{Log}_{\mu}(p_i)$. To reconstruct the mesh deformation, we map the coefficients back to M by setting $p_i^{\mathrm{recon}} = \mathrm{Exp}_{\mu}(V^T \mathrm{Log}_{\mu}(p_i))$. We then compute the geodesic distance between p_i and p_i^{recon}. We think of this distance as the reconstruction error. In fact, since $M = G_T^{N_{tri}}$, this is done by first computing the geodesic distance associated with the deformation of each one the triangles in $\mathcal{M}_i$. This enables us to define reconstruction error in terms of Squared Geodesic Error (SGE), computed per triangle[10]. We also, for a fixed i, average the SGE over all triangles in $\mathcal{M}_i$ to compute the Mean SGE (MSGE). This defines the performance of the model for p_i, the mesh deformation. Finally, the overall performance of the model is defined by averaging the MSGE over all test examples. The MSGE results are summarized in Fig. 5.9 (left). To visualize the performance, we average the SGE, per triangle, over all test examples and display these per-triangle errors over the mesh associated with the point of tangency of the model (Fig. 5.10).

Figure 5.10a shows that V_{L} performs very poorly; a shape model of women fails to model men. While the errors for V_{S} are much lower (Fig. 5.10b), there are still noticeable errors due to poor generalization. The surprise is Fig. 5.10c, which shows the result for V_{Γ}: the parallel transport dramatically improves the female model (Fig. 5.10a) to the point it fares comparably with the male model (Fig. 5.10b), although *the only information used from the male data is the mean.* Combining transported and local models lets us do even better. Figure 5.10d shows the result of V_{F}; it significantly improves over V_{S} or V_{Γ}. Figure 5.10e shows the regularized model, V_{λ}, which has the same dimensionality as V_{L} and still performs well. Figure 5.11 shows selected results for test bodies; see Appendix B for additional results and reconstructions.

[10] Geodesics with respect to G_T, not to the 2D body surface.

5.6.2 From Normal-Weight to Obesity.

A good statistical shape model of obese women is important for fashion and health applications[11] but is difficult to build since the data are scarce as reflected by their paucity in existing body shape datasets. Here $\mathcal{D}_{\mathrm{L}}$ stands for 1000 shapes of women with BMI ≤ 30 while $\mathcal{D}_{\mathrm{S}}$ consists of only 50 shapes of women with BMI > 30. Figure 5.8c shows μ_{S}; μ_{L} is not shown as it was very similar to the μ_{L} from the gender experiment. Figures 5.9 (right) and 5.12 summarize the results. Compared with the gender experiment there are two main differences: 1) Here V_Γ is already much better than V_{S} so fusion only makes a small difference. 2) Error bars in are larger (Fig. 5.9, right) than before (Fig. 5.9, left) due to the limited amount of *test* data available for high-BMI women; this is truly a small-sample class: we were able to obtain only 50 test examples. Compared with using V_{S}, mesh reconstruction is noticeably improved using our method (V_{F}). In both the gender and BMI experiments, results for V_λ look nearly identical to V_{F}, and are not shown. See Appendix B for individual reconstruction results.

5.7 Conclusion

We have described a principled approach for transporting a covariance matrix across a Riemannian manifold and for using the transported covariance to improve the estimation of another covariance on that manifold. This goes beyond previous methods of domain adaptation and transfer learning for $\mathbb{R}^n$-valued data. We have proven that the parallel transport of a covariance matrix preserves the statistics as long as the parallel transport is metric. The approach is broadly applicable to any manifold in which a metric parallel transport can be computed. This includes, but is not limited to, mesh deformation models as we showed here by our experiments and image descriptors (omitted here; see [42] for

[11] BMI is known to be a poor predictor of risk for cardiovascular disease and diabetes, while body shape, as captured by hip to waist ratio, is better. An open question, and beyond our scope, is whether more detailed body shape properties are better predictors. Accurate shape modeling is the first step toward an answer.

more details).

Finally, while LC is often considered a natural choice, and we find it works very well for CT of real-world data, future work should explore implications of other types of metric parallel transport on CT.

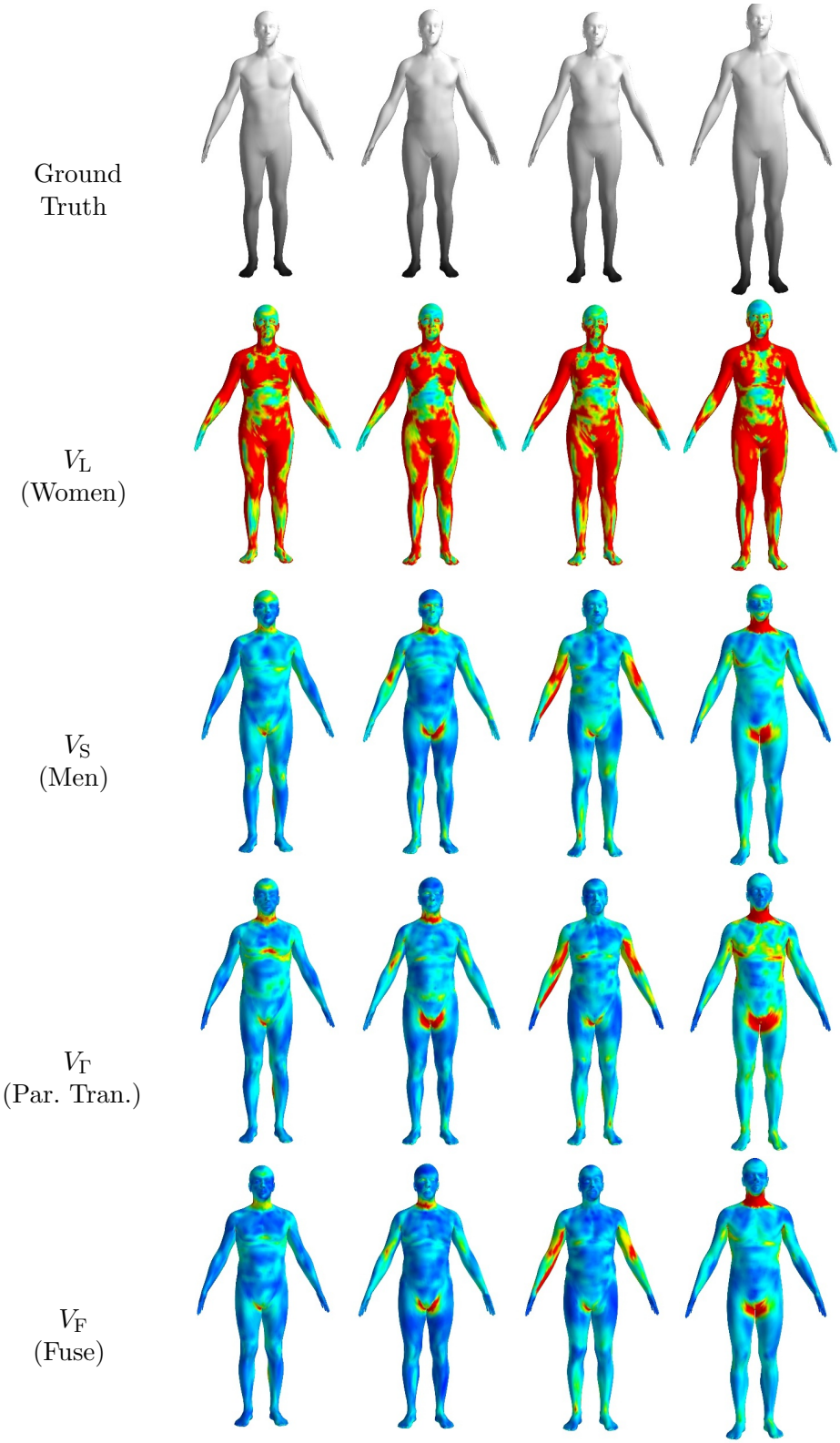
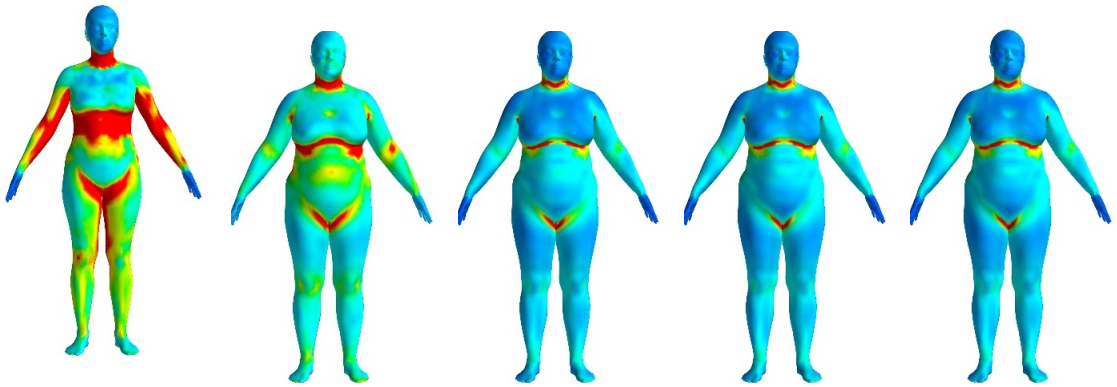


Figure 5.11: Selected results: Gender. Each column represents a different test body. The heat maps are overlaid on the reconstructions using different models.



(a) V_{L} (b) V_{S} (c) V_{Γ} (d) V_{F} (e) V_{λ}

Figure 5.12: Model mean error: BMI. Analogous to Fig. 5.10.

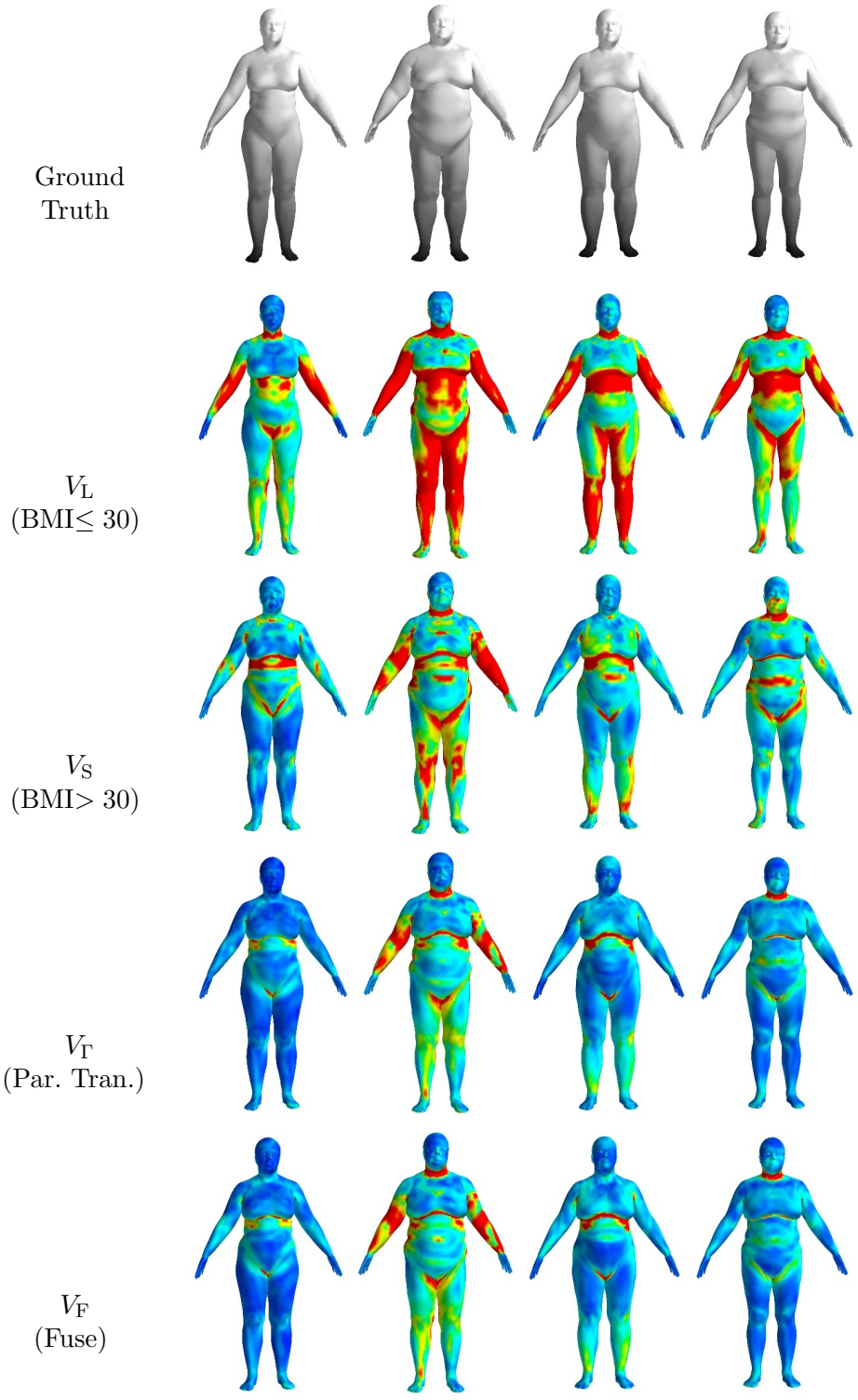


Figure 5.13: Selected results: BMI. Analogous to Fig. 5.13.

Chapter 6

Final Words and Future Directions

Throughout this work we showed how 2D and 3D deformable shapes are represented via matrix Lie groups, how to build statistical models on these manifolds, and the advantages of such representations over standard Euclidean approaches.

Several of the key ideas behind the Contour Person model (Chapter 3), which we first introduced in [43], have already provided the basis for additional models and applications of 2D articulated-but-detailed human shape; see our recent works (Guan *et al.* [57] and Zuffi *et al.* [155]), as well as Zuffi and Black [154] and Jhuang *et al.* [75].

Our CP model is view-based and in Chapter 3 we have only shown examples for frontal bodies. A key next step is to extend this to other discrete views. A general solution to the human pose and shape estimation problem will require an inference method to search over a discrete set of views; however, for achieving this aim, rather than using the single-contour approach described here, it is preferable to employ a multiple-contour approach such as the one we have recently employed in Zuffi *et al.* [155]. In that work, we built a variant of the CP representation that supports an arbitrary range of poses (including more complicated self-occlusions).

We expect that due to their superiority as generative models, the emerging detailed 2D models of articulated shape, perhaps together with richer appearance models (*e.g.*, see [78]), will gradually replace the traditional Pictorial Structures models.

The Lie Shapes representation (Chapter 4), which we first introduced in [41], provides an elegant way to represent deformable shape without many of the problems associated with Euclidean representation while keeping a low computational cost. We showed how this representation leads to better distances and better statistical models.

Future work should incorporate our representation into factored statistical models such as SCAPE [6] (which factors body-shape variation into separate pose, shape, and pose-dependent deformations) or its derivative DRAPE [58] (which adds a clothing deformation layer on top of SCAPE). Here our group structure is ideal for composition of deformations. For example, SCAPE factorizes a local deformation Q as $R_{\text{pose}}Q_{\text{shape}}Q_{\text{po.dep.}}$, where R_{pose} is a rotation matrix that affects the entire body part, while Q_{shape} and $Q_{\text{po.dep.}}$ are Euclidean deformations and po.dep. is short for 'pose-dependent'. With our representation, this will now have form

$$Y = \overbrace{R_{\text{pose}}R_{\text{shape}}R_{\text{po.dep.}}}^{R\in\text{SO}(3)}R_X^T\overbrace{A_{\text{shape}}A_{\text{po.dep.}}}^{A\in G_A}\overbrace{S_{\text{shape}}S_{\text{po.dep.}}}^{S\in G_S}R_X X. \tag{6.1}$$

This composition is consistent and more natural as here the different kinds of the deformations (rotation, planar, scaling) are grouped accordingly.

As our representation can be used through a simple drop-and-replacement within existing models, we anticipate wide application for Lie shapes in computer vision and computer graphics. In particular, the ability to easily compute meaningful averages on the manifold of mesh deformations suggests that classical problems such as key-frames, shape interpolation, and blend skinning, should be revisited so their solutions can benefit from the advantages of our representation. Each one of the three components in our representation of a local deformation (R, A and S) can be modeled using a different method. For example, consider blend skinning: a particular part of a skeleton may have different influences on the

R, A, and S of a particular triangle of interest. In addition, the averaging associated with the blending should be done directly on the manifold of triangle deformations using the intrinsic mean, and is thus expected to yield better results than the traditional Euclidean averaging. More generally than blend skinning, there is no reason to expect that the optimal type of statistical distribution for modeling rotations (R) should be the same as the one for the planar deformations (A) or the scaling (S). We expect that a considerable gain will be achieved once statistical models take this into account.

On the manifold of mesh deformations, in addition to defining new parametric statistical models (*e.g.*, SCAPE-like), it is possible to define models based on nonparametric statistics (as was done, *e.g.*, in [10] for other shape spaces). For example, in Ghosh *et al.* [48] triangle deformations are modeled using a Bayesian nonparametric statistical model in order to automatically segment an articulated mesh into its different parts. We plan to explore the effect of adapting their model to use our manifold-based representation.

Likewise, Markov-random-fields methods for mesh regularization can be adapted to take into account geodesic distances between local deformations of adjacent triangles; that is, the local potentials will be defined through statistics on manifolds rather than the inferior Euclidean statistics. Closely related to this line of thinking is the notion of *natural 3D shape statistics*: similarly to natural image statistics, an ensemble of real-world 3D shapes presents us with a way to define and learn the local statistical characteristics of shape deformations. To the best of our knowledge, and unlike the well-studied case of image statistics, this direction has yet to be explored; we believe that our manifold provides a natural tool for such statistics. Additional concepts related to graphical models, such as belief propagation, may also be generalized to work with manifold-valued data. Specifically in our context, it makes sense to represent the skeleton of a 3D human shape via a graphical model whose random variables at the nodes represent the joints and take values in SO(3). Coupled with our manifold of mesh deformation to represent the skin, this gives a unifying probabilistic representation for variability of 3D human shapes using nonlinear manifolds.

Our manifold-based representation, much like its related Euclidean representations, is based on deformations of single triangles. It is possible to push the manifold representation further by explicitly encoding more constraints on the way adjacent triangles deform. For example, the deformation of a pair of triangles sharing an edge is fully defined by 9 DoF (instead of $12 = 6 \times 2$) and so. Here we decided against such an approach in order to keep the representation simple as well as to maintain the Lie group structure. However, because of the possible gain that can be achieved due to the reduction of DoF, it is worth pursuing research on such more complicated manifolds of triangle deformations even if these manifolds will not be Lie groups.

An additional promising future research direction is to combine the Lie shapes representation with the emerging new information-theoretic methods on Lie-groups; see, *e.g.*, the recent textbooks by Chirikjian [20, 21]. While there are some obstacles here – for example, G_A is not unimodular – such an approach can be very useful to answer questions such as the following: across a corpus of meshes of human bodies, which triangles are more informative than others? A satisfying answer to this question can lead to useful modification of the Riemannian metric.

We also showed how to generalize a transfer learning idea from $\mathbb{R}^n$-valued data to manifold-valued data (Chapter 5). This new tool for statistics on manifolds is not restricted to either shape-deformation manifolds or matrix Lie groups. Rather, it is applicable to every geodesically-complete Riemannian manifold. We expect that this tool will provide the basis for tackling interesting problems related to scarce manifold-valued data. We here focused on covariance matrices and Riemannian PGA models. The method applies transparently to other types of linear subspaces. for example, the parallel-transport-based framework suggested by Wei *et al.* [147] for learning image deformations can be readily improved by switching from Lie-algebraic PGA models and non-metric parallel transport to Riemannian PGA models and a metric parallel transport.

We view Riemannian covariance transport as a particular case of the more general – and

yet-to-be-explored – case of *Riemannian statistics transport*. Future work will extend the framework to more complex statistics, *e.g.* by modeling multi-modal data with transported mixtures. Other learning problems like canonical correlation analysis, weakly-paired data [85], probabilistic PGA, and classification could be treated similarly.

In particular, Riemannian statistics transport provides a natural tool for transporting the statistics of shapes of few people in many poses to shapes of new people of whom only one pose is available. Consequently, this should enable reposing people in new poses in a more principled way than the current state-of-the-art approaches. We expect this kind of tool to have an important impact on applications in, *e.g.*, animating avatars.

But let us now go back to the issue of covariance transport. In some sense, parallel transport is nothing more than a generalization, from Euclidean spaces to manifolds, of ordinary translation. Thus, it is a useful tool whenever we want to generalize techniques from Euclidean statistics that employ (even if only implicitly) translation. On the face of it, it seems that the other side of the coin is negative; that is, if in a given situation with manifold-valued data we have no reason to expect that translation would have helped us in a hypothetical analogous Euclidean scenario, then it seems like parallel transport will have no merit. We argue, however, that a useful statistical lemonade can be still made from these lemons. This is because while parallel transport indeed generalizes translation, this generalization depends on the curve along which the parallel transport is done as well as on the Riemannian metric (recall our discussion is restricted to *metric* parallel transport). Importantly, while in Chapter 5 we focused on parallel transport along geodesics, both the covariance transport framework and the covariance transport theorem hold for any differentiable curve in M. It should be understood that choices of different curves, between the same two points, lead to different results in terms of the transported covariance. An interesting future research direction is to explore how the performance of the transported covariance can guide us in picking the best path between two points on M – and this path need not be a geodesic. It is also interesting to consider integrating the results over an ensemble of possible curves, possibly weighted by how probable each curve is. As we

pointed out, the transported covariance is affected by not only the choice of the curve but also the Riemannian metric. Thus, the performance of the transported covariance may help us understand what choices for Riemannian metric are better than others. Moreover, throughout this dissertation we focused on known manifolds. However, we believe that combining the covariance transport (presented here) with the learning of a Riemannian metric (as we presented in Hauberg *et al.* [65]) will prove to be a powerful tool[1].

To summarize, our final conclusion from this dissertation is that before diving into the statistical modeling, it behooves us to pay attention to the problem at hand and to consider whether the space of interest has a particular geometrical structure that can be exploited.

[1]The idea of working with these two new tools in conjunction was first suggested by Søren Hauberg.

Appendix A

Mathematics

In this appendix, we cover several mathematical definitions that are referred to throughout this work. The appendix is by no means intended to serve as a complete reference to the topics it touches upon. Rather, it is only meant to provide a quick reference to make the main text more self-contained.

A.1 Topology

Let us motivate the discussion. A *topological space* is a concept more general than a *metric space*: every metric space is a topological space but the converse is false; *i.e.*, there are topological spaces on which no metric can be defined. Compared with topological spaces, the notion of metric spaces is less abstract in the sense it is easy to wrap our mind around a space in which we can measure distances between points, convergence can be defined through the notion of a distance, a limit point of a convergent sequence is unique, and so on. In most practical problems in computer vision and pattern theory, the spaces of interest are metric spaces, and not merely topological. This raises the question: why should we care about topological spaces? One reason is that often it is useful to define and

understand concepts in their most abstract setting. One example is that using topological terminology it is possible to define manifolds without having to rely on an ambient space; even if there is a natural ambient space, this kind of a definition relieves us from many annoying technical details. Another concrete example is that topological terminology helps us to see immediately that full-rank $n \times p$ rectangular matrices form an open subset (and thus, an open submanifold) of $\mathbb{R}^{n\times p}$. In particular, this holds for invertible matrices and thus we get that invertible matrices form an open set. Of course, one can reach the same conclusion by sticking to metric space terms, but this requires a significantly greater effort. The main reference in this Section is [108].

Definition A.1.1 (Topological space)**.** Let X be space. A *topology* on X, is any collection τ of subsets of X such that:

1. The $\emptyset$ and X are in τ;
2. The intersection of the elements of any finite subcollection of τ is in τ:

$$\{U_1, U_2, \ldots, U_n\} \subset \tau \Longrightarrow \bigcap_{i=1}^{n} U_i \in \tau \ . \tag{A.1}$$

3. The union of the elements of any subcollection of τ is in τ:

$$\{U_\alpha\}_{\alpha\in I} \subset \tau \Longrightarrow \bigcup_{\alpha\in I} U_\alpha \in \tau \ , \tag{A.2}$$

   where I is some index set[1].

The ordered pair (X, τ) is called a *topological space*.

When τ is understood from the context, we often omit τ and say that "X is a topological space" although, strictly speaking, different topologies on X give rise to different topological spaces.

[1] Note this includes cases when I is uncountable.

Remark A.1.1. The word "space" in the first sentence of Definition A.1.1 means the exact same thing as the word "set": a collection of elements. Since we often refer to subsets of the set X as "sets" themselves, it is convenient to make an artificial distinction between X and its subsets by using the term "space" for X. Of course, since $X \subset X$, we may also regard X as its own subset.

Definition A.1.2 (Open and closed sets)**.** If (X, τ) is a topological space, the members of τ are called *open sets* (or open subsets of X). A complement of an open set (with respect to X) is called *closed*.

Example A.1.1 (Sets that are both open and closed)**.** It is easy to see that X and $\emptyset$ are both open and closed. If some additional conditions hold, then X and $\emptyset$ are the only sets that are both open and closed.

Definition A.1.3 (Open cover)**.** If (X, τ) is a topological space, then any subcollection of τ whose union is X is called an *open cover* of X.

In particular, since $X \in \tau$ (by Definition A.1.1), τ itself is an open cover of X, and thus there always exists at least one open cover.

Remark A.1.2. As pointed out by Royden [126], the term 'open cover' is an abuse of language: the adjective 'open' refers to the sets themselves, not to the cover. For example, in the term 'finite cover', the adjective 'finite' refers to the cover and does not imply that each one of the sets is finite. Be that as it may, this terminology is well established in mathematics.

Definition A.1.4 (Second-countable space)**.** A topological space (X, τ) is said to be *second-countable* if its topology has a countable base; namely, if there exists a countable collection $\mathcal{U} = \{U_i\}_{i=1}^{\infty}$, where $\mathcal{U} \subset \tau$, such that every $U \in \tau$ can be written as a union of elements of some subcollection of $\mathcal{U}$.

Fact A.1.1. A countable product of second-countable spaces is second-countable.

Definition A.1.5 (Hausdorff space)**.** A topological space X is called *Hausdorff* if for

every pair of points $x, y \in X$, there exist open sets U and V such that $x \in U$, $y \in V$, and $U \cap V = \emptyset$.

Fact A.1.2. A product space of Hausdorff spaces is Hausdorff.

Example A.1.2. The real line, $\mathbb{R}$, is Hausdorff. $\mathbb{R}^n$ is Hausdorff since it is the product space of n copies of $\mathbb{R}$. Likewise, $\mathbb{R}^n$ is second-countable since $\mathbb{R}$ is second countable: the collection of all open intervals whose endpoints are rational numbers is a countable bases for the standard topology on $\mathbb{R}$.

Remark A.1.3. Every Hausdorff space is T_1; namely, if $x \in X$, then the singleton $\{x\}$ is a closed set.

Example A.1.3. $\{0\} \subset \mathbb{R}$ is closed, and thus its complement $\mathbb{R} - \{0\}$ is open.

Definition A.1.6 (Continuous function)**.** Let X and Y be topological spaces. A function $f : X \to Y$ is called *continuous* if $f^{-1}(V)$ is an open subset of X whenever V is an open subset of Y.

Remark A.1.4. Definition A.1.6 holds even if the spaces X and Y are not metric spaces. However, all spaces of interest in our work are metric spaces. For metric spaces, an equivalent definition for a continuous function can be made using the "$\varepsilon - \delta$" machinery that is familiar from Calculus and does not require knowledge of topology. That said, Definition A.1.6 (in addition to being analogous to the concept of measurable functions or random variables) is still useful even in the context metric spaces as the following example illustrates.

Example A.1.4. The determinant, $\det : \mathbb{R}^{n \times n} \to \mathbb{R}$, is a continuous function, as can be shown by expanding the determinant using the Leibniz formula.

Example A.1.5. Invertible matrices form an open subset of $\mathbb{R}^{n \times n}$. This follows from Definition A.1.6, Example A.1.3 and Example A.1.4.

Corollary A.1.1 (Composition of continuous functions yields a continuous function)**.** *By Definition A.1.6, if X,Y,Z are topological spaces and $f : X \to Y$ and $g : Y \to Z$ are continuous, then so is $g \circ f : X \to Z$*

Proof. Consider $(g \circ f)^{-1}(W) \overset{\text{by def.}}{=} f^{-1}(g^{-1}(W))$ where $W \subset Z$ is open and note that $g^{-1}(W)$ is open. □

Example A.1.6 (Matrix multiplication is continuous)**.** Matrix multiplication,

$$\mathbb{R}^{n_1 \times n_2} \times \mathbb{R}^{n_2 \times n_3} \to \mathbb{R}^{n_1 \times n_3} \ : \ (A, B) \mapsto AB,$$

is a continuous function.

Proposition A.1.2. *Let $n_1 \geq n_2$. The set of full-rank rectangular matrices is open (as a subset of $\mathbb{R}^{n_1 \times n_2}$).*

Before the proof, let us recall the definition of a full-rank matrix:

Definition A.1.7 (Full-rank matrix)**.** Let $n_1 \geq n_2$. A "skinny-tall" rectangular matrix $A \in \mathbb{R}^{n_1 \times n_2}$ is said to be full rank if $rank(A) = n_2$, or equivalently, if $\det(A^T A) \neq 0$.

Thus the set of full rank matrices is given by

$$\left\{A \in \mathbb{R}^{n_1 \times n_2} : A \in f^{-1}(\mathbb{R} - \{0\})\right\} \tag{A.3}$$

where

$$f : \mathbb{R}^{n_1 \times n_2} \to \mathbb{R} \ : \ A \mapsto \det(A^T A). \tag{A.4}$$

Proof. Let f be as in Eqn. (A.4). By Example A.1.3, the set $\mathbb{R} - \{0\}$ is open. Thus, by Definition A.1.6 it is enough to show that f is continuous, since this would imply that $f^{-1}(\mathbb{R} - \{0\})$ is open in $\mathbb{R}^{n_1 \times n_2}$. The continuity of f follows from Example A.1.4, Example A.1.6 and Corollary A.1.1. □

Proposition A.1.2 generalizes Example A.1.5.

Definition A.1.8 (Homeomorphism)**.** Let X and Y be two topological spaces. A function $f : X \to Y$ is called a *homeomorphism* if f is a continuous invertible function such that f^{-1} is also continuous. In which case, X and Y are said to be *homeomorphic* to each other.

A homeomorphism should not be confused with a *homomorphism*.

Remark A.1.5. The requirement for the continuity of f^{-1} is not superfluous; there exit examples for an invertible continuous function whose inverse is not continuous.

Definition A.1.9 (Connected space)**.** A topological space (X, τ) is said to be *connected* if it cannot be written as the union of two disjoint nonempty open sets. In other words, $X \neq U \cup V$ for every choice of two nonempty sets $U, V \in \tau$.

Definition A.1.10 (Path-connected space)**.** A topological space (X, τ) is called *path-connected* if for every two points p and q in X, there exists a continuous curve, $c : [0, 1] \to X$, such that $c[0] = p$ and $c[1] = q$.

Path-connectedness is a stronger requirement than connectedness. However, under some mild conditions (that hold for all the manifolds used in this work), a connected space is also path-connected.

A.2 Calculus

Definition A.2.1 (Smooth function from $\mathbb{R}^n$ to $\mathbb{R}^m$)**.** If U and V are open subsets of $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively, a function $f : U \to V$ id said to be *smooth* if each of its component functions has continuous partial derivatives of all order.

Definition A.2.2 (Diffeomorphism between two open subsets of $\mathbb{R}^n$)**.** Let $f : U \to V$ be smooth function from an open subset of $\mathbb{R}^n$ to an open subset of $\mathbb{R}^m$. If f is not only smooth but also bijective (which implies $n = m$) with a smooth inverse, then f is called a *diffeomorphism*.

A.3 Real Analysis

Definition A.3.1 (Metric space)**.** A *metric space* is an ordered pair (M, d) where M is a set[2] and $d : M \times M \to \mathbb{R}$ is a *metric* on M; namely, for every $x, y, z \in M$, the following properties hold:

$$d(x, y) \geq 0 \tag{A.5}$$

$$d(x, y) = 0 \iff x = y \tag{A.6}$$

$$d(x, y) = d(y, x) \tag{A.7}$$

$$d(x, z) \leq d(x, y) + d(y, z) \ . \tag{A.8}$$

When it is understood from the context which metric is used, we usually refer to M alone as a metric space (*i.e.*, omitting d), although strictly speaking it is the (M, d) *pair* that forms the metric space; different metrics on the same set M give rise to different metric spaces. We usually think of d as a *distance function*. Note that a metric space need not be linear. In particular, if M is a manifold (see Chapter 2), then it is also space (or a set). To make the manifold a metric space, we need to define a metric d.

Definition A.3.2 (Normed vector space)**.** A *Normed vector space* is an ordered pair $(V, \|\|)$ where V is a vector space and $\|\| : V \to \mathbb{R}$ is a *norm* on V; namely, for every $x, y \in V$, and every $\alpha \in \mathbb{R}$, the following properties hold:

$$\|\alpha x\| = |\alpha| \|x\| \tag{A.9}$$

$$\|x + y\| \leq \|x\| + \|y\| \tag{A.10}$$

$$\|x\| = 0 \Rightarrow x = 0_V \tag{A.11}$$

where 0_V is the zero vector of V.

Remark A.3.1. Every norm implies a metric. The converse is false. If the metric space is

[2] Or a space – the words "set" and "space" can be used interchangeably here.

not linear, then a norm cannot even be defined; however, it is worth pointing out that the converse is still false even if the space is linear.

Definition A.3.3 (Frobenius norm)**.** Let $A \in \mathbb{R}^{m\times n}$. Its *Frobenius norm* is given by

$$\|A\|_F = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} A_{i,j}^2 \right)^{1/2} . \tag{A.12}$$

Definition A.3.4 (Inner-product space)**.** An *inner-product space* (over the field $\mathbb{F}$) is an ordered pair $(V, \langle \cdot, \cdot \rangle)$ where V is a vector space and $\langle \cdot, \cdot \rangle : V \times V \to \mathbb{F}$ is an *inner product* on V; namely, for every $x, y, z \in V$ and every $\alpha, \beta \in \mathbb{F}$, the following properties hold:

$$\langle x, y \rangle = \overline{\langle y, x \rangle} \tag{A.13}$$

$$\langle \alpha x + \beta y, z \rangle = \alpha \langle x, z \rangle + \beta \langle y, z \rangle \tag{A.14}$$

$$\langle x, x \rangle \geq 0 \tag{A.15}$$

$$\langle x, x \rangle = 0 \iff x = 0 . \tag{A.16}$$

Every inner product implies a norm:

$$\|x\| = \sqrt{\langle x, x \rangle} , \tag{A.17}$$

the converse is false.

Definition A.3.5 (Angle)**.** Let $(V, \langle \cdot, \cdot \rangle)$ be an inner-product space, and let $x, y \in V$. The *angle* between x and y is defined by:

$$\text{angle}(x, y) = \arccos \frac{\langle x, x \rangle}{\|x\| \cdot \|x\|} . \tag{A.18}$$

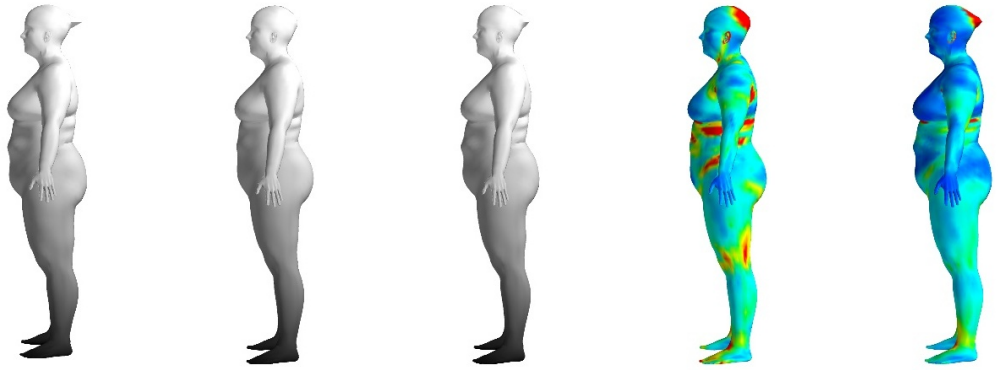
Appendix B

Additional Results

This appendix contains additional results for Chapter 5.

See figures in the next pages. We first show results for 10 test examples from the BMI experiment, and then another 10 from the gender experiment. Each figure shows, from left to right: Ground Truth; V_{S} reconstruction; V_{F} reconstruction; V_{S} reconstruction error; V_{F} reconstruction error. Frontal view is shown in the top row and profile is shown in the bottom. Typically differences in reconstruction results are most noticeable in regions where the corresponding error maps differ significantly in colors.

(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}



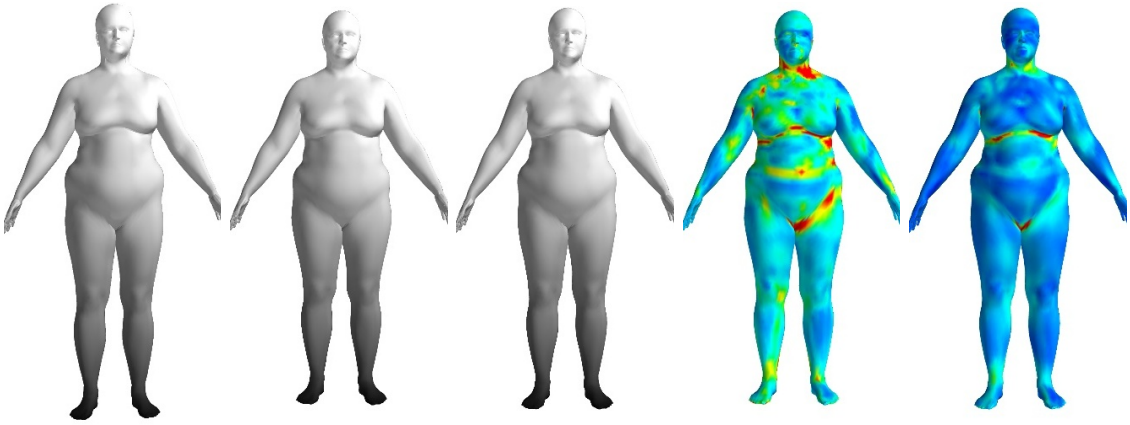
(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

Figure B.1: BMI reconstruction example.

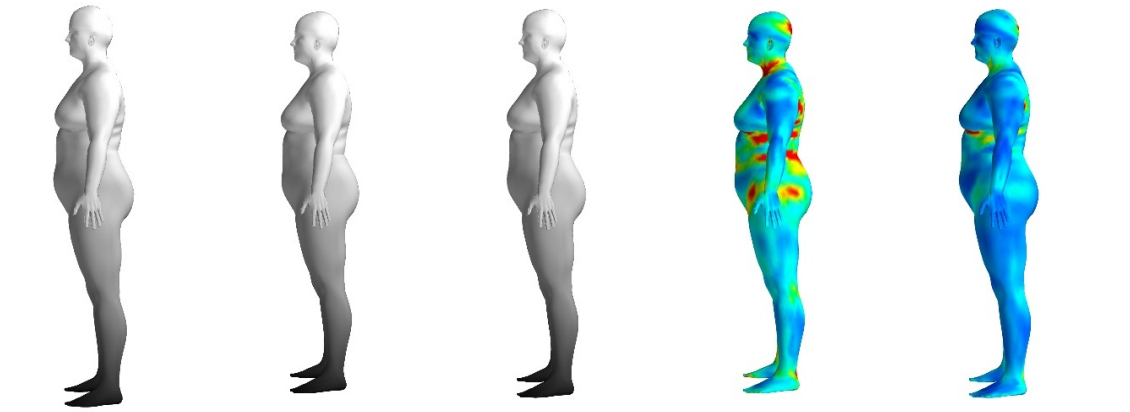
(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}

(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

Figure B.2: BMI reconstruction example.

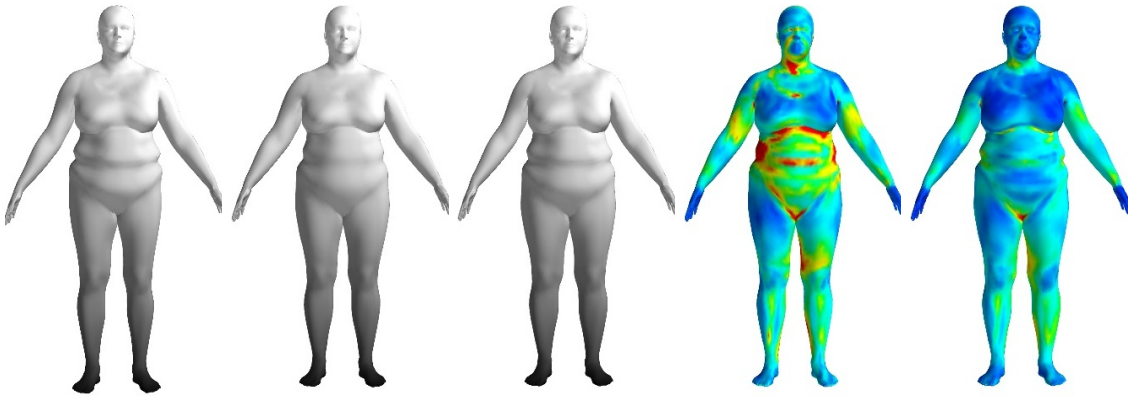


(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}

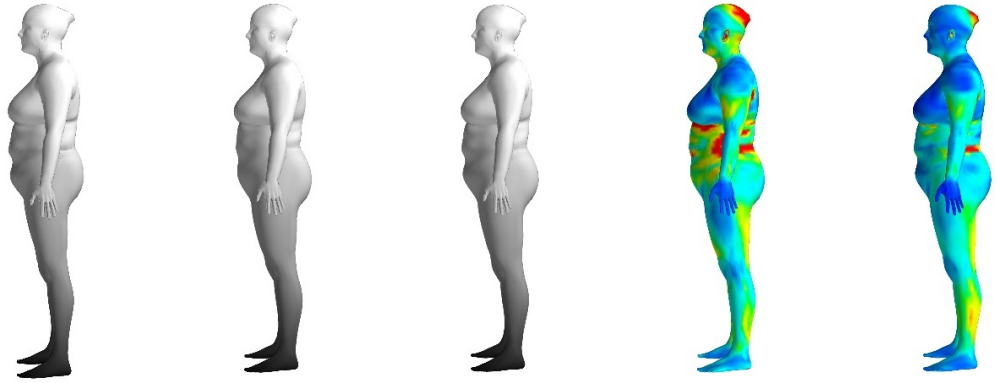


(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

Figure B.3: BMI reconstruction example.



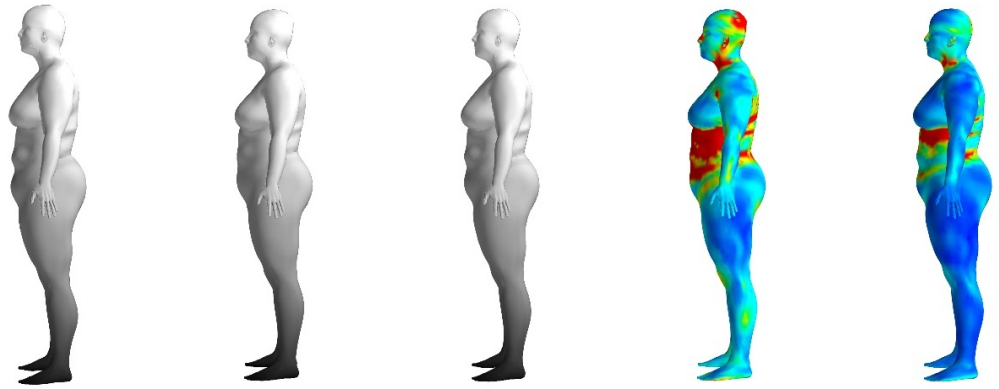
(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}



(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

Figure B.4: BMI reconstruction example.

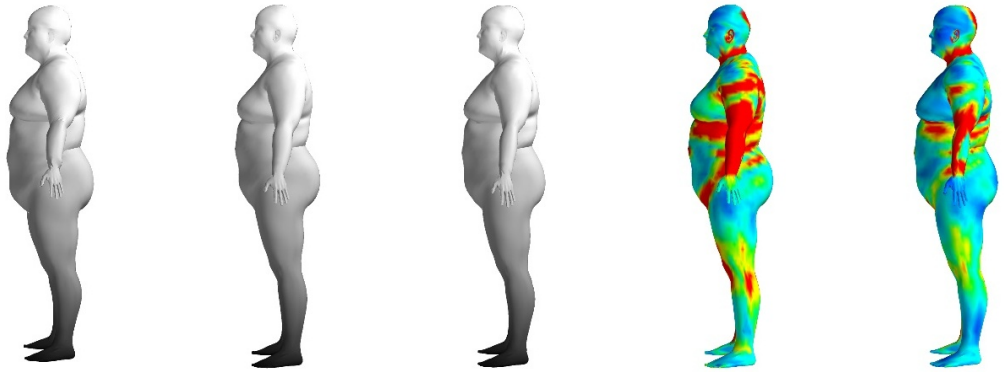
(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}



(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

Figure B.5: BMI reconstruction example.

(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}



(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

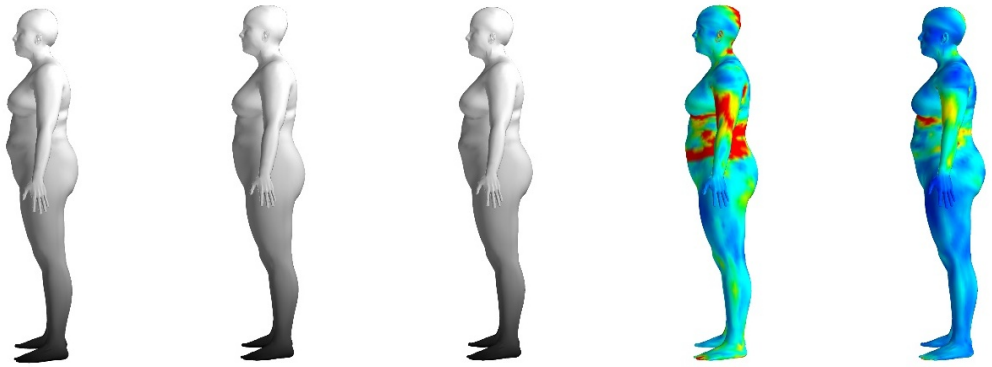
Figure B.6: BMI reconstruction example.

(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}

(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

Figure B.7: BMI reconstruction example.

(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}



(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

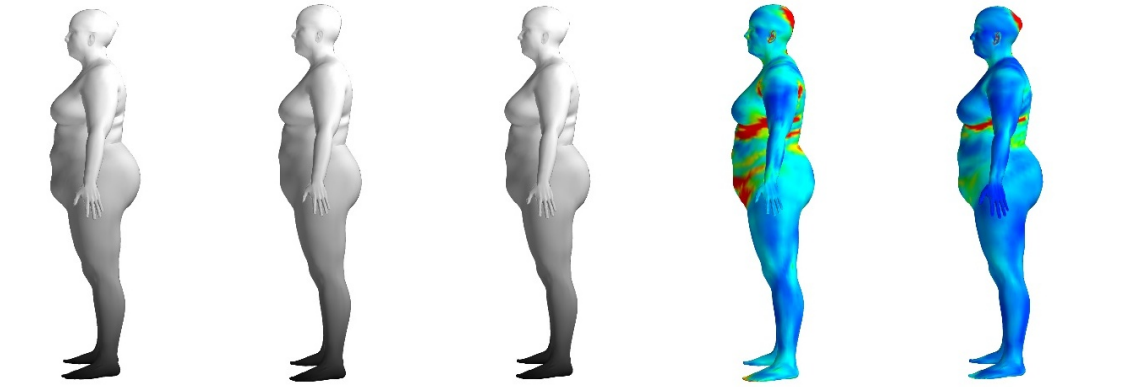
Figure B.8: BMI reconstruction example.

(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}

(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

Figure B.9: BMI reconstruction example.

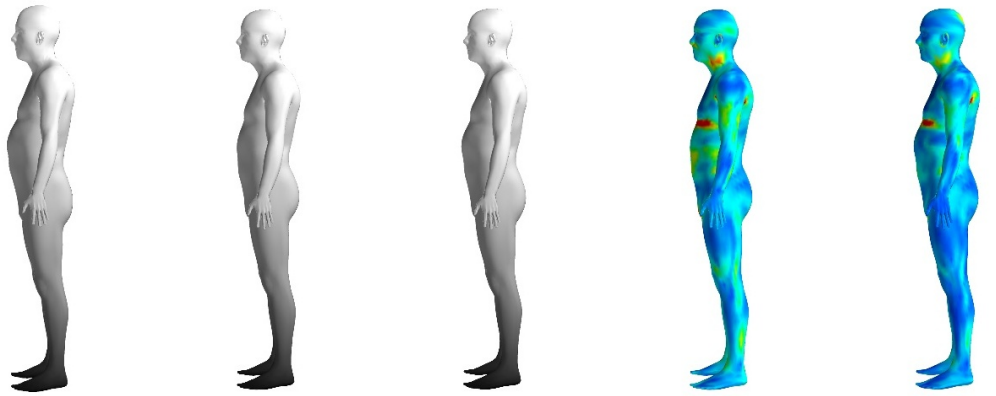
(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}



(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

Figure B.10: BMI reconstruction example.

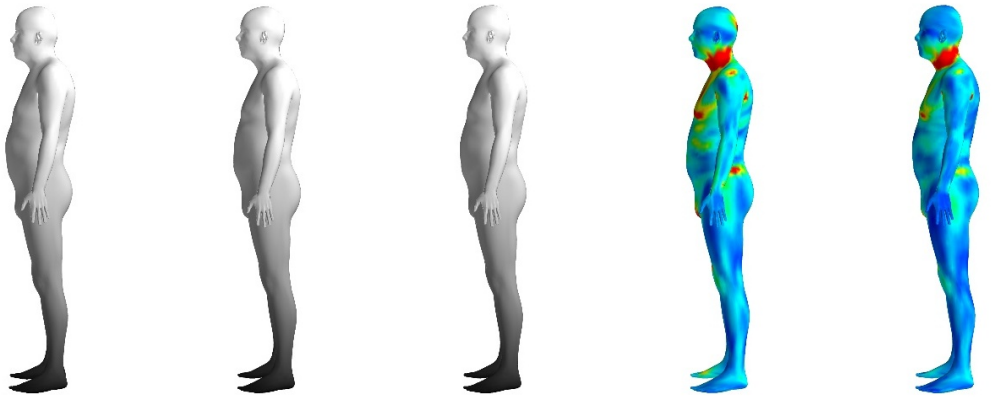
(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}



(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

Figure B.11: Gender reconstruction example.

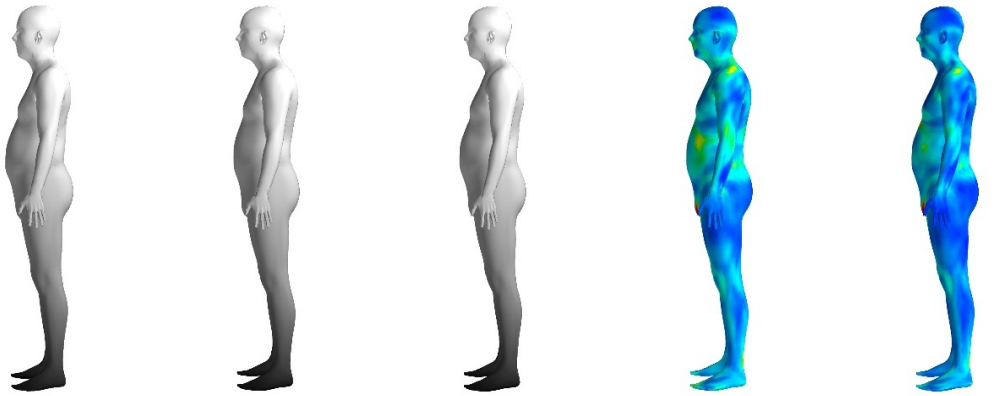
(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}



(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

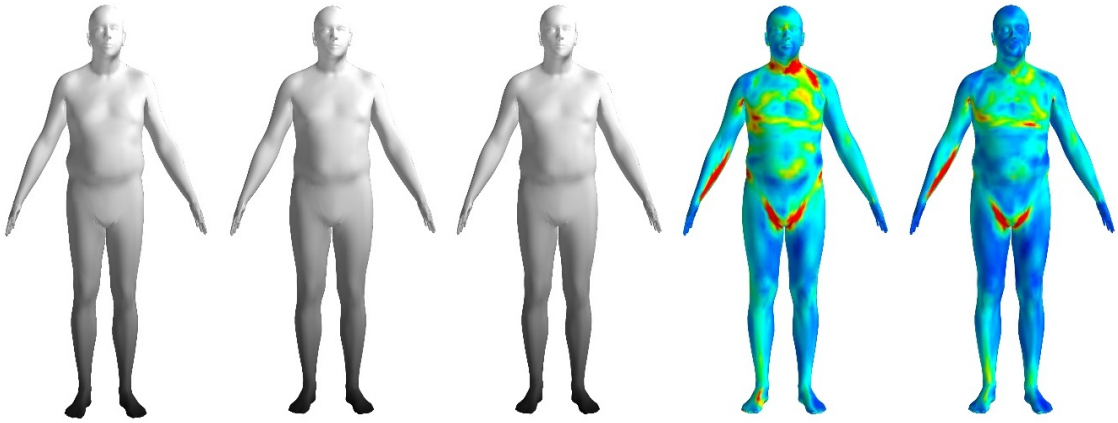
Figure B.12: Gender reconstruction example.

(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}

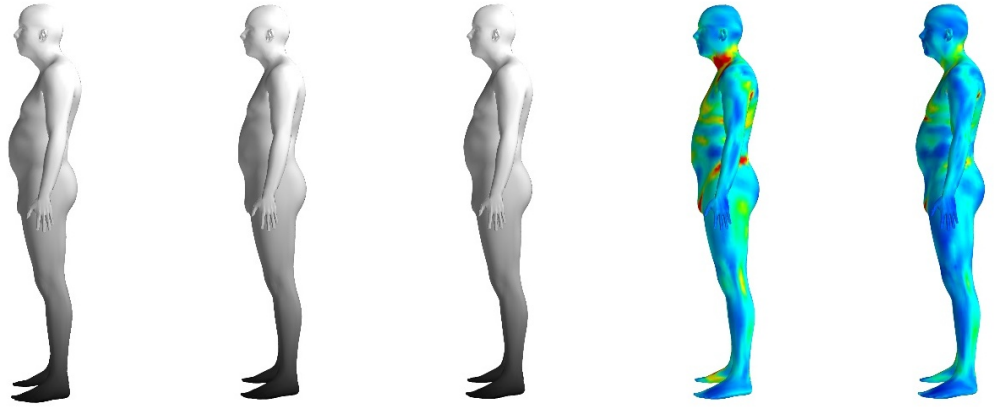


(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

Figure B.13: Gender reconstruction example.



(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}



(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

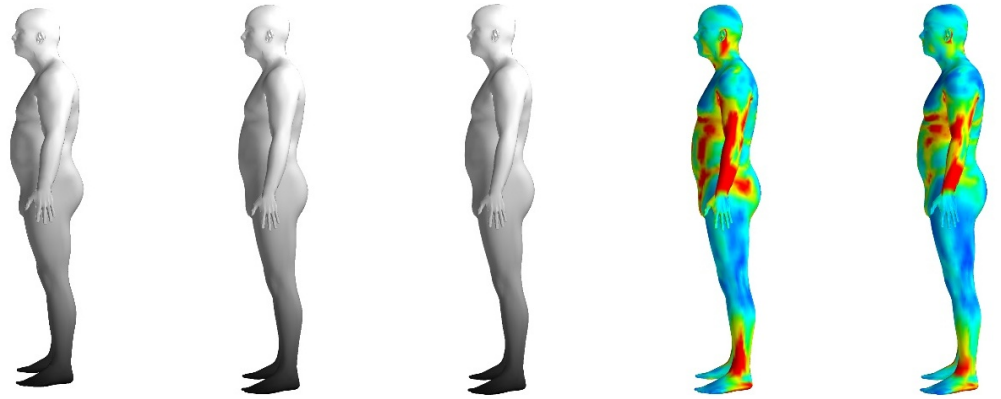
Figure B.14: Gender reconstruction example.

(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}

(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

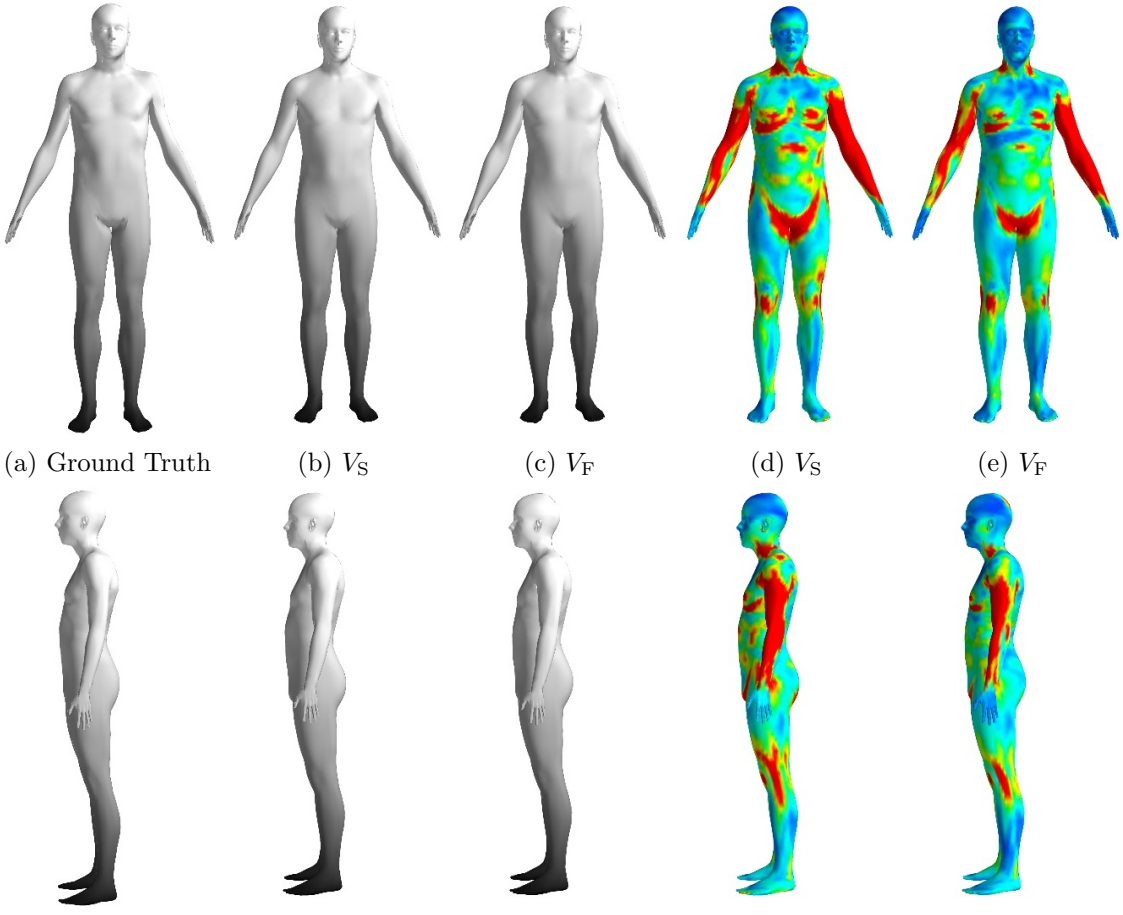
Figure B.15: Gender reconstruction example.

(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}



(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

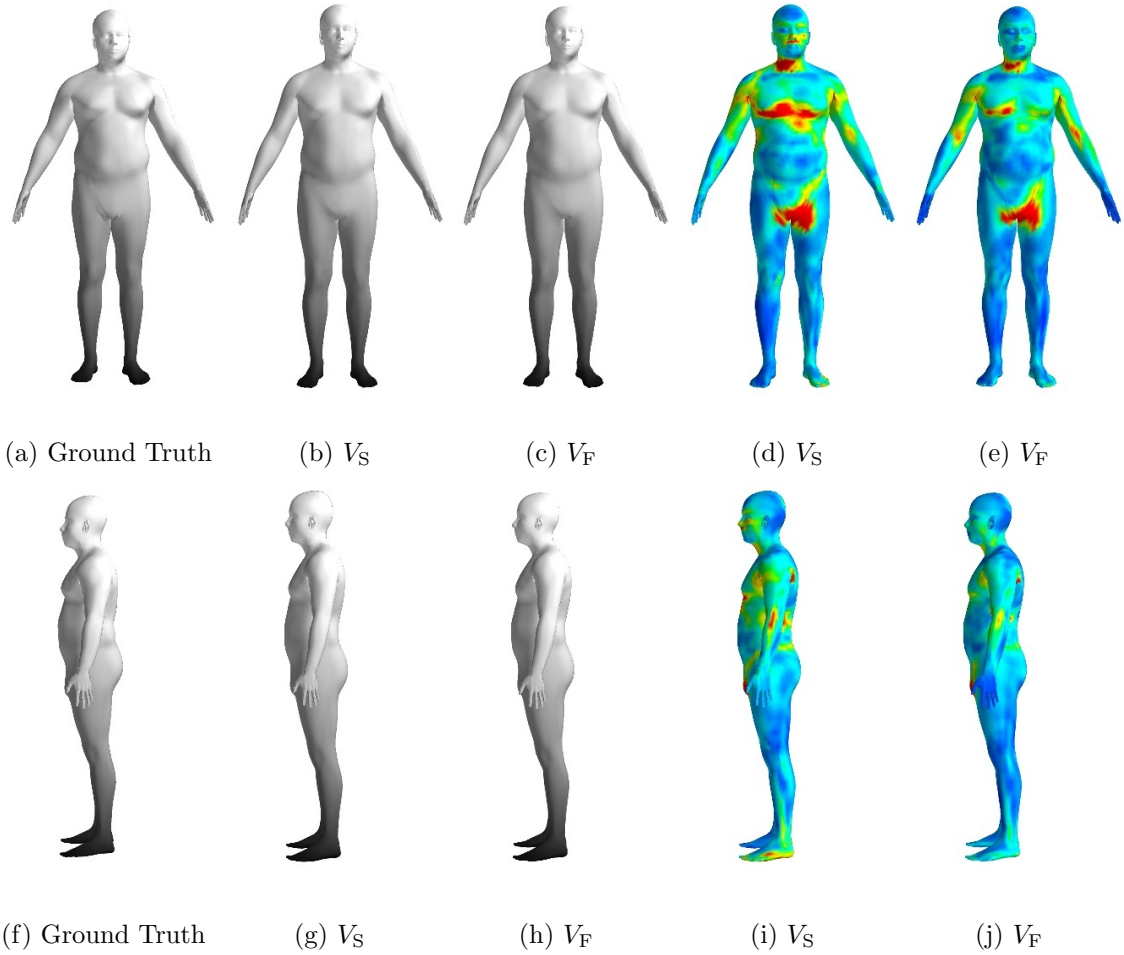
Figure B.16: Gender reconstruction example.



(a) Ground Truth (b) V_S (c) V_F (d) V_S (e) V_F

(f) Ground Truth (g) V_S (h) V_F (i) V_S (j) V_F

Figure B.17: Gender reconstruction example.

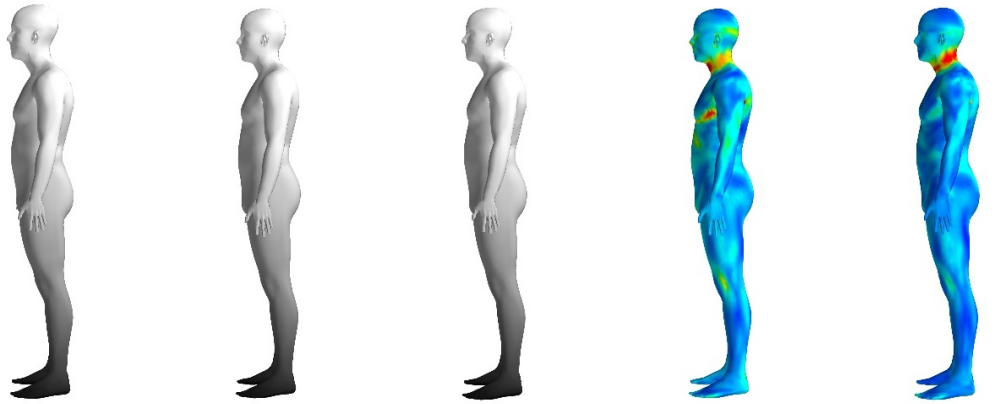


(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}

(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

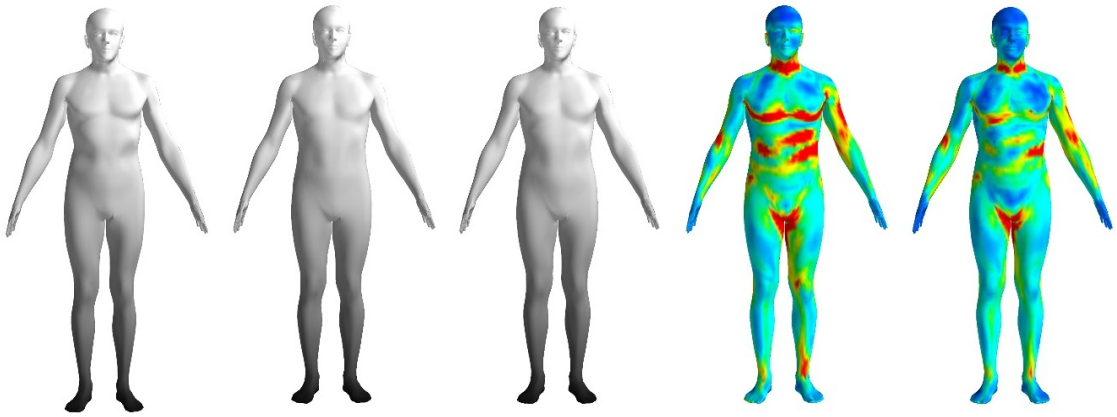
Figure B.18: Gender reconstruction example.

(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}



(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

Figure B.19: Gender reconstruction example.



(a) Ground Truth (b) V_{S} (c) V_{F} (d) V_{S} (e) V_{F}

(f) Ground Truth (g) V_{S} (h) V_{F} (i) V_{S} (j) V_{F}

Figure B.20: Gender reconstruction example.

Bibliography

[1] P.A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009. 11, 26, 95, 169, 173

[2] M. Alexa. Linear combination of transformations. In *ACM ToG*, volume 21, pages 380–387, 2002. 32, 119

[3] Amari and H. Nagaoka. *Methods of information geometry*. American Mathematical Society, 2000. 67

[4] M. Andriluka, S. Roth, and B. Schiele. Pictorial structures revisited: People detection and articulated pose estimation. In *CVPR*, pages 1014–1021. IEEE, 2009. 6, 81, 83, 85, 107

[5] E. Andruchow, G. Larotonda, L. Recht, and A. Varela. The left-invariant metric in the general linear group. *arXiv:1109.0520*, 2011. 49, 50

[6] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis. SCAPE: Shape completion and animation of people. *ACM ToG*, 24(3):408–416, 2005. 5, 6, 7, 81, 94, 95, 96, 113, 118, 120, 131, 133, 156, 193

[7] P.J. Basser, J. Mattiello, and D. LeBihan. MR diffusion tensor spectroscopy and imaging. *Biophysical J.*, 66(1):259–267, 1994. 10

[8] A. Baumberg and D. Hogg. Learning flexible models from image sequences. In *ECCV*, volume 1, pages 299–308, 1994. 84, 97

[9] E. Begelfor and M. Werman. Affine invariance revisited. In *CVPR*, volume 2, pages 2087–2094. IEEE, 2006. 11

[10] A. Bhattacharya and D.B. Dunson. Nonparametric bayesian density estimation on manifolds with applications to planar shapes. *Biometrika*, 97(4):851–865, 2010. 68, 162, 194

[11] E. Bienenstock, S. Geman, and D.F. Potter. Compositionality, MDL priors, and object recognition. *NIPS*, pages 838–844, 1997. 5

[12] M. Bray, P. Kohli, and P.H.S. Torr. PoseCut: Simultaneous segmentation and 3D pose estimation of humans using dynamic graph-cuts. In *ECCV*, pages 642–655, 2006. 85, 110

[13] A.M. Bronstein, M.M. Bronstein, and R. Kimmel. Generalized multidimensional scaling: a framework for isometry-invariant partial surface matching. *Proceedings of the National Academy of Sciences of the USA*, 103(5):1168–1172, 2006. 159

[14] A.O. Bălan. *Detailed Human Shape and Pose from Images*. PhD thesis, Brown University, Providence, Rhode Island, 2010. 118, 131, 133, 151

[15] A.O. Bălan, M.J. Black, H. Haussecker, and L. Sigal. Shining a light on human pose: On shadows, shading and the estimation of pose and shape. In *ICCV*, pages 1–8. IEEE, 2007. 81, 120

[16] A.O. Bălan, L. Sigal, M.J. Black, J.E. Davis, and H.W. Haussecker. Detailed human shape and pose from images. In *CVPR*, pages 1–8. IEEE, 2007. 6, 7, 81, 94, 110, 120, 128, 156

[17] H.E. Cetingul and R. Vidal. Intrinsic mean shift for clustering on stiefel and grassmann manifolds. In *CVPR*, pages 1896–1902. IEEE, 2009. 11

[18] T-J. Cham and J. Rehg. A multiple hypothesis approach to figure tracking. In *CVPR*, pages 239–245. IEEE, 1999. 6, 83

[19] I. Chao, U. Pinkall, P. Sanan, and P. Schröder. A simple geometric model for elastic deformations. *ACM ToG*, 29(4):38, 2010. 118, 119

[20] G.S. Chirikjian. Stochastic models, information theory, and Lie groups: Classical results and geometric methods (hardback) book.(series: Applied and numerical harmonic analysis, vol. 1). 2009. 195

[21] G.S. Chirikjian. *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*, volume 2. Birkhäuser Boston, 2011. 195

[22] T.F. Cootes, D. Cooper, C.J. Taylor, and J. Graham. Active shape models - their training and application. *CVIU*, 61(1):38–59, Jan 1995. 83, 84

[23] M. Covell. Eigen-points: control-point location using principal component analyses. In *Automatic Face and Gesture Recognition*, pages 122–127. IEEE, 1996. 84

[24] M. Covell and C. Bregler. Eigen-points [image matching]. In *ICIP*, volume 3, pages 471–474. IEEE, 1996. 84

[25] Y. Dai, J. Trumpf, H. Li, N. Barnes, and R. Hartley. Rotation averaging with application to camera-rig calibration. In *ACCV*, pages 335–346. Springer, 2010. 162

[26] H. Daumé III and D. Marcu. Domain adaptation for statistical classifiers. *JAIR*, 26(1):101–126, May 2006. 162

[27] M.P. Do Carmo. *Riemannian geometry*. Birkhäuser Boston, 1992. 29, 52, 62, 170

[28] A. Edelman, T.A. Arias, and S.T. Smith. The geometry of algorithms with orthogonality constraints. *SIMAX*, 20(2):303–353, 1998. 172, 173

[29] A. Faktor and M. Irani. Clustering by composition – unsupervised discovery of image categories. In *ECCV*, pages 474–487. Springer, 2012. 5

[30] P.F. Felzenszwalb and D.P. Huttenlocher. Pictorial structures for object recognition. *IJCV*, 61(1):55–79, 2005. 6, 83

[31] V. Ferrari, M. Marin-Jiminez, , and A. Zisserman. Progressive search space reduction for human pose estimation. In *CVPR*. IEEE, 2008. 83, 85

[32] M. Fischler and R. Elschlager. The representation and matching of pictorial structures. *IEEE Trans. on Computer*, 22(1):67–92, Jan 1973. 6, 77, 83

[33] R Fisher. Dispersion on a sphere. *Proc. R. Soc. A: Math. & Phy. Sci.*, 217(1130):295–305, 1953. 66, 162

[34] P.T. Fletcher. Geodesic regression and the theory of least squares on Riemannian manifolds. *IJCV*, pages 1–15, 2012. 162

[35] P.T. Fletcher and S. Joshi. Principal geodesic analysis on symmetric spaces: Statistics of diffusion tensors. *Computer Vision and Mathematical Methods in Medical and Biomedical Image Analysis*, pages 87–98, 2004. 119, 162

[36] P.T. Fletcher, S. Joshi, C. Lu, and S. Pizer. Gaussian distributions on Lie groups and their application to statistical shape analysis. In *Information Processing in Medical Imaging*, pages 450–462. Springer, 2003. 69, 73, 119, 162

[37] P.T. Fletcher, C. Lu, and S. Joshi. Statistics of shape via principal geodesic analysis on Lie groups. In *CVPR*, volume 1, pages I–95. IEEE, 2003. 69, 73, 119, 151, 162

[38] P.T. Fletcher, C. Lu, S.M. Pizer, and S. Joshi. Principal geodesic analysis for the study of nonlinear statistics of shape. *IEEE Transactions on Medical Imaging*, 23(8):995–1005, 2004. 69, 70, 71, 119, 162, 163

[39] P.T. Fletcher, S. Venkatasubramanian, and S. Joshi. The geometric median on Riemannian manifolds with application to robust atlas estimation. *NeuroImage*, 45(1):S143–S152, 2009. 162

[40] L. Freifeld. Personal communication. iv

[41] O. Freifeld and M.J. Black. Lie bodies: A manifold representation of 3D human shape. In A. Fitzgibbon et al. (Eds.), editor, *ECCV*, pages 1–14. Springer-Verlag, 2012. 10, 21, 193

[42] O. Freifeld, S. Hauberg, and M.J. Black. Riemannian covariance transport. In *Submitted*. IEEE, 2013. 21, 113, 161, 184, 187

[43] O. Freifeld, A. Weiss, S. Zuffi, and M.J. Black. Contour people: A parameterized model of 2D articulated human shape. In *CVPR*, pages 639–646. IEEE, 2010. 10, 21, 112, 192

[44] A. Frome, F. Sha, Y. Singer, and J. Malik. Learning globally-consistent local distance functions for shape-based image retrieval and classification. In *ICCV*, pages 1–8. IEEE, 2007. 12

[45] A. Frome, Y. Singer, and J. Malik. Image retrieval and classification using local distance functions. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *NIPS 19 (NIPS)*, pages 417–424, Cambridge, MA, 2007. MIT Press. 12

[46] D. Gavrila. A Bayesian, exemplar-based approach to hierarchical shape matching. *IEEE Transactions on PAMI*, 29:1408–1421, 2007. 85

[47] S. Geman, D.F. Potter, and Z. Chi. Composition systems. *Quarterly of Applied Mathematics*, 60(4):707–736, 2002. 5

[48] S. Ghosh, E. Sudderth, M. Loper, and M.J. Black. From deformations to parts: Motion-based segmentation of 3d objects. In *NIPS 25*, pages 2006–2014, 2012. 7, 113, 120, 133, 194

[49] A. Goh and R. Vidal. Clustering and dimensionality reduction on Riemannian manifolds. In *CVPR*, pages 1–7. IEEE, 2008. 162

[50] K. Grauman, G. Shakhnarovich, and T. Darrell. Inferring 3D structure with a statistical image-based shape model. In *ICCV*, pages 641–647. IEEE, 2003. 85

[51] U. Grenander. *General pattern theory: A mathematical study of regular structures*. Clarendon Press, 1993. 4, 10, 86, 113

[52] U. Grenander. *Elements of pattern theory*. Johns Hopkins University Press, 1996. 4, 10, 88, 90

[53] U. Grenander and M.I. Miller. Representations of knowledge in complex systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 549–603, 1994. 3, 10, 86, 90, 92

[54] U. Grenander and M.I. Miller. Computational anatomy: An emerging discipline. *Quarterly of applied mathematics*, 56(4):617–694, 1998. 119

[55] U. Grenander and M.I. Miller. *Pattern theory: from representation to inference*. Oxford University Press, USA, 2007. 4, 90

[56] U. Grenander, M.I. Miller, and A. Srivastava. Hilbert-schmidt lower bounds for estimators on matrix Lie groups for ATR. *IEEE Transactions on PAMI*, 20(8):790–802, 1998. 162

[57] P. Guan, O. Freifeld, and M.J. Black. A 2d human body model dressed in eigen clothing. In *ECCV*, pages 285–298. Springer, 2010. 110, 112, 192

[58] P. Guan, L. Reiss, D.A. Hirshberg, A. Weiss, and M.J. Black. Drape: Dressing any person. *ACM ToG*, 31(4):35, 2012. 7, 110, 120, 147, 193

[59] P. Guan, A. Weiss, A.O. Bălan, and M.J. Black. Estimating human shape and pose from a single image. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1381–1388. IEEE, 2009. 120, 156

[60] J. Ham, D. Lee, and L. Saul. Semisupervised alignment of manifolds. In *UAI*, volume 10, pages 120–127, 2005. 162

[61] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*, volume 2. Cambridge University Press, 2000. 54

[62] N. Hasler, H. Ackermann, B. Rosenhahn, T. Thormahlen, and H.P. Seidel. Multilinear pose and body shape estimation of dressed subjects from image sets. In *CVPR*, pages 1823–1830. IEEE, 2010. 7, 113, 118, 120, 133

[63] N. Hasler, C. Stoll, M. Sunkel, B. Rosenhahn, and H.P. Seidel. A statistical model of human pose and body shape. *Computer Graphics Forum*, 28(2):337–346, 2009. 6, 7, 118, 120, 133

[64] T. Hastie and R. Tibshirani. Discriminant adaptive nearest neighbor classification. *IEEE Transactions on PAMI*, 18(6):607–616, June 1996. 12

[65] S. Hauberg, O. Freifeld, and M.J. Black. A geometric take on metric learning. In *NIPS 25*, pages 2033–2041, 2012. 12, 197

[66] S. Hauberg, F. Lauze, and K.S. Pedersen. Unscented Kalman filtering on Riemannian manifolds. *JMIV*, pages 1–18, 2012. 162, 163, 174, 178

[67] S. Hauberg, S. Sommer, and K.S. Pedersen. Natural metrics and least-committed priors for articulated tracking. *IVC*, 30(6-7):453–461, 2012. 162

[68] J. He, L. Balzano, and A. Szlam. Incremental gradient on the grassmannian for online foreground and background separation in subsampled video. In *CVPR*, pages 1568–1575. IEEE, 2012. 11

[69] N.J. Higham. The scaling and squaring method for the matrix exponential revisited. *SIAM Journal of Matrix Analysis and Applications*, pages 1179–1193, 2005. 142

[70] J. Hinkle, P. Muralidharan, P.T. Fletcher, and S. Joshi. Polynomial regression on Riemannian manifolds. In *ECCV*, pages 1–14. Springer Berlin Heidelberg, 2012. 162

[71] G. E. Hinton. Using relaxation to find a puppet. In *The AISB Summer Conference*, pages 148–157, 1976. 6, 83

[72] D.A. Hirshberg, M. Loper, E. Rachlin, and M.J. Black. Coregistration: Simultaneous alignment and modeling of articulated 3d shape. In *ECCV*, pages 242–255. Springer Berlin Heidelberg, 2012. 6, 7, 120, 133, 159

[73] S. Huckemann, T. Hotz, and A. Munk. Intrinsic shape analysis: Geodesic PCA for Riemannian manifolds modulo isometric Lie group actions. *Statistica Sinica*, 20:1–100, 2010. 69, 162

[74] M. Irani and O. Boiman. Similarity by composition. In *NIPS*, page 177. The MIT Press, 2006. 5

[75] H. Jhuang, J. Gall, C. Schmid, and M.J. Black. Towards understanding action recognition. In *Submitted*, 2013. 80, 192

[76] S. X. Ju, M.J. Black, and Y. Yacoob. Cardboard people: A parameterized model of articulated motion. In *Int. Conf. Automatic Face and Gesture Recognition*, pages 38–44, 1996. 6, 83

[77] H. Karcher. Riemannian center of mass and mollifier smoothing. *Communications on pure and applied mathematics*, 30(5):509–541, 1977. 68

[78] L. Karlinsky and S. Ullman. Using linking features in learning non-parametric part models. In A. Fitzgibbon et al. (Eds.), editor, *ECCV*, pages 326–339. Springer-Verlag, 2012. 193

[79] Y. Ke and R. Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In *CVPR*, volume 2, pages II–506. IEEE, 2004. 9

[80] D.G. Kendall. Shape manifolds, procrustean metrics, and complex projective spaces. *Bulletin of the London Mathematical Society*, 16(2):81–121, 1984. 9, 119

[81] C. Kervrann and F. Heitz. A hierarchical markov modeling approach for the segmentation and tracking of deformable shapes. *Graphical Models and Image Processing*, 60(3):173 – 195, 1998. 85

[82] A. Kheyfets, W.A. Miller, and G.A. Newton. Schild's ladder parallel transport procedure for an arbitrary connection. *IJTP*, 39(12):2891–2898, 2000. 174

[83] M. Kilian, N.J. Mitra, and H. Pottmann. Geometric modeling in shape space. In *ACM ToG*, volume 26, page 64, 2007. 119, 147

[84] D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009. 78

[85] C. H. Lampert and O. Krömer. Weakly-paired maximum covariance analysis for multimodal dimensionality reduction and transfer learning. In *ECCV*, pages 566–579. Springer, 2010. 196

[86] X. Lan and D. Huttenlocher. Beyond trees: Common factor models for 2D human pose recovery. In *ICCV*, pages 470–477. IEEE, 2005. 6, 83

[87] G. Lebanon. *Riemannian geometry and statistical machine learning*. PhD thesis, Carnegie Mellon University, Language Technologies Institute, School of Computer Science, 2005. 67

[88] J.M. Lee. *Riemannian manifolds: an introduction to curvature*, volume 176. Springer, 1997. 29, 52, 62

[89] J.M. Lee. *Introduction to smooth manifolds*, volume 218. Springer Verlag, 2003. 33, 41, 52, 56, 62

[90] J.M. Lee. *Introduction to topological manifolds*, volume 202. Springer, second edition, 2010. 52, 53

[91] M. Leotta and J. Mundy. Vehicle surveillance with a generic, adaptive, 3-D vehicle model. *IEEE Transactions on PAMI*, (99):1–1, 2011. 7

[92] X. Li and J. Bilmes. A bayesian divergence prior for classifier adaptation. In *Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS-2007)*, March 2007. 162

[93] D. Lin, E. Grimson, and J.W. Fisher III. Learning visual flows: A Lie algebraic approach. In *CVPR*, pages 747–754. IEEE, 2009. 23

[94] D. Lin, E. Grimson, and J.W. Fisher III. Modeling and estimating persistent motion with geometric flows. In *CVPR*, pages 1–8. IEEE, 2010. 23

[95] Z. Lin and L.S. Davis. Shape-Based Human Detection and Segmentation via Hierarchical Part-Template Matching. *IEEE Transactions on PAMI*, 32(4):604–618, 2010. 85

[96] M. Lorenzi, N. Ayache, and X. Pennec. Schild's ladder for the parallel transport of deformations in time series of images. In *IPMI*, pages 463–474. Springer, 2011. 163, 174, 175

[97] M. Lorenzi and X. Pennec. Geodesics, parallel transport & one-parameter subgroups for diffeomorphic image registration. *IJCV*, pages 1–17, 2012. 163

[98] S. Lovett. *Differential geometry of manifolds*. AK Peters, 2010. 52

[99] Yui Man Lui. Advances in matrix manifolds for computer vision. *Journal of Image and Vision Computing*, 30(6):380–388, 2012. 162

[100] Ameesh Makadia and Kostas Daniilidis. Direct 3d-rotation estimation from spherical images via a generalized shift theorem. In *CVPR*, volume 2, pages II–217. IEEE, 2003. 9

[101] Tomasz Malisiewicz and Alexei A. Efros. Recognition by association via learning per-exemplar distances. In *CVPR*, pages 1–8. IEEE, 2008. 12

[102] K.V. Mardia and P.E. Jupp. *Directional statistics*, volume 494. Wiley, 2009. 162

[103] D. R. Martin, Charless C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on PAMI*, 26(5):530–549, 2004. 108

[104] E. G. Miller and Christophe Chefd'hotel. Practical non-parametric density estimation on a transformation group for vision. In *CVPR*, volume 2, pages II–114. IEEE, 2003. 67, 131, 162

[105] M.I. Miller, G.E. Christensen, Y. Amit, and U. Grenander. Mathematical textbook of deformable neuroanatomies. *Proceedings of the National Academy of Sciences*, 90(24):11944, 1993. 119

[106] C.W. Misner, K.S. Thorne, and J.A. Wheeler. *Gravitation*. W.H. Freeman, 1973. 174

[107] D. Mumford and A. Desolneux. *Pattern theory: the stochastic analysis of real-world signals*. AK Peters, 2010. 4, 113

[108] J.R. Munkres. *Topology*. Prentice-Hall, Inc. (Upper Saddle River, NJ), second edition, 2000. 53, 199

[109] M.K. Murray and J.W. Rice. *Differential geometry and statistics*, volume 48. Chapman & Hall/CRC, 1993. 67

[110] R.M. Murray, Z. Li, and S.S. Sastry. *A mathematical introduction to robotic manipulation*. CRC, 1994. 141

[111] L. Noakes. A global algorithm for geodesics. *JAMS-Series A*, 65(1):37–50, 1998. 174

[112] E.-J. Ong and S. Gong. A dynamic human model using hybrid 2D-3D representations in hierarchical pca space. In *BMVC*, pages 33–42, 1999. 84

[113] X. Pennec. Probabilities and statistics on Riemannian manifolds: Basic tools for geometric measurements. In *NSIP*, pages 194–198, 1999. 68, 151, 162, 182, 183, 184

[114] X. Pennec, P. Fillard, and N. Ayache. A riemannian framework for tensor computing. *IJCV*, 66(1):41–66, 2006. 162

[115] X. Pennec and M. Lorenzi. Which parallel transport for the statistical analysis of longitudinal deformations? In *Colloque GRETSI'11*, 2011. 162, 174, 175

[116] F. Porikli, O. Tuzel, and P. Meer. Covariance tracking using model update based on Lie algebra. In *CVPR*, volume 1, pages 728–735. IEEE, 2006. 10

[117] D.F. Potter. *Compositional pattern recognition*. PhD thesis, Brown University, 1999. 5

[118] D. Ramanan and S. Baker. Local distance functions: A taxonomy, new algorithms, and an evaluation. *IEEE Transactions on PAMI*, 33(4):794–806, 2011. 12

[119] D. Ramanan and D.A. Forsyth. Finding and tracking people from the bottom up. In *CVPR*, volume 2, pages 467–474. IEEE, 2003. 6, 83

[120] K. Robinette, S. Blackwell, H. Daanen, M. Boehmer, S. Fleming, T. Brill, D. Hoeferlin, and D. Burnsides. Civilian American and European Surface Anthropometry Resource (CAESAR) final report. Technical Report AFRL-HE-WP-TR-2002-0169, US Air Force Research Laboratory, 2002. 118, 152

[121] G. Rosman, A.M. Bronstein, M.M. Bronstein, and R. Kimmel. Articulated motion segmentation of point clouds by group-valued regularization. In *Eurographics Workshop on 3D Object Retrieval*, pages 77–84. The Eurographics Association, 2012. 10

[122] G. Rosman, M.M. Bronstein, A.M. Bronstein, A. Wolf, and R. Kimmel. Group-valued regularization framework for motion segmentation of dynamic non-rigid shapes. *Scale Space and Variational Methods in Computer Vision*, pages 725–736, 2012. 10

[123] G. Rosman, Y. Wang, X.C. Tai, R. Kimmel, and Bruckstein A.M. Fast regularization of matrix-valued images. In A. Fitzgibbon et al. (Eds.), editor, *ECCV*, pages 173–186. Springer-Verlag, 2012. 10

[124] C. Rother, V. Kolmogorov, and A. Blake. GrabCut: Interactive foreground extraction using iterated graph cuts. *ACM ToG*, 23:309–314, 2004. 81, 85, 107

[125] S. Roweis. EM algorithms for PCA and SPCA. *NIPS*, pages 626–632, 1998. 4

[126] H. Royden. *Real Analysis: 2d Ed.* Macmillan, 1968. 200

[127] P. Saisan, G. Doretto, Y.N. Wu, and S. Soatto. Dynamic texture recognition. In *CVPR*, volume 2, pages II–58. IEEE, 2001. 11

[128] M. Salzmann and P. Fua. Linear local models for monocular reconstruction of deformable surfaces. *IEEE Transactions on PAMI*, 33(5):931–944, May 2011. 7

[129] J. Schäfer and K. Strimmer. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *SAGMB*, 4(1), 2005. 162, 181

[130] M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. In *NIPS 16*, 2004. 12

[131] S. Shalev-Shwartz, Y. Singer, and A.Y. Ng. Online and batch learning of pseudometrics. In *ICML*, pages 94–101. ACM, 2004. 12

[132] K. Shoemake. Animating rotation with quaternion curves. *SIGGRAPH computer graphics*, 19(3):245–254, 1985. 159

[133] L. Sigal and M.J. Black. Predicting 3D people from 2D pictures. In *AMDO*, volume LNCS 4069, pages 185–195, 2006. 6, 83

[134] S. Soatto, G. Doretto, and Ying Nian Wu. Dynamic textures. In *ICCV*, volume 2, pages 439–446. IEEE, 2001. 11

[135] S. Sommer, F. Lauze, S. Hauberg, and M. Nielsen. Manifold valued statistics, exact principal geodesic analysis and the effect of linear approximations. *ECCV*, pages 43–56, 2010. 70, 162

[136] S. Sommer, F. Lauze, and M. Nielsen. The differential of the exponential map, jacobi fields and exact principal geodesic analysis. *arXiv preprint arXiv:1008.1902*, 2010. 70

[137] A. Srivastava, U. Grenander, G.R. Jensen, and M.I. Miller. Jump-diffusion markov processes on orthogonal groups for object pose estimation. *Journal of Statistical Planning and Inference*, 103(1-2):15–37, 2002. 162

[138] A. Srivastava, I. Jermyn, and S. Joshi. Riemannian analysis of probability density functions with applications in vision. In *CVPR*, pages 1–8. IEEE, 2007. 11

[139] A. Srivastava and E. Klassen. Monte carlo extrinsic estimators of manifold-valued parameters. *IEEE Transactions on Signal Processing*, 50(2):299–308, 2002. 162

[140] A. Srivastava, P. Turaga, and S. Kurtek. On advances in differential-geometric approaches for 2d and 3d shape analyses and activity recognition. *Journal of Image and Vision Computing*, 30(6):398–416, 2012. 162

[141] R. Subbarao and P. Meer. Nonlinear mean shift over riemannian manifolds. *IJCV*, 84(1):1–20, 2009. 162

[142] R.W. Sumner and J. Popović. Deformation transfer for triangle meshes. *ACM ToG*, 23(3):399–405, 2004. 94, 118, 120, 128, 131, 138

[143] A. Tsoli and M.J. Black. Shape-and pose-invariant correspondences using probabilistic geodesic surface embedding. In *German Conference on Pattern Recognition*, pages 256–265. Springer, 2011. 159

[144] P. Turaga, A. Veeraraghavan, and R. Chellappa. Statistical analysis on stiefel and grassmann manifolds with applications in computer vision. In *CVPR*, pages 1–8. IEEE, 2008. 11

[145] O. Tuzel, F. Porikli, and P. Meer. Region covariance: A fast descriptor for detection and classification. *ECCV*, pages 589–600, 2006. 10

[146] M. Vaillant, M. Miller, L. Younes, and A. Trouvé. Statistics on diffeomorphisms via tangent space representations. *NeuroImage*, 23:161–169, 2004. 119, 162

[147] D. Wei, D. Lin, and J.W. Fisher III. Learning deformations with parallel transport. In *ECCV*, volume II, pages 287–300, 2012. 23, 163, 164, 195

[148] K.Q. Weinberger and L.K. Saul. Distance metric learning for large margin nearest neighbor classification. *JMLR*, 10:207–244, 2009. 12

[149] A. Weiss, D.A. Hirshberg, and M.J. Black. Home 3D body scans from noisy image and range data. In *ICCV*, pages 1951–1958. IEEE, November 2011. 7, 120, 156

[150] K.C. Wolfe, M. Mashner, and G.S. Chirikjian. Bayesian fusion on lie groups. *Journal of Algebraic Statistics*, 2(1):75–97, 2011. 162

[151] E.P. Xing, A.Y. Ng, M.I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. *NIPS*, 15:505–512, 2002. 12

[152] Y. Ying and P. Li. Distance metric learning with eigenvalue optimization. *JMLR*, 13:1–26, 2012. 12

[153] W. Zhang. *Statistical inference and probabilistic modeling in compositional vision*. PhD thesis, Brown University, 2009. 5

[154] S. Zuffi and M.J. Black. Puppet flow. Technical report, 2013. 80, 192

[155] S. Zuffi, O. Freifeld, and M.J. Black. From pictorial structures to deformable structures. In *CVPR*, pages 3546–3553. IEEE, 2012. 112, 192