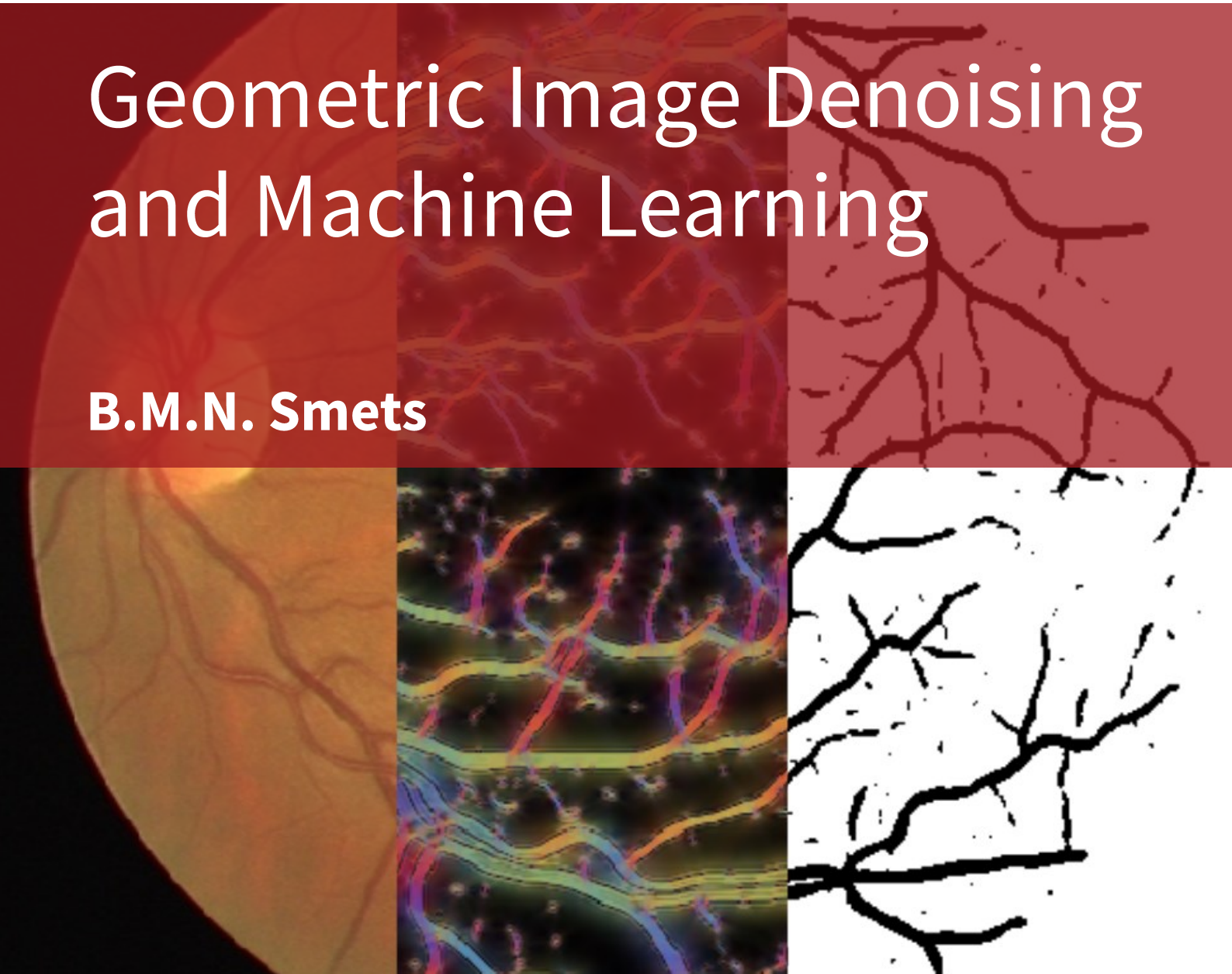


Geometric Image Denoising and Machine Learning

B.M.N. Smets



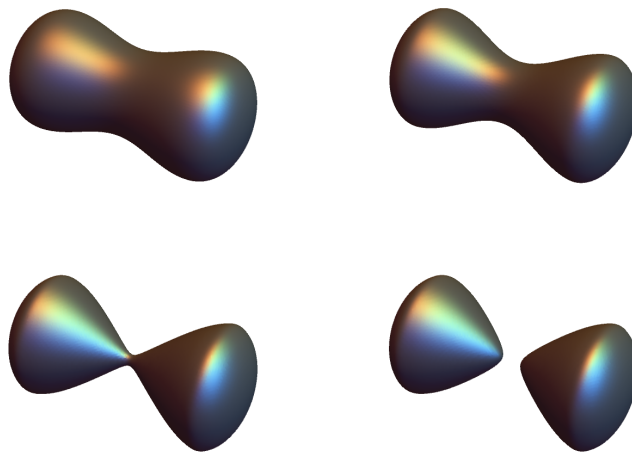
Master Thesis / 2019

Industrial and Applied Mathematics

Geometric Image Denoising and Machine Learning

Master Thesis

B.M.N. Smets



Supervisors:

dr.ir. R. Duits
dr.ir. E.J. Bekkers
dr.ir. J.W. Portegies

June 9, 2019

Contents

Introduction	1
1 Geometric Theory	3
1.1 The Space of 2D Positions and Orientations $\mathbb{M}_2$	3
1.1.1 Construction of the Manifold	3
1.1.2 Construction of the Tangent Bundle	6
1.1.3 Left Invariance	9
1.1.4 Non-Commutativity and Lie Brackets	10
1.1.5 Metric Tensor, Gradient & Divergence	13
1.1.6 Domains and Boundaries	16
1.1.7 Group Distance and Estimates	17
1.2 Gauge Frames	18
1.2.1 Exponential Curves	18
1.2.2 Alignment	19
1.2.3 Metric, Gradient and Divergence	21
1.3 Orientation Scores	21
1.3.1 Invertible Transformation	22
1.3.2 Visualization	22
1.3.3 Orientation Confidence & Isotropy Factor	23
2 Image Denoising	25
2.1 Introduction	25
2.2 Coherence Enhancing Diffusion	25
2.3 Total Variation Flow	26
2.3.1 Weak Formulation	26
2.3.2 Strong Formulation	27
2.3.3 Enhancement PDE	30
2.3.4 Isotropy Controlled Flow	31
2.4 Mean Curvature Flow	32
2.4.1 Mean Curvature	32
2.4.2 Enhancement PDE	32

2.5	Implementation Notes	33
2.5.1	Numerical Stability	33
2.5.2	Timescales & Total Variation Scaling	34
2.6	Experiments	34
2.6.1	Isotropic without Coherence Enhancement	34
2.6.2	Isotropic with Coherence Enhancement	38
2.6.3	Anisotropic	39
2.6.4	Post-processing	41
2.6.5	3-Stage TVF Method	42
3	Machine Learning	44
3.1	Spatial Convolution Neural Network	44
3.1.1	Layered Computation	44
3.1.2	Loss & Optimization	45
3.2	Designed Equivariance	47
3.3	Group Convolution Neural Network	49
3.3.1	Lifting	49
3.3.2	Group Convolution	50
3.3.3	Projection	51
3.4	DRIVE Segmentation	55
3.4.1	Networks	55
3.4.2	Results	55
	Appendices	59
	Appendix A PDE Guided CNN Design	59
	Bibliography	64

Introduction

Problem & Approach

Many modern image analysis techniques take advantage of lifting an image to the space of positions and orientations. The idea is to assign to each point of the image not simply a single intensity (or 3 intensities in case of colors) but an intensity for each orientation, these intensities then tell us how strongly aligned the image is to that particular orientation at that point. Decomposing an image like this (via the *orientation score* transform) gives us a tool to process it in an orientation aware manner. The general image processing pipeline this introduces is illustrated in figure 1, compare the blurriness of the crossings of the spatial method versus the orientation score method.

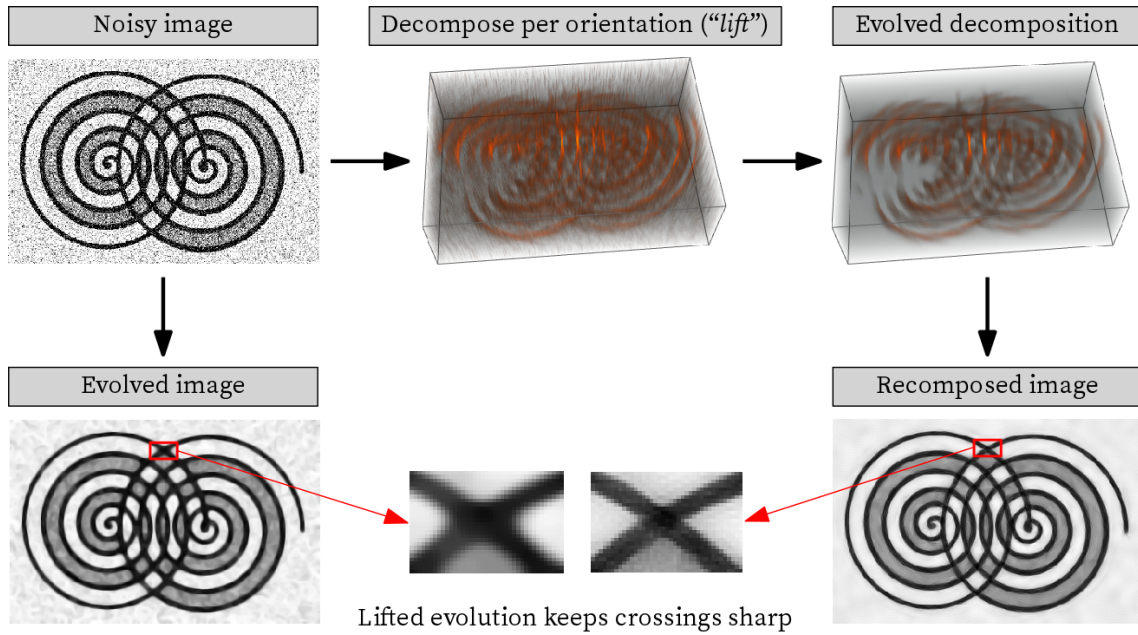


Figure 1: Preserving the structure of crossings during denoising on *Orientation Scores* (OS).

Restricting ourselves to 2D images we start by introducing the homogeneous space of 2D positions and orientations, we develop a set of geometric tools on this space that take into account the natural interaction between and composition of translations and rotations.

This higher level geometry can then serve as the backdrop on which we can formulate PDE methods. When doing image processing it is desirable, if not required, that the methods used are equivariant under rotation and translation, i.e. the output should rotate/translate together with the input. In a Euclidian setting special care needs to be taken when formulating PDEs that they are roto-translation equivariant, on the space of positions and orientations this property comes much more easily.

Describing image processing techniques with PDEs has the benefit of geometric interpretability. An important modern image processing tool that lacks a general interpretation is the *Convolutional Neural Networks* (CNN), being able to describe CNNs in terms of PDEs would go some way in alleviating this deficiency¹ by giving a concise high level mathematical description of the action of the network, we will give some initial results that contribute to such a PDE framework. This would also open up such networks to existing mathematical analysis techniques.

Conventional CNNs are translation equivariant but not, by themselves, rotation invariant. In tasks such as classifying objects this is an obvious shortcoming as figure 2 illustrates. In current practice this is dealt with by data augmentation: the network is trained multiple times on instances of the same image rotated under various angles. This data augmentation is inefficient and extends the training time by an order of magnitude.

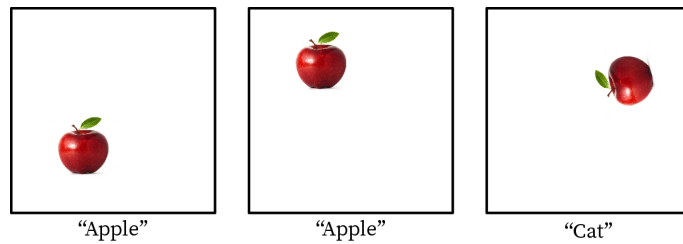


Figure 2: Classifying objects: current CNNs are translation invariant but not rotation invariant without data augmentation, this can lead to situations where an object is labeled differently depending on its orientation.

But we can wonder if the structure of a CNN makes it translation equivariant is it not possible to design it in such a way that it is also rotation equivariant from the start? The answer is yes, by giving it the same structure as we did our denoising methods: lift the image to the space of positions and orientations.

Research Context

Our work in geometric denoising of 2D images complements work done by our supervisors in contour enhancement [2], crossing-preserving filtering [3] and tractography [4], all of which focus on the enhancement of 2D and 3D medical imaging types such as X-Ray and MRI. In each of these PDEs are formulated on the space of positions and orientations in order to disentangle crossings.

In a more general setting similar work is being done in studying mathematical techniques for denoising and enhancement using the same tools as we will use such as total variation flow [5], mean curvature flow [6, 7] and other geometric models that deconstruct images based on orientation [8].

With respect to machine learning a similar trend of introducing non-Euclidean geometry is happening both in Eindhoven [9] and abroad [10, 11, 12] with the aim of endowing neural networks with more structural properties such as rotation equivariance without data augmentation.

The correspondence between PDEs and convolutional neural networks has also been previously noted and used to solve PDEs [13]. However to our knowledge has not been used the other way around: letting PDEs inspire network design.

We will give a modest first step in applying this idea in Chapter 3 by introducing morphological convolution to the last layer. An overview of how a PDE can dictate CNN design (and how morphological convolutions arise) is given in appendix A. This idea, while still being fleshed out, has yielded positive results in our first experiments, giving us reason to pursue it further.

¹See [1] for a general critique on black box machine learning models and why interpretability matters.

Chapter 1

Geometric Theory

1.1 The Space of 2D Positions and Orientations $\mathbb{M}_2$

A roto-translation in the plane is a translation followed by a rotation, which we can represent by elements of $\mathbb{R}^2 \times S^1$. Two roto-translations performed one after another result in a net roto-translation, so these operations form a group. And while translations commute and rotations commute, roto-translations do not, this composition and lack of commutativity is illustrated in figure 1.1.

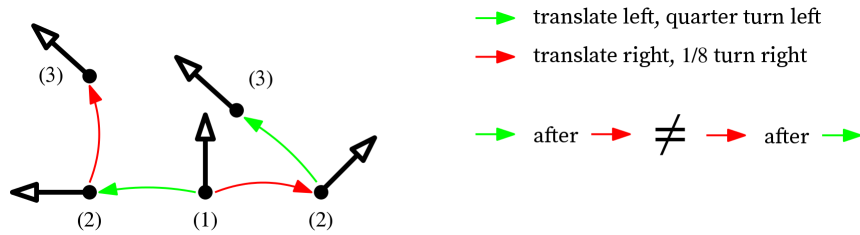


Figure 1.1: Roto-translations can be composed to form a non-commutative group.

Consequently we do not want to use $\mathbb{R}^2 \times S^1$ as a direct product group, but specify an indirect product.

1.1.1 Construction of the Manifold

First we make rotations in the plane concrete.

Definition 1.1 (Rotation Operator). Let $s \in S^1$ parameterized by angle θ , then $\tilde{R}_s : \mathbb{R}^2 \rightarrow \mathbb{R}^2$ gives a rotation of the plane by acting on the orthonormal basis $(\mathbf{e}_1, \mathbf{e}_2)$:

$$\tilde{R}_s \mathbf{v} := R_\theta \mathbf{v} := R_\theta (v^1 \mathbf{e}_1 + v^2 \mathbf{e}_2) := (\cos \theta v^1 - \sin \theta v^2) \mathbf{e}_1 + (\sin \theta v^1 + \cos \theta v^2) \mathbf{e}_2.$$

Remark 1.2 (Notation). We will be using the convention where covariant components are indexed using subscripts and contravariant are indexed with superscripts (like v^1 and v^2 above), when there is possible confusion with powers we will use extra brackets, for example $(v^i)^j$ (where the inner superscript is an index and the outer superscript a power).

Now we can specify exactly how roto-translations are composed and define the space of positions and orientations by representing elements of S^1 with their respective rotation operators.

Definition 1.3 (The 2D space of positions and orientations).

$$\mathbb{M}_2 := \mathbb{R}^2 \rtimes S^1 := (\mathbb{R}^2 \times S^1, \odot).$$

Where $\odot$ denotes the composition of roto-translations. Let $(\mathbf{x}_1, R_1), (\mathbf{x}_2, R_2) \in \mathbb{R}^2 \times S^1$:

$$(\mathbf{x}_1, R_1) \odot (\mathbf{x}_2, R_2) := (\mathbf{x}_1 + R_1 \mathbf{x}_2, R_1 R_2).$$

We will represent elements of $\mathbb{M}_2$ in the following manner:

$$(\mathbf{x}, \mathbf{n}) \in \mathbb{R}^2 \times \{\mathbf{n} \in \mathbb{R}^2 \mid \|\mathbf{n}\| = 1\}. \quad (1.1)$$

Where $\mathbf{x} = x^1 \mathbf{e}_1 + x^2 \mathbf{e}_2$ straightforwardly represents the translation and $\mathbf{n} = n^1 \mathbf{e}_1 + n^2 \mathbf{e}_2$ is the result of the rotation acting on a reference vector, in our case we use $\mathbf{e}_1$. Put another way: if we have a rotation operator R_s associated with some $s \in S^1$ then $\mathbf{n}_s = R_s \mathbf{e}_1$. Since there is no ambiguity we will also denote this rotation operator as $R_{\mathbf{n}}$, so that $R_{\mathbf{n}} \mathbf{e}_1 = \mathbf{n}$. The neutral element is then written as $I = (\mathbf{0}, \mathbf{e}_1)$.

The following definitions give some convenient shorthands.

Definition 1.4 (Orientation vector shorthands). Let $\mathbf{n} = n^1 \mathbf{e}_1 + n^2 \mathbf{e}_2$ then

$$\begin{aligned} \bar{\mathbf{n}} &:= R_{\mathbf{n}} \mathbf{e}_2 = -n^2 \mathbf{e}_1 + n^1 \mathbf{e}_2, \\ \mathbf{n}^{-1} &:= R_{\mathbf{n}}^{-1} \mathbf{e}_1 = n^1 \mathbf{e}_1 - n^2 \mathbf{e}_2. \end{aligned}$$

Which are just the vectors rotated $\frac{\pi}{2}$ counter-clockwise resp. mirrored over the reference spatial axis as illustrated in figure 1.2

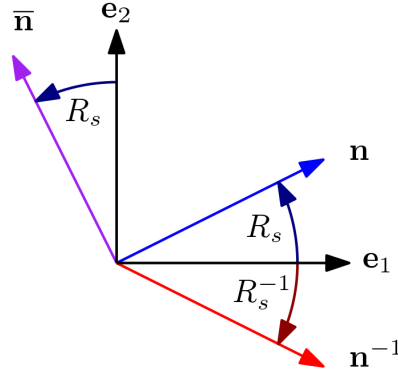


Figure 1.2: Relation between rotation vectors and operators.

Under this representation the group operation is:

Definition 1.5 (Group operation).

$$(\mathbf{x}, \mathbf{n}) \odot (\mathbf{y}, \mathbf{m}) := (\mathbf{x} + R_{\mathbf{n}} \mathbf{y}, R_{\mathbf{n}} \mathbf{m}).$$

Note that this product is not commutative and that $(\mathbf{x}, \mathbf{n})$ is applied *before* $(\mathbf{y}, \mathbf{m})$ is. In the same vein the inverse of a group element is given by:

Definition 1.6 (Group inverse).

$$(\mathbf{x}, \mathbf{n})^{-1} := (-R_{\mathbf{n}}^{-1} \mathbf{x}, n^1 \mathbf{e}_1 - n^2 \mathbf{e}_2) = (-R_{\mathbf{n}^{-1}} \mathbf{x}, \mathbf{n}^{-1}).$$

For the purpose of computation it's often helpful to represent the rotation operator in the familiar matrix fashion:

$$R_{\mathbf{n}} = \begin{pmatrix} n^1 & -n^2 \\ n^2 & n^1 \end{pmatrix}. \quad (1.2)$$

In fact the entire group element can be represented in matrix form:

$$(\mathbf{x}, \mathbf{n}) = \begin{pmatrix} n^1 & -n^2 & x^1 \\ n^2 & n^1 & x^2 \\ 0 & 0 & 1 \end{pmatrix}, \quad (1.3)$$

where now the usual matrix multiplication handles the group operation with $A \odot B = AB$.

Some care now needs to be taken with choosing charts for $\mathbb{M}_2$ since it is not homeomorphic to $\mathbb{R}^3$. Consider the following atlas of $\mathbb{M}_2$:

Definition 1.7 (Helical atlas of $\mathbb{M}_2$).

$$\begin{aligned} \Phi &= \{ \phi_z \mid z \in \mathbb{Z} \}, \\ \phi_z &: \{ (\mathbf{x}, \mathbf{n}) \in \mathbb{M}_2 \mid n^1 \neq 2(z \bmod 2) - 1 \} \rightarrow \mathbb{R}^2 \times (\pi(z-1), \pi(z+1)), \\ \phi_z((\mathbf{x}, \mathbf{n})) &= \begin{pmatrix} x^1 \\ x^2 \\ \pi z + \text{Arg}((-1)^z(n^1 + in^2)) \end{pmatrix}, \\ \phi_z^{-1}(x, y, \theta) &= (x\mathbf{e}_1 + y\mathbf{e}_2, \cos \theta \mathbf{e}_1 + \sin \theta \mathbf{e}_2). \end{aligned}$$

With this atlas we can establish the smoothness of $\mathbb{M}_2$.

Proposition 1.8. $\mathbb{M}_2$ is a smooth manifold.

Proof. We can check the transition functions¹ to confirm the compatibility of the charts, let $z_1, z_2 \in \mathbb{Z}$, then:

$$\phi_{z_1} \circ \phi_{z_2}^{-1} \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} = \phi_{z_1}(x\mathbf{e}_1 + y\mathbf{e}_2, \cos \theta \mathbf{e}_1 + \sin \theta \mathbf{e}_2) = \begin{pmatrix} x \\ y \\ \theta + \pi(z_1 - z_2 + Z_{z_1, z_2}(\theta)) \end{pmatrix},$$

where

$$Z_{z_1, z_2}(\theta) := \begin{cases} 0 & \text{if } z_1 - z_2 \text{ is even,} \\ 2 \lfloor \{ \theta < \pi z_2 \} \rfloor - 1 & \text{else.} \end{cases}$$

In any case the spatial components transform as the identity. For the orientation component we need to distinguish between even and odd charts. When $z_1 - z_2$ is even we are either transitioning between two even charts or two odd charts, in either case the domain of the charts is the same and we simply add a constant multiple of 2π to the orientation coordinate, clearly this is a smooth transform.

When we are transitioning between even and odd charts then we have to take into account the non-identical domains. The intersection of the domains of an even and odd map is those $\mathbf{n}$ for which $n_2 \neq 0$. In the coordinates of ϕ_{z_2} this equates to $\theta \neq \pi z_2$, which is exactly where the discontinuity of Z_{z_1, z_2} is, everywhere else it is a constant function. Hence on the two open non-connected subdomains $\theta < \pi z_2$ and $\theta > \pi z_2$ the transition function is the identity plus a constant factor depending on the subdomain and as such is smooth.

We can conclude that any two charts are C^∞ -compatible and so $\mathbb{M}_2$ is a smooth manifold. ■

¹Checking the transition function between a single even and odd chart would suffice here since they cover the manifold, but sometimes you finish climbing over the mountain before someone points out the tunnel.

While we only need two charts (an even and an odd one) to cover the whole manifold we can still use the whole atlas $\Phi = \{\phi_z \mid z \in \mathbb{Z}\}$ since it will allow us to use the entirety of $\mathbb{R}^3$ as parameter space. This will allow us to map any smooth curve in $\mathbb{M}_2$ to a smooth curve in $\mathbb{R}^3$ and vice versa, something which would not be possible in any subset of the atlas because of jumps caused by the periodic nature of S^1 . To see why we refer to this atlas as helical take a look at figure 1.3.

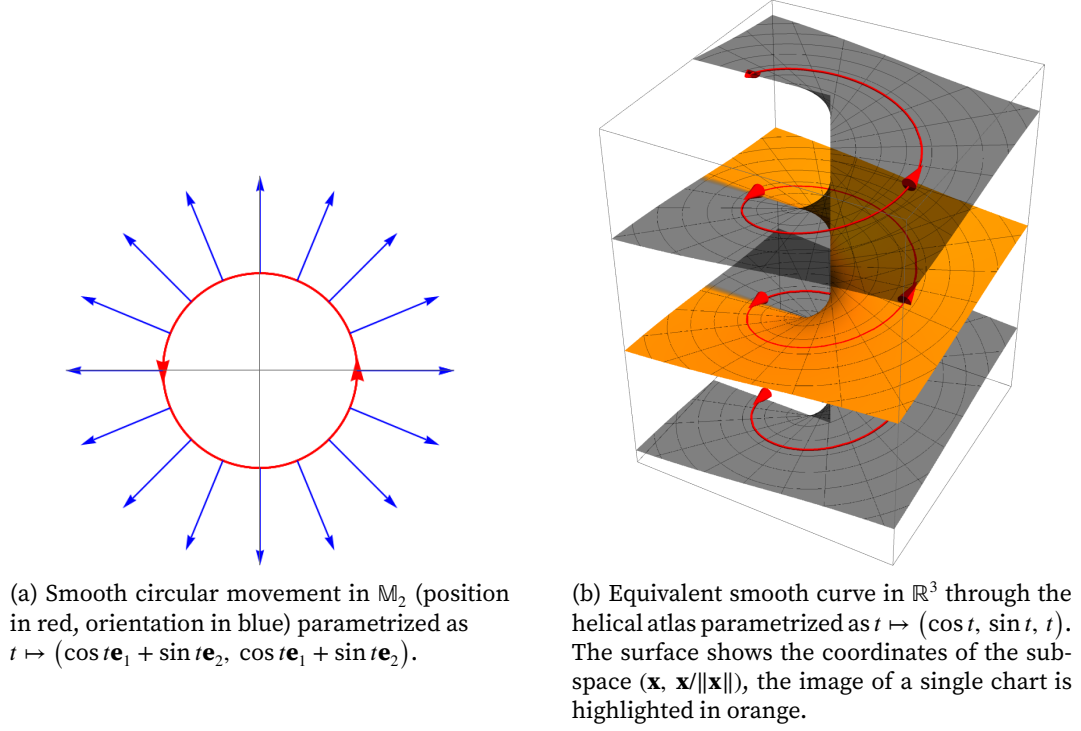


Figure 1.3: Illustrating the structure of the helical atlas.

1.1.2 Construction of the Tangent Bundle

Let's now consider how to interpret movement inside $\mathbb{M}_2$, given we are at a particular point $(\mathbf{x}, \mathbf{n})$ we have three distinct ways of changing our position. Firstly we can keep our orientation $\mathbf{n}$ constant but move along it (forward or backward), secondly we can move orthogonal to $\mathbf{n}$, sidestepping to the left or right with respect to our current orientation. Lastly we can stay in place and turn around. These three movements are illustrated in figure 1.4, we will refer to them as longitudinal, lateral and rotational movement respectively. While this choice of movements is somewhat arbitrary from a mathematical standpoint and the following derivations works equally well with different (orthogonal) choices the reader will agree that intuitively these three are a straightforward choice.

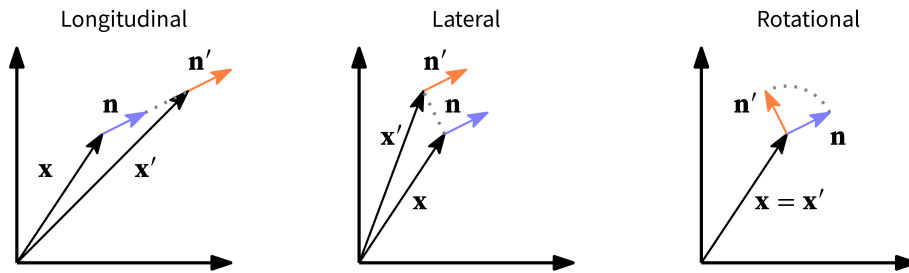


Figure 1.4: Types of movements in $\mathbb{M}_2$

We can represent these movements through $g = (\mathbf{x}, \mathbf{n}) = (x^1 \mathbf{e}_1 + x^2 \mathbf{e}_2, n^1 \mathbf{e}_1 + n^2 \mathbf{e}_2)$ by the following smooth $\mathbb{R} \rightarrow \mathbb{M}_2$ curves:

$$\begin{aligned}
\gamma_{1,g}(t) &= (\mathbf{x}, \mathbf{n}) \odot (t\mathbf{e}_1, \mathbf{e}_1) \\
&= (\mathbf{x} + R_{\mathbf{n}} t \mathbf{e}_1, R_{\mathbf{n}} \mathbf{e}_1) \\
&= (\mathbf{x} + t\mathbf{n}, \mathbf{n}), \\
\gamma_{2,g}(t) &= (\mathbf{x}, \mathbf{n}) \odot (t\mathbf{e}_2, \mathbf{e}_1) \\
&= (\mathbf{x} + R_{\mathbf{n}} t \mathbf{e}_2, R_{\mathbf{n}} \mathbf{e}_1) \\
&= (\mathbf{x} + t\bar{\mathbf{n}}, \mathbf{n}), \\
\gamma_{3,g}(t) &= (\mathbf{x}, \mathbf{n}) \odot (\mathbf{0}, \cos t \mathbf{e}_1 + \sin t \mathbf{e}_2) \\
&= (\mathbf{x} + R_{\mathbf{n}} \mathbf{0}, \cos t R_{\mathbf{n}} \mathbf{e}_1 + \sin t R_{\mathbf{n}} \mathbf{e}_2) \\
&= (\mathbf{x}, \cos t \mathbf{n} + \sin t \bar{\mathbf{n}}).
\end{aligned} \tag{1.4}$$

For all curves we have $\gamma_{1,g}(0) = \gamma_{2,g}(0) = \gamma_{3,g}(0) = g = (\mathbf{x}, \mathbf{n})$. Using these we can now define the following linear operators on some function $f \in C^1(\mathbb{M}_2, \mathbb{R})$:

Definition 1.9 (Left invariant differential operators).

$$\mathcal{A}_i|_g f := (f \circ \gamma_{i,g})'(0) \quad \forall i \in \{1, 2, 3\}.$$

The reason we call these operator *left-invariant* will be become clear in the next section.

These operators obey the *Leibniz's rule* as we would expect calling them differential operators.

Lemma 1.10 (Leibniz's rule). *Let $f_1, f_2 \in C^1(\mathbb{M}_2, \mathbb{R})$ then for $i \in \{1, 2, 3\}$:*

$$\mathcal{A}_i(f_1 f_2) = f_1 \mathcal{A}_i f_2 + f_2 \mathcal{A}_i f_1.$$

Proof. Let $g \in \mathbb{M}_2$, then:

$$\begin{aligned}
\mathcal{A}_i|_g(f_1 f_2) &= ((f_1 f_2) \circ \gamma_{i,g})'(0) \\
&= (f_1 \circ \gamma_{i,g} \cdot f_2 \circ \gamma_{i,g})'(0) \\
&= f_1 \circ \gamma_{i,g}(0) \cdot (f_2 \circ \gamma_{i,g})'(0) + f_2 \circ \gamma_{i,g}(0) \cdot (f_1 \circ \gamma_{i,g})'(0) \\
&= f_1(g) \cdot \mathcal{A}_i|_g f_2 + f_2(g) \cdot \mathcal{A}_i|_g f_1.
\end{aligned}$$

■

To make these operators more concrete we need to select a chart $\phi \in \Phi$, so that there exists $t_0 > 0$ for all $|t| < t_0 : \gamma_{1/2/3}(t) \in \text{dom}(\phi)$. Since the domains of the charts are open and cover the manifold this is always possible and in fact we can always select ϕ so that $t_0 = \frac{\pi}{2}$. Inserting the chart and its inverse allows us to apply the chain rule:

$$\mathcal{A}_i|_g f = (f \circ \phi^{-1} \circ \phi \circ \gamma_{i,g})'(0) = \sum_{j=1}^3 \partial_j (f \circ \phi^{-1})(\phi(g)) \cdot (\phi_j \circ \gamma_{i,g})'(0). \tag{1.5}$$

We will work these out further for each curve. Let $\phi(g) = (x, y, \theta)$ then the curves translate through the chart as follows:

$$\phi \circ \gamma_{1,g}(t) = \begin{pmatrix} x + t \cos \theta \\ y + t \sin \theta \\ \theta \end{pmatrix}, \quad \phi \circ \gamma_{2,g}(t) = \begin{pmatrix} x - t \sin \theta \\ y + t \cos \theta \\ \theta \end{pmatrix}, \quad \phi \circ \gamma_{3,g}(t) = \begin{pmatrix} x \\ y \\ \theta + t \end{pmatrix}. \tag{1.6}$$

Using these in (1.5) we get:

Proposition 1.11 ($(\mathcal{A}_i)_{i=1}^3$ in the helical atlas). Let $\phi \in \Phi$ so that $\phi(g) = (x, y, \theta)$ and $f \in C^1(\mathbb{M}_2, \mathbb{R})$ then:

$$\begin{aligned}\mathcal{A}_1|_g f &= \partial_1 (f \circ \phi^{-1})(\phi(g)) \cos \theta + \partial_2 (f \circ \phi^{-1})(\phi(g)) \sin \theta, \\ \mathcal{A}_2|_g f &= -\partial_1 (f \circ \phi^{-1})(\phi(g)) \sin \theta + \partial_2 (f \circ \phi^{-1})(\phi(g)) \cos \theta, \\ \mathcal{A}_3|_g f &= \partial_3 (f \circ \phi^{-1})(\phi(g)).\end{aligned}$$

Or less verbose:

$$\begin{aligned}\mathcal{A}_1|_g &= \cos \theta \partial_x + \sin \theta \partial_y, \\ \mathcal{A}_2|_g &= -\sin \theta \partial_x + \cos \theta \partial_y, \\ \mathcal{A}_3|_g &= \partial_\theta,\end{aligned}$$

with $\partial_{x/y/\theta} f(g) := \partial_{1/2/3} (f \circ \phi^{-1})(\phi(g))$.

We will prove that these three operators are linearly independent by applying them to the following three functions:

$$\begin{aligned}f_{1,g} \circ \phi^{-1}(x, y, \theta) &= n^1(x - x^1) + n^2(y - x^2), \\ f_{2,g} \circ \phi^{-1}(x, y, \theta) &= -n^2(x - x^1) + n^1(y - x^2), \\ f_{3,g} \circ \phi^{-1}(x, y, \theta) &= \theta.\end{aligned}\tag{1.7}$$

It's easy to check that $\mathcal{A}_i|_g f_{j,g} = \delta_{ij}$, which means that the only way for $\sum_{i=1}^3 \lambda^i \mathcal{A}_i|_g$ to equal zero is if all λ^i 's are zero, confirming that $\mathcal{A}_1|_g, \mathcal{A}_2|_g$ and $\mathcal{A}_3|_g$ are linearly independent. Since $\dim(\mathbb{M}_2) = 3$ it follows that the dimension of the tangent space at any point is the same, we conclude that our three operators are a basis for the tangent space and so we can say:

Proposition 1.12 (Tangent bundle basis).

$$T_g \mathbb{M}_2 = \langle \mathcal{A}_1|_g, \mathcal{A}_2|_g, \mathcal{A}_3|_g \rangle,$$

and extend it to the tangent bundle:

$$T\mathbb{M}_2 = \{ (g, \dot{\mathbf{g}}) \mid g \in \mathbb{M}_2, \dot{\mathbf{g}} \in T_g \mathbb{M}_2 \}.$$

Remark 1.13 (Notation). As we have been doing we will denote group elements with normal lower case letters ($g, h, \dots$), tangent vector we will denote with bold lower case letters ($\mathbf{v}, \mathbf{w}, \dots$) or if we want to be more explicit about which tangent space these belong to we bold and dot the name of the group element to whose tangent space the vector belongs ($\dot{\mathbf{g}}, \dot{\mathbf{h}}, \dots$). The exception to this rule are basis vectors which we denote by cursive capital letters ($\mathcal{A}, \mathcal{B}, \dots$) to stress that they are defined as operators. For *vector fields* (i.e. maps from the manifold into the tangent bundle so that each $g \in \mathbb{M}_2$ maps to an element of its own tangent space $T_g \mathbb{M}_2$) we will use the common notation of normal upper case letters ($V, W, \dots$) to distinguish them from individual tangent vectors.

Using the basis $(\mathcal{A}_i|_g)_{i=1}^3$ any $\dot{\mathbf{g}}$ can be written as:

$$\dot{\mathbf{g}} = \dot{g}^1 \mathcal{A}_1|_g + \dot{g}^2 \mathcal{A}_2|_g + \dot{g}^3 \mathcal{A}_3|_g, \quad \dot{g}^1, \dot{g}^2, \dot{g}^3 \in \mathbb{R}.\tag{1.8}$$

In fact often we will just identify a tangent vector with its real components: $\dot{\mathbf{g}} \simeq (\dot{g}^1, \dot{g}^2, \dot{g}^3)^T$ with respect to this basis. To return to the vehicle dynamics analogy, $\dot{g}^1$ can be thought of as the longitudinal or straight line velocity, $\dot{g}^2$ as the lateral or drift velocity and $\dot{g}^3$ as the yaw-rate.

We will encounter smooth vector fields sufficiently often to warrant a symbol for their set.

Definition 1.14 (Smooth vector fields). $\mathfrak{X}(\mathbb{M}_2)$ is the set of all vector fields on $\mathbb{M}_2$ that when applied to a smooth function on the manifold yields another smooth function on the manifold.

Naturally the left invariant component of such a vector field are similarly smooth.

Corollary 1.15. *Smooth vector fields have smooth component functions with respect to the left invariant frame.*

Definition 1.16 (Dual basis). We define the co-vectors $(\omega^i|_g)_{i=1}^3 \in T_g^*\mathbb{M}_2$ by their action on the tangent basis vectors:

$$\omega^i|_g(\mathcal{A}_j|_g) = \delta_j^i.$$

Being the dual basis to the left-invariant basis naturally makes these co-vectors into the left-invariant basis of the cotangent bundle.

Corollary 1.17 (Cotangent bundle basis).

$$T_g^*\mathbb{M}_2 = \langle \omega^1|_g, \omega^2|_g, \omega^3|_g \rangle,$$

which extends to the cotangent bundle:

$$T^*\mathbb{M}_2 = \{(g, \omega) \mid g \in \mathbb{M}_2, \omega \in T_g^*\mathbb{M}_2\}.$$

1.1.3 Left Invariance

Calling the components of $\dot{\mathbf{g}} \in T_g\mathbb{M}_2$ with respect to the basis $(\mathcal{A}_i|_g)_{i=1}^3$ velocities implies that they are somehow invariant, obviously a car's speed and yaw-rate is not dependent on where the observer is looking from. Indeed this is the case with respect to multiplication from the left with a group element, hence the name *left invariant*. To specify this property formally we introduce some definitions.

Definition 1.18 (Left-regular representation on $\mathbb{M}_2$). Let $f : \mathbb{M}_2 \rightarrow \mathbb{R}$, then we define:

$$(\mathcal{L}_g f)(h) = f(g^{-1} \odot h), \quad \forall g, h \in \mathbb{M}_2.$$

Definition 1.19 (Left-regular representation on $\mathbb{R}^2$). Let $f : \mathbb{R}^2 \rightarrow \mathbb{R}$ and $g = (\mathbf{x}, \mathbf{n}) \in \mathbb{M}_2$, then we define:

$$(\mathcal{U}_g f)(\mathbf{y}) = f(R_{\mathbf{n}}^{-1}(\mathbf{y} - \mathbf{x})), \quad \forall \mathbf{y} \in \mathbb{R}^2.$$

And an alternative notation for left multiplication, both in the manifold and in the parameter space of a chart:

Definition 1.20 (Left multiplication as operators).

$$\begin{aligned} L_g(h) &= g \odot h, \quad \forall g, h \in \mathbb{M}_2, \\ L_{\phi, g}(x, y, \theta) &= \phi(g \odot \phi^{-1}(x, y, \theta)), \quad \forall (x, y, \theta) \in \text{dom}(\phi). \end{aligned}$$

Theorem 1.21 (Left invariance of $\mathcal{A}_i$). Let $f \in C^1(\mathbb{M}_2, \mathbb{R})$ and $g \in \mathbb{M}_2$ then:

$$\mathcal{A}_i(\mathcal{L}_g f) = \mathcal{L}_g(\mathcal{A}_i f).$$

Proof. Let $\phi(g) = (x_g, y_g, \theta_g)$ for some choice of chart $\phi \in \Phi$. Then we have the following:

$$\begin{aligned}
\mathcal{A}_1|_g (\mathcal{L}_g f) &= \partial_1 (\mathcal{L}_g f \circ \phi^{-1}) (\phi(g)) \cos \theta_g + \partial_2 (\mathcal{L}_g f \circ \phi^{-1}) (\phi(g)) \sin \theta_g \\
&= \partial_1 (f \circ \phi^{-1} \circ L_{\phi, g^{-1}}) (\phi(g)) \cos \theta_g + \partial_2 (f \circ \phi^{-1} \circ L_{\phi, g^{-1}}) (\phi(g)) \sin \theta_g \\
&= \sum_{i=1}^3 \partial_i (f \circ \phi^{-1}) (L_{\phi, g^{-1}}(\phi(g))) \partial_1 L_{\phi, g^{-1}, i} (\phi(g)) \cos \theta_g \\
&\quad + \sum_{j=1}^3 \partial_j (f \circ \phi^{-1}) (L_{\phi, g^{-1}}(\phi(g))) \partial_2 L_{\phi, g^{-1}, i} (\phi(g)) \sin \theta_g \\
&= \partial_1 (f \circ \phi^{-1}) (\phi(I)) \cos^2 \theta_g + \partial_2 (f \circ \phi^{-1}) (\phi(I)) \sin \theta_g \cos \theta_g \\
&\quad + \partial_1 (f \circ \phi^{-1}) (\phi(I)) \sin^2 \theta_g - \partial_2 (f \circ \phi^{-1}) (\phi(I)) \cos \theta_g \sin \theta_g \\
&= \partial_1 (f \circ \phi^{-1}) (\phi(I)) \\
&= \mathcal{A}_1|_I f.
\end{aligned}$$

And similarly we find that $\mathcal{A}_2|_g (\mathcal{L}_g f) = \mathcal{A}_2|_I f$ and $\mathcal{A}_3|_g (\mathcal{L}_g f) = \mathcal{A}_3|_I f$. Now take $g, h \in \mathbb{M}_2$:

$$\begin{aligned}
(\mathcal{A}_i (\mathcal{L}_g f)) (h) &= \mathcal{A}_i|_h (\mathcal{L}_g f) \\
&= \mathcal{A}_i|_h (\mathcal{L}_h \mathcal{L}_{h^{-1} \odot g} f) \\
&= \mathcal{A}_i|_I (\mathcal{L}_{h^{-1} \odot g} f) \\
&= \mathcal{A}_i|_{g^{-1} \odot h} (\mathcal{L}_{g^{-1} \odot h} \mathcal{L}_{h^{-1} \odot g} f) \\
&= \mathcal{A}_i|_{g^{-1} \odot h} f \\
&= (\mathcal{A}_i f) (g^{-1} \odot h) \\
&= (\mathcal{L}_g (\mathcal{A}_i f)) (h).
\end{aligned}$$

■

Meaning it does not matter whether we apply the operator first and then roto-translate the function or the other way around, a most desirable property for image processing.

Because of this left-invariant property we refer to $(\mathcal{A}_i|_g)_{i=1}^3$ as the *left-invariant derivative frame* at g and define *left-invariant vector fields* as follows:

Definition 1.22 (Left invariant vector field). Let $\lambda^1, \lambda^2, \lambda^3 \in \mathbb{R}$, then we call $\left\{ \sum_{i=1}^3 \lambda^i \mathcal{A}_i|_g \mid \forall g \in \mathbb{M}_2 \right\}$ a left-invariant vector field.

Note how such a vector field is entirely determined by the three λ 's.

1.1.4 Non-Commutativity and Lie Brackets

Now that we have constructed first-order differential operators we can introduce repeated differentiation. Notation-wise we will shorten repeated differentiation as follows.

Definition 1.23 (Repeated differentiation).

$$(\mathcal{A}_{ij}f)(g) := \mathcal{A}_{ij}\Big|_g f := (\mathcal{A}_j \circ \mathcal{A}_i f)(g) = \mathcal{A}_j\Big|_g (h \mapsto \mathcal{A}_i\Big|_h f).$$

Since the group operation $\odot$ does not commute vector fields do not commute either (in general). To indicate in what way commutativity is broken we use the *Lie bracket*.

Definition 1.24 (Lie bracket). Let $V, W \in \mathfrak{X}$ then

$$[V, W] := V \circ W - W \circ V$$

is the Lie bracket of V and W , where we compose vector fields as:

$$((V \circ W)f)(g) := V\Big|_g (h \mapsto W\Big|_h f).$$

Immediately from its definition we can deduce the following properties of the Lie bracket.

Corollary 1.25 (Lie bracket properties). Let $U, V, W \in \mathfrak{X}$ then we have:

1. *Bilinearity*, let $a, b \in \mathbb{R}$:

$$\begin{aligned} [aU + bV, W] &= a[U, W] + b[V, W] \\ [U, aV + bW] &= a[U, V] + b[U, W]. \end{aligned}$$

2. *Antisymmetry*:

$$[V, W] = -[W, V].$$

3. *Jacobi Identity*:

$$[U, [V, W]] + [V, [W, U]] + [W, [U, V]] = 0.$$

4. Let $f_1, f_2 \in C^\infty$, then:

$$[f_1 U, f_2 V] = f_1 f_2 [U, V] + (f_1 U f_2) V - (f_2 V f_1) U.$$

5. *Leibniz rule*, let $f_1, f_2 \in C^\infty$:

$$[U, V](f_1 f_2) = f_1 [U, V] f_2 + f_2 [U, V] f_1.$$

Since the Lie bracket obeys the Leibniz rule it is a derivation and we can infer that it produces a tangent vector field, when we calculate the bracket for $\mathbb{M}_2$ we will see that that is exactly the case².

It suffices to calculate the Lie brackets for the basis left-invariant vector fields $(\mathcal{A}_i)_{i=1}^3$, from which we can then calculate the Lie bracket for an arbitrary smooth vector field using the above properties. Obviously $[\mathcal{A}_i, \mathcal{A}_i] = 0$ and since we have antisymmetry it suffices to calculate $[\mathcal{A}_1, \mathcal{A}_2]$, $[\mathcal{A}_1, \mathcal{A}_3]$ and $[\mathcal{A}_2, \mathcal{A}_3]$.

Lemma 1.26 (Lie brackets of the left invariant frame).

$$[\mathcal{A}_1, \mathcal{A}_2] = \mathbf{0}, \quad [\mathcal{A}_1, \mathcal{A}_3] = -\mathcal{A}_2, \quad [\mathcal{A}_2, \mathcal{A}_3] = \mathcal{A}_1.$$

Proof. Using the fact that in $\mathbb{R}^3$ we have $[\partial_i, \partial_j] = 0$ and proposition 1.11 we calculate:

$$\begin{aligned} [\mathcal{A}_1, \mathcal{A}_2] &= [\cos \theta \partial_x + \sin \theta \partial_y, -\sin \theta \partial_x + \cos \theta \partial_y] \\ &= -[\cos \theta \partial_x, \sin \theta \partial_x] + [\cos \theta \partial_x, \cos \theta \partial_y] - [\sin \theta \partial_y, \sin \theta \partial_x] + [\sin \theta \partial_y, \cos \theta \partial_y] \\ &= -\cos \theta \sin \theta [\partial_x, \partial_x] + \cos^2 \theta [\partial_x, \partial_y] - \sin^2 \theta [\partial_y, \partial_x] + \cos \theta \sin \theta [\partial_y, \partial_y] \\ &= \mathbf{0}. \end{aligned}$$

²For a general proof see [14, Lemma 8.25]

$$\begin{aligned}
[\mathcal{A}_1, \mathcal{A}_3] &= [\cos \theta \partial_x + \sin \theta \partial_y, \partial_\theta] \\
&= [\cos \theta \partial_x, \partial_\theta] + [\sin \theta \partial_y, \partial_\theta] \\
&= \cos \theta [\partial_x, \partial_\theta] - (\partial_\theta \cos \theta) \partial_x + \sin \theta [\partial_y, \partial_\theta] - (\partial_\theta \sin \theta) \partial_y \\
&= \sin \theta \partial_x - \cos \theta \partial_y \\
&= -\mathcal{A}_2.
\end{aligned}$$

$$\begin{aligned}
[\mathcal{A}_2, \mathcal{A}_3] &= [-\sin \theta \partial_x + \cos \theta \partial_y, \partial_\theta] \\
&= -[\sin \theta \partial_x, \partial_\theta] + [\cos \theta \partial_y, \partial_\theta] \\
&= -\sin \theta [\partial_x, \partial_\theta] + (\partial_\theta \sin \theta) \partial_x + \cos \theta [\partial_y, \partial_\theta] - (\partial_\theta \cos \theta) \partial_y \\
&= \cos \theta \partial_x + \sin \theta \partial_y \\
&= \mathcal{A}_1.
\end{aligned}$$

■

Which also proves our claim that (at least on $\mathbb{M}_2$) the Lie bracket produces a tangent vector field, and moreover if $V, W \in \mathfrak{X}$ then also $[V, W] \in \mathfrak{X}$. Schematically the relations between 2nd order derivatives that we found can be summarized in the following diagrams.

$$\begin{array}{ccc}
\begin{array}{ccc} f & \xrightarrow{\mathcal{A}_1} & \mathcal{A}_1 f \\ \downarrow \mathcal{A}_2 & & \downarrow \mathcal{A}_2 \\ \mathcal{A}_2 f & \xrightarrow{\mathcal{A}_1} & \mathcal{A}_{12} f = \mathcal{A}_{21} f \end{array} &
\begin{array}{ccc} f & \xrightarrow{\mathcal{A}_1} & \mathcal{A}_1 f \xrightarrow{\mathcal{A}_3} \mathcal{A}_{13} f \\ \downarrow \mathcal{A}_3 & & \downarrow \mathcal{A}_3 \\ \mathcal{A}_3 f & & \downarrow \mathcal{A}_1 \\ \mathcal{A}_{31} f & \xleftarrow{-\mathcal{A}_2} & \end{array} &
\begin{array}{ccc} f & \xrightarrow{\mathcal{A}_2} & \mathcal{A}_2 f \xrightarrow{\mathcal{A}_3} \mathcal{A}_{23} f \\ \downarrow \mathcal{A}_3 & & \downarrow \mathcal{A}_3 \\ \mathcal{A}_3 f & & \downarrow \mathcal{A}_2 \\ \mathcal{A}_{32} f & \xleftarrow{+\mathcal{A}_1} & \end{array}
\end{array} \quad (1.9)$$

A more intuitive geometric interpretation of the Lie bracket and these diagrams can be seen in figure 1.5.

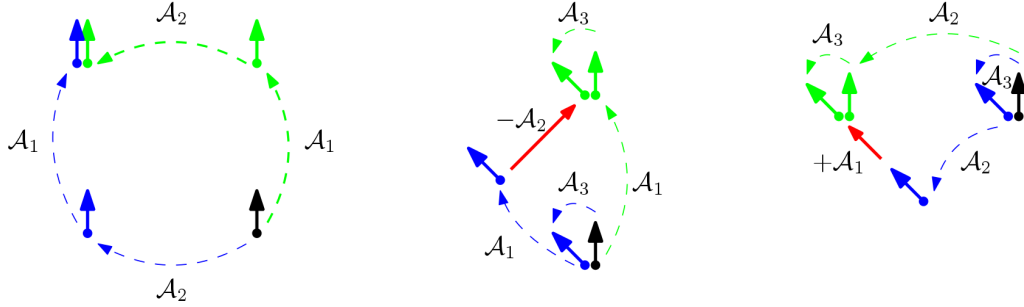


Figure 1.5: Geometric interpretation of the $\mathbb{M}_2$ Lie brackets where we interpret $\mathcal{A}_1$ as a small step forward, $\mathcal{A}_2$ as a small step to the left and $\mathcal{A}_3$ as a small counter-clockwise rotation. Starting at the black marker we can take the green order or the blue order of operations, the Lie bracket (in red) is then the move we need to make to reconcile the difference in final position and orientation.

The action of the Lie bracket on the basis vectors can be encoded using the structure constants.

Definition 1.27 (Structure constants). $c_{ij}^k \in \mathbb{R}$ obeying

$$[\mathcal{A}_i, \mathcal{A}_j] = \sum_{k=1}^3 c_{ij}^k \mathcal{A}_k$$

are called the structure constants.

Under this definition the structure constants of $\mathbb{M}_2$ follow immediately from lemma 1.26.

Corollary 1.28. *The non-zero structure constants of $\mathbb{M}_2$ are*

$$c_{13}^2 = -1, \quad c_{31}^2 = 1, \quad c_{23}^1 = 1, \quad c_{32}^1 = -1.$$

1.1.5 Metric Tensor, Gradient & Divergence

Now that we have the tangent bundle we can define a metric on it. Two issues need to be considered:

1. how do we weigh yaw-rate against spatial velocity? If we assigned a unit, say meters, to the spatial part of the manifold we need some way to weigh meters against radians.
2. how do we weigh lateral velocity against longitudinal velocity given that we might prefer longitudinal over lateral movement to some degree?

The first issue we tackle by introducing the constants $D_S, D_A > 0$ that will weigh the spatial respectively the angular components of the tangent vector. To tackle the second issue we introduce $0 < \zeta \leq 1$ to give an additional independent weight to lateral movement.

Using these we define the following metric tensor field, where we recall that ω^i is the covector dual to $\mathcal{A}_i$ per definition 1.16:

Definition 1.29 (Riemannian metric tensor). Let $D_S, D_A > 0$ and $0 < \zeta \leq 1$, then we define:

$$(\cdot, \cdot)_{\mathcal{G}_\zeta} := \mathcal{G}_\zeta^{\text{left}} := \frac{1}{D_S} \omega^1 \otimes \omega^1 + \frac{1}{\zeta^2 D_S} \omega^2 \otimes \omega^2 + \frac{1}{D_A} \omega^3 \otimes \omega^3.$$

Or more concretely if we have two vectors written $\mathbf{v}, \mathbf{w} \in T_g \mathbb{M}_2$ so that $\mathbf{v} = \sum_{i=1}^3 v^i \mathcal{A}_i|_g$ and $\mathbf{w} = \sum_{i=1}^3 w^i \mathcal{A}_i|_g$, then:

$$(\mathbf{v}, \mathbf{w})_{\mathcal{G}_\zeta}|_g := \mathcal{G}_\zeta^{\text{left}}|_g(\mathbf{v}, \mathbf{w}) := \frac{v^1 w^1}{D_S} + \frac{v^2 w^2}{\zeta^2 D_S} + \frac{v^3 w^3}{D_A}.$$

Take note of the fact that under this metric the left-invariant frame is orthogonal. In matrix notation (as linear operator with respect to the left invariant basis) we can write the metric and its determinant (with respect to the left-invariant basis) as:

$$\mathcal{G}_\zeta^{\text{left}} \simeq \begin{pmatrix} D_S^{-1} & 0 & 0 \\ 0 & \zeta^{-2} D_S^{-1} & 0 \\ 0 & 0 & D_A^{-1} \end{pmatrix}, \quad \det(\mathcal{G}_\zeta^{\text{left}}) = \frac{1}{\zeta^2 D_S^2 D_A}. \quad (1.10)$$

Which are constant along the manifold thanks to our choice of left-invariant differential frame.

An important particular case we will consider is $\zeta \rightarrow 0$, this gives the sub-Riemannian case where we only allow curves $g(t) = (\mathbf{x}(t), \mathbf{n}(t))$ where $\frac{\dot{\mathbf{x}}}{\|\dot{\mathbf{x}}\|} = \mathbf{n}$. In this case the tangent space would have as basis just $\{\mathcal{A}_1, \mathcal{A}_3\}$ with metric:

Definition 1.30 (Sub-Riemannian metric tensor). Let $D_S, D_A > 0$, then we define:

$$(\cdot, \cdot)_{\mathcal{G}_0} := \mathcal{G}_0^{\text{left}} := \frac{1}{D_S} \omega^1 \otimes \omega^1 + \frac{1}{D_A} \omega^3 \otimes \omega^3.$$

Or more concretely let $\mathbf{v}, \mathbf{w} \in T_g \mathbb{M}_2$ so that $\mathbf{v} = v^1 \mathcal{A}_1|_g + v^3 \mathcal{A}_3|_g$ and $\mathbf{w} = w^1 \mathcal{A}_1|_g + w^3 \mathcal{A}_3|_g$, then:

$$(\mathbf{v}, \mathbf{w})_{\mathcal{G}_0^{\text{left}}}|_g := \mathcal{G}_0^{\text{left}}|_g(\mathbf{v}, \mathbf{w}) := \frac{v^1 w^1}{D_S} + \frac{v^3 w^3}{D_A}.$$

Or in matrix notation with determinant:

$$\mathcal{G}_0^{\text{left}} \simeq \begin{pmatrix} D_S^{-1} & 0 \\ 0 & D_A^{-1} \end{pmatrix}, \quad \det(\mathcal{G}_0^{\text{left}}) = \frac{1}{D_S D_A}. \quad (1.11)$$

The vehicle analogue here would be a car being driven within the limits of its traction, i.e. no drift.

For brevity of notation we introduce the following.

Definition 1.31.

$$D_i = \begin{cases} D_S & \text{if } i = 1, \\ \zeta^2 D_S & \text{if } i = 2, \\ D_A & \text{if } i = 3. \end{cases}$$

Which allows, for example, to abbreviate the metric as:

$$\mathcal{G}_\zeta^{\text{left}} = \sum_{i=1}^3 \frac{1}{D_i} \omega^i \otimes \omega^i. \quad (1.12)$$

The metric defines an isomorphism between the tangent space and cotangent space by only filling the first argument but keeping the second slot open as follows:

Definition 1.32 (Metric Isomorphism).

We overload the symbol $\mathcal{G}_\zeta$ as $\mathcal{G}_\zeta : T\mathbb{M}_2 \rightarrow T^*\mathbb{M}_2$ so that:

$$\mathcal{G}_\zeta(\mathbf{v}) := (\mathbf{v}, \cdot)_{\mathcal{G}_\zeta} \Rightarrow (\mathcal{G}_\zeta(\mathbf{v}))(\mathbf{w}) = (\mathbf{v}, \mathbf{w})_{\mathcal{G}_\zeta}.$$

If we have $\mathbf{v} = \sum_{i=1}^3 v^i \mathcal{A}_i|_g$ this gives:

$$\mathcal{G}_\zeta^{\text{left}}(\mathbf{v}) = \frac{v^1}{D_S} \omega^1 + \frac{v^2}{\zeta^2 D_S} \omega^2 + \frac{v^3}{D_A} \omega^3, \quad \mathcal{G}_0^{\text{left}}(\mathbf{v}) = \frac{v^1}{D_S} \omega^1 + \frac{v^3}{D_A} \omega^3. \quad (1.13)$$

It is easy to see that this map is bijective and has the following inverse.

Corollary 1.33 (Inverse Metric Isomorphism). *Given a covector $\mathbf{v} = \sum_{i=1}^3 v_i \omega^i|_g$ we have:*

$$\mathcal{G}_\zeta^{-1}|_g(\mathbf{v}) = v_1 D_S \mathcal{A}_1|_g + v_2 \zeta^2 D_S \mathcal{A}_2|_g + v_3 D_A \mathcal{A}_3|_g.$$

The volume form belonging to these metrics is as follows:

Corollary 1.34 (Volume form).

$$d\mu^{\text{left}}(g) = \begin{cases} \frac{1}{\zeta D_S \sqrt{D_A}} \omega^1|_g \wedge \omega^2|_g \wedge \omega^3|_g & \text{for } \zeta > 0, \\ \frac{1}{\sqrt{D_S D_A}} \omega^1|_g \wedge \omega^3|_g & \text{for } \zeta = 0. \end{cases}$$

Remark 1.35. For the purpose of dimensional consistency we will take a brief look at units. Since we intend to work on images we will use pixels (px) as our spatial unit and use seconds (s) for time, angles are of course dimensionless radians but we will still denote them with rad as if they have a unit. With those units we choose D_S to be dimensionless, which also makes ζ dimensionless and gives D_A the unit of $\frac{rad^2}{px^2}$.

Remark 1.36. In this chapter the constants D_S and D_A take the role that ξ takes in other works

[15, 16]. The relationship being as follows:

$$\xi = \sqrt{\frac{D_A}{D_S}}. \quad (1.14)$$

Whether one introduces D_S and D_A or ξ to proportion the angular against the spatial dimension is a matter of preference, in this chapter we will use D_S , D_A to stay consistent with [17].

From the metric we derive the gradient $\nabla_\zeta f$ of a differentiable function f as the unique vector that satisfies $(\nabla_\zeta f, V)_{\mathcal{G}_\zeta} = V f$ for any vector field V .

Lemma 1.37 (Riemannian gradient). *With respect to the tangent basis $(\mathcal{A}_i)_{i=1}^3$ and the metric $\mathcal{G}_\zeta^{\text{left}}$ the Riemannian gradient of a differentiable function f is given by:*

$$\nabla_\zeta^{\text{left}} f(g) = D_S (\mathcal{A}_1|_g f) \mathcal{A}_1|_g + \zeta^2 D_S (\mathcal{A}_2|_g f) \mathcal{A}_2|_g + D_A (\mathcal{A}_3|_g f) \mathcal{A}_3|_g. \quad (1.15)$$

Or for a cleaner notation we can suppress the argument g and use the D_i constants:

$$\nabla_\zeta^{\text{left}} f = D_S (\mathcal{A}_1 f) \mathcal{A}_1 + \zeta^2 D_S (\mathcal{A}_2 f) \mathcal{A}_2 + D_A (\mathcal{A}_3 f) \mathcal{A}_3 = \sum_{i=1}^3 D_i (\mathcal{A}_i f) \mathcal{A}_i.$$

Proof. Let V be any vector field.

$$\begin{aligned} (\nabla_\zeta^{\text{left}} f, V)_{\mathcal{G}_\zeta^{\text{left}}} = V f &\iff \mathcal{G}_\zeta^{\text{left}} (\nabla_\zeta f) (V) = V f \\ &\iff \mathcal{G}_\zeta^{\text{left}} (\nabla_\zeta f) = (\mathcal{A}_1 f) \omega^1 + (\mathcal{A}_2 f) \omega^2 + (\mathcal{A}_3 f) \omega^3 \\ &\iff \nabla_\zeta^{\text{left}} f = \mathcal{G}_\zeta^{\text{left}-1} ((\mathcal{A}_1 f) \omega^1 + (\mathcal{A}_2 f) \omega^2 + (\mathcal{A}_3 f) \omega^3) \\ &\iff \nabla_\zeta^{\text{left}} f = D_S (\mathcal{A}_1 f) \mathcal{A}_1 + \zeta^2 D_S (\mathcal{A}_2 f) \mathcal{A}_2 + D_A (\mathcal{A}_3 f) \mathcal{A}_3. \end{aligned}$$

We can readily see that this is a unique assignment since the maps $\mathcal{G}_\zeta^{\text{left}} / \mathcal{G}_\zeta^{\text{left}-1}$ give us an isomorphism. ■

Given a $f \in C^1(\mathbb{M}_2, \mathbb{R})$ the norm of the gradient is given by:

$$\|\nabla_\zeta f\|_{\mathcal{G}_\zeta^{\text{left}}} = \sqrt{D_S (\mathcal{A}_1 f)^2 + \zeta^2 D_S (\mathcal{A}_2 f)^2 + D_A (\mathcal{A}_3 f)^2}. \quad (1.16)$$

Note how the gradient and its norm behave predictably as $\zeta \rightarrow 0$. For the sub-Riemannian case of $\zeta = 0$ we reduce the gradient to its relevant components:

Corollary 1.38 (Sub-Riemannian gradient). *With respect to the tangent basis $(\mathcal{A}_i)_{i=1}^3$ the sub-Riemannian gradient of a function $f \in C^1(\mathbb{M}_2)$ is given by:*

$$\nabla_0^{\text{left}} f = D_S (\mathcal{A}_1 f) \mathcal{A}_1 + D_A (\mathcal{A}_3 f) \mathcal{A}_3,$$

and its attendant norm:

$$\|\nabla_0^{\text{left}} f\|_{\mathcal{G}_0^{\text{left}}} = \sqrt{D_S (\mathcal{A}_1 f)^2 + D_A (\mathcal{A}_3 f)^2}. \quad (1.17)$$

Now consider a vector field $V(g) = v^1(g) \mathcal{A}_1|_g + v^2(g) \mathcal{A}_2|_g + v^3(g) \mathcal{A}_3|_g$ where $v^i \in C^1(\mathbb{M}_2, \mathbb{R})$. Then we have the following formula for the divergence, which simplifies beautifully since the metric is constant across the left-invariant frames:

$$\text{div}(V) = \sum_{i=1}^3 \frac{1}{|\det(\mathcal{G}_\zeta^{\text{left}})|} \mathcal{A}_i \left(\left| \det(\mathcal{G}_\zeta^{\text{left}}) \right| v_i \right) = \sum_{i=1}^3 \mathcal{A}_i v^i. \quad (1.18)$$

This expression for the divergence in $\mathbb{M}_2$ can actually be related to the divergences in its constituent parts $\mathbb{R}^2$ and S^1 . Consider the static differential frame $(\partial_x, \partial_y, \partial_\theta)$ induced by a chart $\phi \in \Phi$ in which V has components (v^x, v^y, v^θ) , then (recall that $g = (x^1 \mathbf{e}_1 + x^2 \mathbf{e}_2, n^1 \mathbf{e}_1 + n^2 \mathbf{e}_2)$):

$$\begin{aligned}
\operatorname{div}(V)|_g &= \sum_{i=1}^3 \mathcal{A}_i|_g v^i \\
&= (n^1 \partial_x + n^2 \partial_y) (n^1 v_x(g) + n^2 v_y(g)) \\
&\quad + (-n^2 \partial_x + n^1 \partial_y) (-n^2 v^x(g) + n^1 v^y(g)) \\
&\quad + \partial_\theta v^\theta(g) \\
&= (n^1)^2 \partial_x v^x(g) + n^1 n^2 \partial_x v^y(g) + n^1 n^2 \partial_y v^x(g) + (n^2)^2 \partial_y v^y(g) \\
&\quad + (n^2)^2 \partial_x v^x(g) - n^1 n^2 \partial_x v^y(g) - n^1 n^2 \partial_y v^x(g) + (n^1)^2 \partial_y v^y(g) \\
&\quad + \partial_\theta v^\theta(g) \\
&= ((n^1)^2 + (n^2)^2) \partial_x v^x(g) + ((n^1)^2 + (n^2)^2) \partial_y v^y(g) + \partial_\theta v^\theta(g) \\
&= \partial_x v^x(g) + \partial_y v^y(g) + \partial_\theta v^\theta(g) \\
&= \operatorname{div}_{\mathbb{R}^2} V|_g + \operatorname{div}_{S^1} V|_g.
\end{aligned}$$

We can treat the sub-Riemannian case similarly to find the following general relation:

Corollary 1.39 (Divergence). *Let $V(g) = v^1(g)\mathcal{A}_1|_g + v^2(g)\mathcal{A}_2|_g + v^3(g)\mathcal{A}_3|_g$ where $v^i \in C^1(\mathbb{M}_2, \mathbb{R})$ then:*

$$\operatorname{div}(V) = \begin{cases} \mathcal{A}_1 v^1 + \mathcal{A}_2 v^2 + \mathcal{A}_3 v^3 = \operatorname{div}_{\mathbb{R}^2}(V) + \operatorname{div}_{S^1}(V) & \text{for } \zeta > 0, \\ \mathcal{A}_1 v^1 + \mathcal{A}_3 v^3 = \partial_{\mathbf{n}}(n^1 v^x + n^2 v^y) + \operatorname{div}_{S^1}(V) & \text{for } \zeta = 0. \end{cases}$$

On $\mathbb{R}^n$ we have the familiar identity $\operatorname{div}(fV) = f \operatorname{div}(V) + \nabla f \cdot V$, on $\mathbb{M}_2$ we have a similar rule.

Lemma 1.40 (Divergence product rule). *Given a function $f \in C^1(\mathbb{M}_2, \mathbb{R})$ and differentiable vector field V we have:*

$$\operatorname{div}(fV) = f \operatorname{div}(V) + V f = f \operatorname{div}(V) + \left(\nabla_\zeta^{\text{left}} f, V \right)_{\mathcal{G}_\zeta^{\text{left}}}.$$

Proof.

$$\begin{aligned}
\operatorname{div}(fV) &= \sum_{i=1}^3 \mathcal{A}_i (f v^i) \\
&= \sum_{i=1}^3 (f \mathcal{A}_i v^i + v^i \mathcal{A}_i f) && \text{(Lemma 1.10)} \\
&= f \sum_{i=1}^3 \mathcal{A}_i v^i + \left(\sum_{i=1}^3 v^i \mathcal{A}_i \right) f \\
&= f \operatorname{div}(V) + V f.
\end{aligned}$$

■

1.1.6 Domains and Boundaries

In practice we will not be dealing with the full manifold since images are compactly supported functions with an easily defined boundary. We want to confine our theory to spatially restricted but directionally unrestricted subsets of $\mathbb{M}_2$, we give such subsets the following name:

Definition 1.41 (Aperture). $\Omega \subset \mathbb{M}_2$ is an aperture if there exists a compact subset $\Omega' \subset \mathbb{R}^2$ which has a piecewise smooth boundary such that $\Omega = \Omega' \times S^1$, we write $\Omega \stackrel{\text{ap}}{\subset} \mathbb{M}_2$.

This leads directly to:

Corollary 1.42. *An aperture of $\mathbb{M}_2$ is compact and has a piecewise smooth boundary.*

Restricting ourselves to functions with aperture domains will allow us to use tools that demand a high degree of regularity of the domain boundary. In practice this is still a very loose requirement since images are generally supported on plain rectangles.

1.1.7 Group Distance and Estimates

The metric on the tangent space induces a distance on the elements of the Lie Group as the arc length of the shortest curve connecting the two elements. Because of the left invariance of the metric it suffices to look at the distance from the neutral element to the rest of the group.

Definition 1.43 (Group distance from the origin). Let $g \in \mathbb{M}_2$, then:

$$d_{\mathcal{G}_\zeta}(g) := \inf \left\{ \int_0^1 \sqrt{\mathcal{G}_\zeta|_g(\dot{\gamma}(t), \dot{\gamma}(t))} dt \mid \gamma : \mathbb{R} \rightarrow \mathbb{M}_2 \text{ is piecewise smooth and } \gamma(0) = \mathbf{0}, \gamma(1) = g \right\}.$$

This is somewhat cumbersome to calculate, but we can get a good local estimate on this norm by looking at the following differential equation.

$$\begin{cases} \dot{\gamma}_g(t) = \sum_{i=1}^3 c^i \mathcal{A}_i|_{\gamma(t)}, \\ \gamma_g(0) = \mathbf{0}, \quad \gamma_g(1) = g. \end{cases} \quad (1.19)$$

Which defines a curve with a constant tangent vector relative to the left invariant frame that starts at the neutral element and ends in the group element of interest (also called an exponential curve, but see remark 1.46).

Letting $g = (x, y, \theta)$ and applying the method of characteristics solves this system with the following auxiliary functions.

Definition 1.44.

$$\begin{aligned} c^1(x, y, \theta) &:= \begin{cases} \frac{\theta}{2} \left(y + x \cot\left(\frac{\theta}{2}\right) \right) & \text{if } \theta \neq 0, \\ x & \text{if } \theta = 0, \end{cases} \\ c^2(x, y, \theta) &:= \begin{cases} \frac{\theta}{2} \left(-x + y \cot\left(\frac{\theta}{2}\right) \right) & \text{if } \theta \neq 0, \\ y & \text{if } \theta = 0, \end{cases} \\ c^3(x, y, \theta) &:= \theta. \end{aligned}$$

The idea is to then estimate the distance from the neutral element to g by the norm of the tangent vector, which is constant, let $g = (x, y, \theta)$ then:

$$\begin{aligned} d_{\mathcal{G}_\zeta}(g) &\approx \|\dot{\gamma}_g(0)\|_{\mathcal{G}_\zeta^{\text{left}}} = \sqrt{\frac{c^1(x, y, \theta)^2}{D_S} + \frac{c^2(x, y, \theta)^2}{\zeta^2 D_S} + \frac{\theta^2}{D_A}}, \\ d_{\mathcal{G}_0}(g) &\approx \|\dot{\gamma}_g(0)\|_{\mathcal{G}_0^{\text{left}}} = \sqrt[4]{\left(\frac{c^1(x, y, \theta)^2}{D_S} + \frac{\theta^2}{D_A}\right)^2 + \frac{c^3(x, y, \theta)^2}{\zeta^2 D_S D_A}}. \end{aligned} \quad (1.20)$$

This estimate is locally equivalent to the distance by the Ball-Box theorem [18], i.e.

$$C_1 \|\dot{\gamma}_g(0)\|_{\mathcal{G}_\zeta^{\text{left}}} \leq d_{\mathcal{G}_\zeta}(g) \leq C_2 \|\dot{\gamma}_g(0)\|_{\mathcal{G}_\zeta^{\text{left}}}, \quad (1.21)$$

with C_1 and C_2 being scalar constants. These estimates are illustrated in figure 1.6. For more on these estimates see [19, 20, 21].

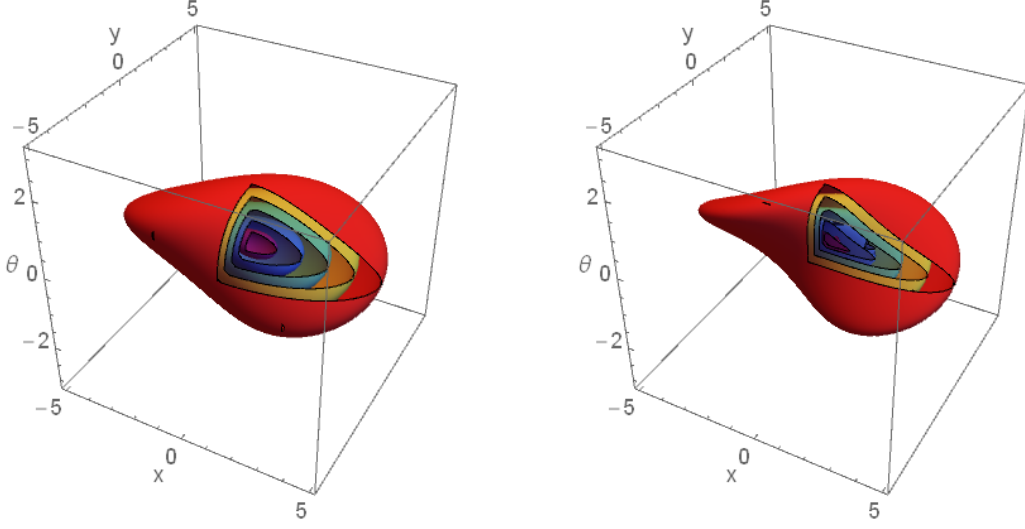


Figure 1.6: Norm estimates for $D_S = 1$, $D_A = 0.1$, $\zeta = 0.5$ for the Riemannian (left) and sub-Riemannian case (right).

1.2 Gauge Frames

The idea of gauge frames in image processing is to choose an orthogonal basis of the tangent space at each location that depends on the data [22], being data dependent this frame is then automatically roto-translation equivariant. We can extend this technique to orientation scores on $\mathbb{M}_2$ so that we can make these operations orientation sensitive [23, 3]. The tool we use to obtain an *aligned* tangent vector is the exponential curve.

1.2.1 Exponential Curves

Exponential curves are the manifold's equivalent to straight lines in Euclidian space, they are the curves that have a constant tangent vector with respect to the left invariant frame (up to reparameterization anyway).

Definition 1.45 (Exponential Curve). Given a $\mathbf{c} = \sum_{i=1}^3 c^i \mathcal{A}_i|_g \in T_g \mathbb{M}_2$ we call $\gamma_{\mathbf{c},g} : \mathbb{R} \rightarrow \mathbb{M}_2$ the exponential curve of $\mathbf{c}$ through g if $\gamma_{\mathbf{c},g}(0) = g$ and $\forall t \in \text{dom}(\gamma_{\mathbf{c},g}) : \dot{\gamma}_{\mathbf{c},g}(t) = \sum_{i=1}^3 c^i \mathcal{A}_i|_{\gamma_{\mathbf{c},g}(t)}$.

Remark 1.46. Our exponential curve definition is not the usual definition from Lie group theory (see [14, Ch. 20] for details). But our definition is more constructive and can be directly used for gauge frame fitting even if it is specific to $\mathbb{M}_2$.

Looking back we see that the movement curves we used in (1.4) to define the left invariant frame are also exponential curves under this definition, where we used the shorthand $\gamma_{i,g} \equiv \gamma_{\mathcal{A}_i|_g,g}$.

Remark that it suffices to look at exponential curves through the Lie group identity element I and that these curves can be transformed to curves through g by left multiplication by g as follows:

$$g \odot \gamma_{\mathbf{c},I}(t) = \gamma_{\mathbf{c},g}(t). \quad (1.22)$$

Alternatively exponential curves can be defined as the exponential map (see [14, Ch. 20]) of t times a constant element of the Lie algebra, hence the name.

1.2.2 Alignment

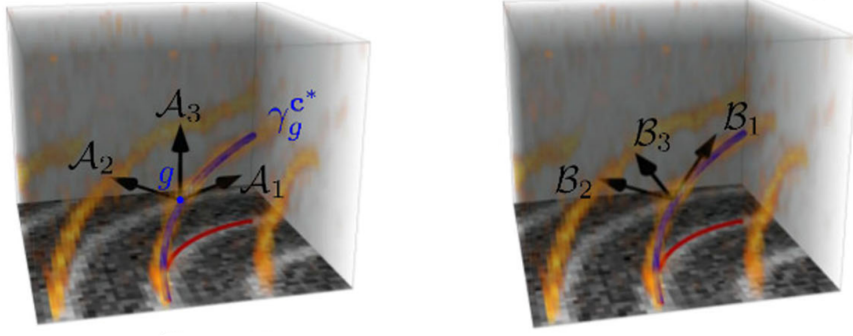


Figure 1.7: Gauge frame fitting: instead of using the left invariant $\mathcal{A}_{1/2/3}$ frame we choose a $\mathcal{B}_{1/2/3}$ frame that is aligned to the local data.

To define a new differential frame we need to select three orthogonal vectors $(\mathcal{B}_i|_g)_{i=1}^3$, we want to select our first vector $\mathcal{B}_1|_g$ aligned with our data at g . Under being *aligned* with our data we understand that if we were on an exponential curve passing through g with tangent $\mathcal{B}_1|_g$ we minimize the change in the components of the gradient³. The idea is illustrated in figure 1.7. Formally:

Definition 1.47 (First Gauge Vector). Let $f \in C^\infty(\mathbb{M}_2, \mathbb{R})$, then

$$\mathcal{B}_1|_g := \operatorname{argmin}_{\mathbf{c} \in T_g \mathbb{M}_2} \left\{ \sum_{i=1}^3 D_i \left(\frac{d}{dt} \mathcal{A}_i|_{\gamma_{\mathbf{c},g}(t)} f \Big|_{t=0} \right)^2 \mid \|\mathbf{c}\|_{\mathcal{G}_\zeta^{\text{left}}} = 1 \right\}.$$

If we have $\mathbf{c} = \sum_{i=1}^3 c^i \mathcal{A}_i|_g$ then we define $\hat{\mathbf{c}} := (c^1, c^2, c^3)^T$. Now looking closer at the quantity we want to minimize:

$$\begin{aligned} \sum_{i=1}^3 D_i \left(\frac{d}{dt} \mathcal{A}_i|_{\gamma_{\mathbf{c},g}(t)} f \Big|_{t=0} \right)^2 &= \sum_{i=1}^3 D_i (\mathbf{c}(\mathcal{A}_i f))^2 \\ &= \sum_{i=1}^3 D_i \left(\left(\sum_{j=1}^3 c^j \mathcal{A}_j|_g \right) \mathcal{A}_i f \right)^2 \\ &= \sum_{i,j=1}^3 \left(D_i^{1/2} c^j \mathcal{A}_{ij}|_g f \right)^2 \\ &= \|MH\hat{\mathbf{c}}\|_{l^2}^2, \end{aligned} \tag{1.23}$$

where we let

$$M := \begin{pmatrix} D_S^{1/2} & & \\ & \zeta D_S^{1/2} & \\ & & D_A^{1/2} \end{pmatrix} \quad \text{and} \quad H := (\mathcal{A}_{ij}|_g f)_{ij}. \tag{1.24}$$

Assuming $\zeta > 0$ we see that M is invertible and that $\|\mathbf{c}\|_{\mathcal{G}_\zeta} = \|M^{-1}\hat{\mathbf{c}}\|_{l^2}$. In the sub-Riemannian case we have the same relation by noting that $c^2 \equiv 0$ and so we modify our definitions:

$$\hat{\mathbf{c}} := (c^1, c^3)^T, \quad M := \begin{pmatrix} D_S^{1/2} & \\ & D_A^{1/2} \end{pmatrix} \quad \text{and} \quad H := \begin{pmatrix} \mathcal{A}_{11}|_g f & \mathcal{A}_{13}|_g f \\ \mathcal{A}_{31}|_g f & \mathcal{A}_{33}|_g f \end{pmatrix}. \tag{1.25}$$

Using these modifications we again have $\|\mathbf{c}\|_{\mathcal{G}_0} = \|M^{-1}\hat{\mathbf{c}}\|_{l^2}$ and (1.23) holds as well. Using these relations we can formulate our minimization problem in the Euler-Lagrange formalism.

³We would like to give a geometric formulation for this vector but were not able to so far.

The Lagrangian of this minimization problem is given by:

$$L(\hat{\mathbf{c}}, \lambda) = \|MH\hat{\mathbf{c}}\|_{l^2}^2 + \lambda \left(1 - \|\mathbf{c}\|_{\mathcal{G}_f}^2\right) = \|MH\hat{\mathbf{c}}\|_{l^2}^2 + \lambda \left(1 - \|M^{-1}\hat{\mathbf{c}}\|_{l^2}^2\right). \quad (1.26)$$

For which we obtain the derivative with respect to $\hat{\mathbf{c}}$:

$$\begin{aligned} \nabla_{\hat{\mathbf{c}}} L(\hat{\mathbf{c}}, \lambda) &= \nabla_{\hat{\mathbf{c}}} \|MH\hat{\mathbf{c}}\|_{l^2}^2 - \lambda \nabla_{\hat{\mathbf{c}}} \|M^{-1}\hat{\mathbf{c}}\|_{l^2}^2 \\ &= 2(MH)^T(MH)\hat{\mathbf{c}} - 2\lambda M^{-2}\hat{\mathbf{c}}. \end{aligned}$$

Which vanishes when:

$$(MH)^T(MH)\hat{\mathbf{c}} = \lambda M^{-2}\hat{\mathbf{c}} \iff H^T M^2 H \hat{\mathbf{c}} = \lambda M^{-2}\hat{\mathbf{c}} \iff M^2 H^T M^2 H \hat{\mathbf{c}} = \lambda \hat{\mathbf{c}}. \quad (1.27)$$

Which proves the following statement.

Proposition 1.48. $\mathcal{B}_1|_g$ expressed in terms of the left invariant basis $(\mathcal{A}_i)_{i=1}^3$ and metric $\mathcal{G}_f$ is a unit eigenvector of the matrix $(M^2 H^T M^2 H)|_g$, where M and H are defined as (1.24) if $\zeta > 0$ and (1.25) if $\zeta = 0$.

We can be sure that we will obtain real eigenvalues and eigenvectors by rewriting (1.27) as follows:

$$\begin{aligned} M^2 H^T M^2 H \hat{\mathbf{c}} = \lambda \hat{\mathbf{c}} &\iff MH^T M^2 H (MM^{-1}) \hat{\mathbf{c}} = \lambda M^{-1} \hat{\mathbf{c}} \\ &\iff (MHM)^T (MHM) (M^{-1} \hat{\mathbf{c}}) = \lambda M^{-1} \hat{\mathbf{c}}, \end{aligned} \quad (1.28)$$

and since $(MHM)^T (MHM)$ is symmetric λ is real.

What eigenvector we choose if there are multiple is up to us, as a rule we take the one with the smallest absolute eigenvalue. Having determined an optimal $\hat{\mathbf{c}}$ we bring it back to the tangent space:

$$\mathcal{B}_1|_g = c^1 \mathcal{A}_1|_g + c^2 \mathcal{A}_2|_g + c^3 \mathcal{A}_3|_g. \quad (1.29)$$

Remark 1.49. As a means of making the method more robust with regards to noise it can be beneficial to apply some component-wise Gaussian blur to $g \mapsto (M^2 H^T M^2 H)|_g$ before calculating eigenvectors. This way the data and consequently the gauge frames will not make abrupt movements as we travel along the manifold.

Having fixed $\mathcal{B}_1$ we still have some freedom in fixing $\mathcal{B}_2$ and $\mathcal{B}_3$, since we choose $\mathcal{B}_1$ to be the vector aligned to the data the remaining choices are somewhat arbitrary. For consistency we will pick the remaining vectors such that if $\mathcal{B}_1|_g = \sqrt{D_S} \mathcal{A}_1|_g$ then also $\mathcal{B}_2|_g = \zeta \sqrt{D_S} \mathcal{A}_2|_g$ and $\mathcal{B}_3|_g = \sqrt{D_A} \mathcal{A}_3|_g$ (i.e. all the gauge basis vectors are normalized left invariant basis vectors).

The way we accomplish this (which is not unique, see also [3, Ch. 3]) is by performing the same spatial rotation on the second and third basis vector as would need to be performed to make $\mathcal{A}_1$ colinear with $\mathcal{B}_1$. If $\mathcal{B}_i|_g = \sum_{j=1}^3 b_i^j \mathcal{A}_j|_g$ and the rotation-matrix R_g at g is such that:

$$\begin{pmatrix} b_1^1 \\ b_1^2 \\ b_1^3 \end{pmatrix} = \sqrt{D_S} R_g \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad (1.30)$$

then we choose the coefficients for $\mathcal{B}_2$ and $\mathcal{B}_3$ as follows:

$$\begin{pmatrix} b_2^1 \\ b_2^2 \\ b_2^3 \end{pmatrix} = \zeta \sqrt{D_S} R_g \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \quad \begin{pmatrix} b_3^1 \\ b_3^2 \\ b_3^3 \end{pmatrix} = \sqrt{D_A} R_g \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}. \quad (1.31)$$

Per usual we define the dual basis vectors by their action on the basis vectors.

Definition 1.50 (Dual Gauge Vectors). Let $\beta^i \in T^*\mathbb{M}_2$ such that:

$$\beta^i(\mathcal{B}_j) = \delta_j^i. \quad (1.32)$$

1.2.3 Metric, Gradient and Divergence

On this new data aligned frame $(\mathcal{B}_i)_{i=1}^n$ we place the following metric.

Definition 1.51 (Gauge frame metric).

$$\mathcal{G}_\zeta^{\text{gauge}} = \beta^1 \otimes \beta^1 + \beta^2 \otimes \beta^2 + \beta^3 \otimes \beta^3,$$

Or in matrix form

$$\mathcal{G}_\zeta^{\text{gauge}} \simeq \begin{pmatrix} 1 & & \\ & 1 & \\ & & 1 \end{pmatrix}, \quad (1.33)$$

for $\zeta > 0$ and for the sub-Riemannian case $\zeta = 0$ we have

$$\mathcal{G}_0^{\text{gauge}} = \beta^1 \otimes \beta^1 + \beta^3 \otimes \beta^3 \quad \text{and} \quad \mathcal{G}_0^{\text{gauge}} \simeq \begin{pmatrix} 1 & \\ & 1 \end{pmatrix}.$$

Note how in the gauge frame the metric does not explicitly depend on f anymore since we already included it in scaling of the new basis vectors, however we still need to carefully distinguish between $\zeta \neq 0$ and $\zeta = 0$.

In the same manner as for the left invariant frame we determine the gradient:

$$\nabla_\zeta^{\text{gauge}} f = \begin{cases} (\mathcal{B}_1 f) \mathcal{B}_1 + (\mathcal{B}_2 f) \mathcal{B}_2 + (\mathcal{B}_3 f) \mathcal{B}_3 & \text{if } f\zeta > 0, \\ (\mathcal{B}_1 f) \mathcal{B}_1 + (\mathcal{B}_3 f) \mathcal{B}_3 & \text{if } \zeta = 0. \end{cases} \quad (1.34)$$

The new divergence for a vector field $\mathbf{v}(g) = \sum_{i=1}^3 v^i(g) \mathcal{B}_i|_g$ is:

$$\text{div } \mathbf{v}(g) = \sum_{i=1}^3 \mathcal{B}_i|_g v^i, \quad (1.35)$$

or sub-Riemannian:

$$\text{div } \mathbf{v}(g) = \mathcal{B}_1|_g v^1 + \mathcal{B}_3|_g v^3, \quad (1.36)$$

With both the left-invariant and gauge frames in hand the PDE techniques that we will develop in the later sections will have two underlying geometries on which to act.

1.3 Orientation Scores

We now turn to the method with which we *lift* an image, represented as a function f on $\mathbb{R}^2$, to a function U_f on $\mathbb{M}_2$ so that we can work on it with the richer tools of the roto-translation manifold. We accomplish this via the *Orientation Score Transform* and refer to the obtained U_f as the *orientation score* of f , where we interpret $U_f(\mathbf{x}, \mathbf{n})$ as how strongly the orientation $\mathbf{n}$ is present at location $\mathbf{x}$ in the image.

This means that denoising an image will consist of applying the orientation score transform, doing some processing on F and finally take the inverse transform to get back an enhanced image. This process is illustrated in figure 1.8.

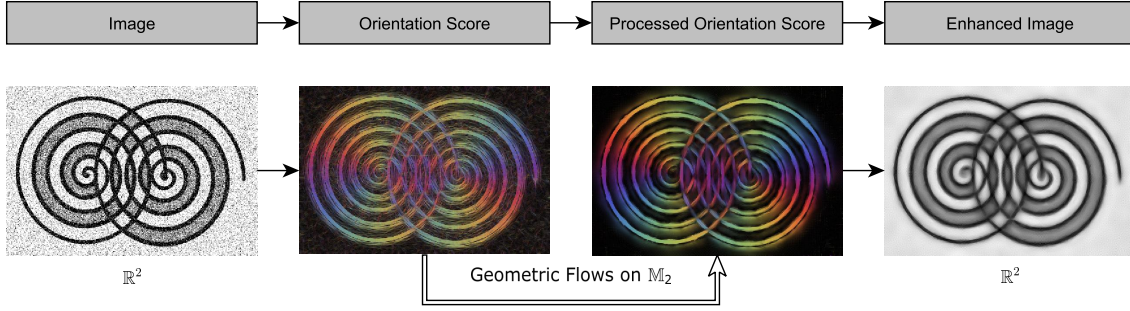


Figure 1.8: Starting from an image as a function on $\mathbb{R}^2$ we *lift* it to $\mathbb{M}_2$ via the orientation score transform. Processing is done in $\mathbb{M}_2$ before inverting the orientation score transform to yield an enhanced image.

1.3.1 Invertible Transformation

In practice image functions will be compactly supported so we can restrict ourselves to $f : \Omega_f \rightarrow \mathbb{R}$ where Ω_f is a compact subset of $\mathbb{R}^2$. Images might also be in color in which case we require several channels, however we will restrict ourselves to single channel images in this chapter since the techniques are easily extended from one to multiple channels by applying them to each channel separately.

Lifting f to $\mathbb{M}_2$ happens through the following transformation:

Definition 1.52 (Orientation Score Transform (OST)). Let $f \in L^2(\mathbb{R}^2)$ with compact support in Ω_f and $g = (\mathbf{x}, \mathbf{n}) \in \mathbb{M}_2$, then:

$$U_f(g) := \mathcal{W}_\psi f(g) := (\mathcal{U}_g \psi, f)_{L^2(\mathbb{R}^2)} = \int_{\Omega_f} \overline{\psi(R_{\mathbf{n}}^{-1}(\mathbf{x} - \mathbf{y}))} f(\mathbf{y}) d\mathbf{y},$$

where $\psi \in L_c^2(\mathbb{M}_2, \mathbb{R})$ is a compactly supported anisotropic filter and $\mathcal{U}_g$ is the left regular representation per definition 1.19.

Observe that by this definition the following holds.

Corollary 1.53.

$$\mathcal{W}_\psi \circ \mathcal{U}_g = \mathcal{L}_g \circ \mathcal{W}_\psi,$$

which is the reason for our interest in left-invariance, we do not want our image processing to act any different on the same image that has been rotated and/or translated, demanding that properties like the one above hold ensures this.

For our application we will be using *cake wavelets* for ψ , so called for their resemblance to pieces of cake in the Fourier domain, see figure 1.9 for an example. More details on this transformation and its invertibility can be found in [24, 2].

Remark 1.54. With our choice of filters we generate an orientation score that is symmetric under change of direction, i.e. $U_f((\mathbf{x}, \mathbf{n})) = U_f((\mathbf{x}, -\mathbf{n}))$. Depending on the application it might be desirable to use filters that are directionally sensitive, again see [24].

1.3.2 Visualization

Orientation scores, being higher dimensional, are challenging to visualize on paper. For the purpose of this report, where we restrict ourselves to 2-dimensional grayscale images, we will visualize orientation scores with what we call a *orientation rainbow* plot.

The idea is that we assign to every pixel a color based on the orientation score as follows:

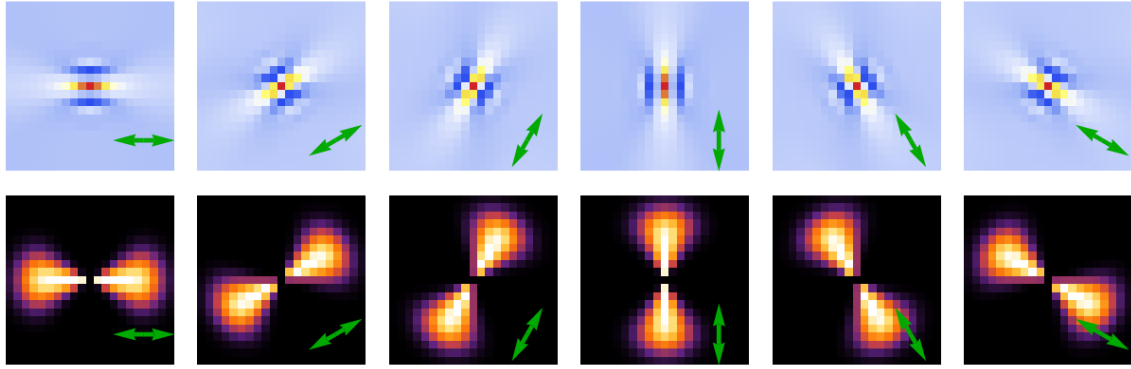


Figure 1.9: Example 21×21 discrete cake wavelets for 6 orientations, the green arrows indicate the orientation sensitivity. Top: kernels in the spatial domain, bottom: absolute values of kernels in the Fourier domain.

- brightness is controlled by the average absolute intensity over all orientations,
- hue corresponds to orientation, where we mix hues based on the absolute intensities present at that spatial location.

An example of what this looks like can be seen in figure 1.10.

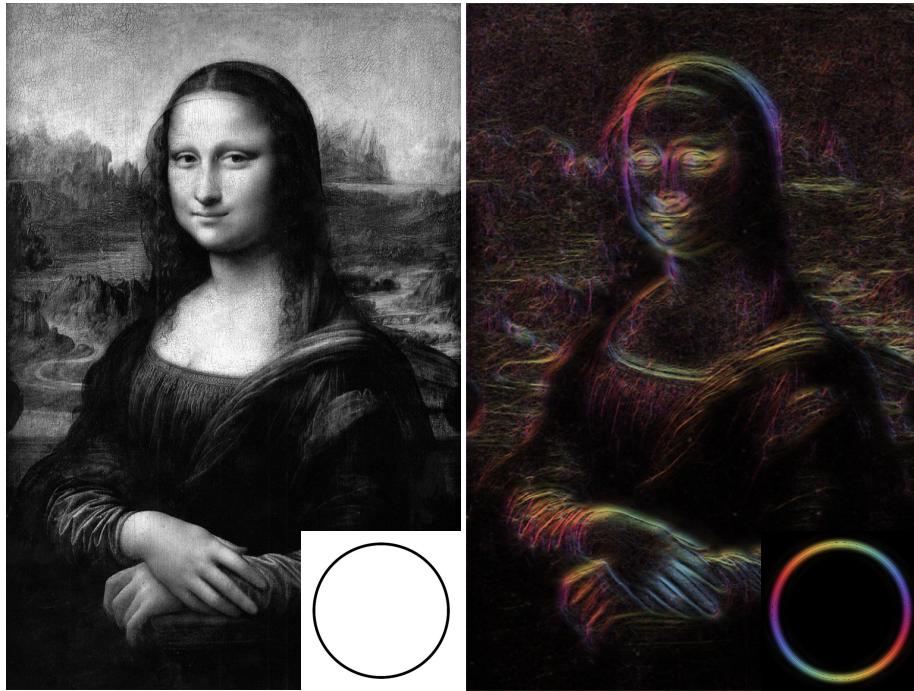


Figure 1.10: Illustrating *rainbow* visualization of orientation scores, left: source image, right: orientation score. Note how the hue changes on the circle as we go around it and how flat areas translate to black (the orientation score vanishes).

1.3.3 Orientation Confidence & Isotropy Factor

A useful transformation of an orientation score is the orientation confidence.

In $\mathbb{R}^3$ orientation confidence is often calculated by the Laplacian in the subspace orthogonal to the line structure, we can take a similar approach in $\mathbb{M}_2$ by taking the Laplacian in the space

spanned by $(\mathcal{A}_2, \mathcal{A}_3)$, as $\mathcal{A}_1$ is implicitly aligned with the local line structure, in a gauge frame $\mathcal{B}_1$ is explicitly aligned with the line structure and so we take the Laplacian in the span of $(\mathcal{B}_2, \mathcal{B}_3)$.

In the sub-Riemannian case this just reduces to the second derivatives in the 1-dimensional spaces spanned by $\mathcal{A}_3$ and $\mathcal{B}_3$ respectively. With that in mind we define orientation confidence in $\mathbb{M}_2$ as:

Definition 1.55 (Orientation confidence).

$$\begin{aligned} C_f(g) &:= \left| \zeta^2 D_S \mathcal{A}_{22} \Big|_g U_f + D_A \mathcal{A}_{33} \Big|_g U_f \right|, & \text{(using the left invariant frame)} \\ &:= \left| \mathcal{B}_{22} \Big|_g U_f + \mathcal{B}_{33} \Big|_g U_f \right|. & \text{(using the gauge frame)} \end{aligned}$$

The orientation confidence gives a number in the range $[0, \infty)$ which does not immediately suite our practical needs, for that reason we transform it as follows.

Definition 1.56 (Isotropy factor). Let $c > 0$ be a chosen scaling constant, then the isotropy factor is defined as⁴:

$$\alpha_f(g) := e^{-\frac{C_f(g)}{c}}.$$

What is convenient about this quantity is that it gives a number in the range $(0, 1]$ with a number close to zero indicating a high degree of anisotropy and a 1 indicating perfect isotropy, this is the quantity that we will use to steer diffusion. For an example in the left invariant frame take a look at figure 1.11.

The choice of c controls how steep the decline of the isotropy factor is, its appropriate value depends on the application and on exactly how the data is represented numerically and is best determined heuristically. For our applications we have used $c = 0.2$.

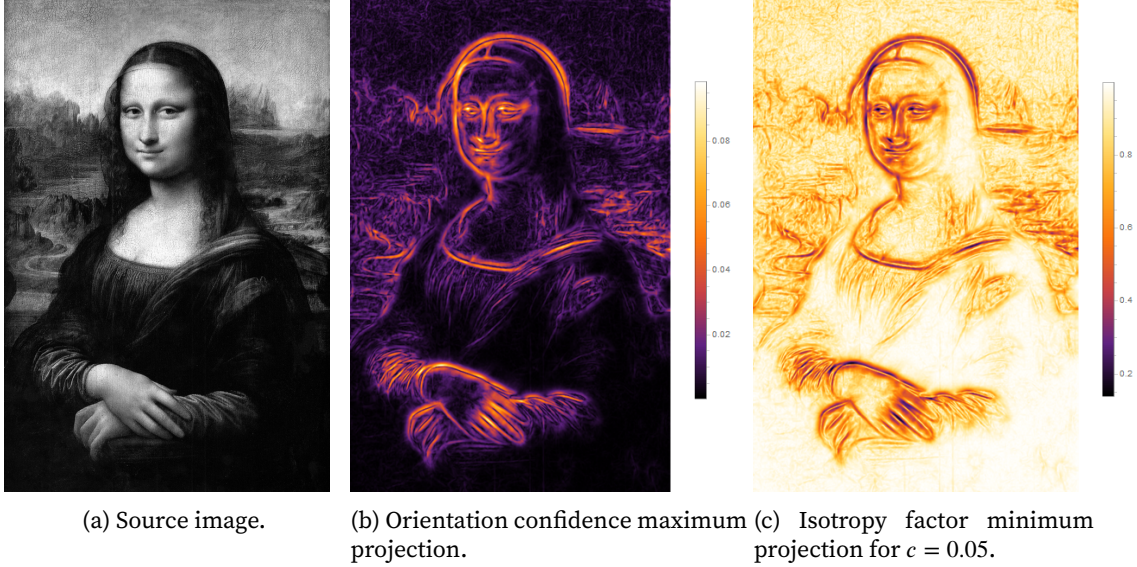


Figure 1.11: Demonstrating orientation confidence in the left invariant frame where we have set $D_S = D_A = \zeta = 1$ for the purpose of illustration.

⁴In other works[23] this quantity is called D_{bb} .

Chapter 2

Image Denoising

2.1 Introduction

This chapter contains an extended reporting of the work we did in the context of the 2019 *SSVM* conference [17]. The primary application in that paper was enhancement of Diffusion-Weighted Magnetic Resonance Imaging, our contribution was adapting the described methods for the enhancement of conventional 2-dimensional images.

We will reprise some of the theory in [17] from the 2-dimensional point of view and present a greater amount of experiments then could be included in the paper.

All formulations in this chapter are coordinate independent and agnostic to our choice of metric (unless expressly specified). If we choose to work in the left invariant frame we take $\mathcal{G}_\zeta \equiv \mathcal{G}_\zeta^{\text{left}}$ and if we choose to work in the gauge frame we take $\mathcal{G}_\zeta \equiv \mathcal{G}_\zeta^{\text{gauge}}$. All the other derived objects such as gradient and divergence are then relative to that metric and/or frame. We summarize how the symbols in this chapter should be interpreted relative to this choice in the following table.

Choice of geometry	Left invariant	Gauge
Frame	$\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3$	$\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3$
Co-frame	$\omega^1, \omega^2, \omega^3$	$\beta^1, \beta^2, \beta^3$
Metric tensor $\mathcal{G}_\zeta$	$\mathcal{G}_\zeta^{\text{left}}$ (def. 1.29)	$\mathcal{G}_\zeta^{\text{gauge}}$ (def. 1.51)
Gradient ∇_ζ	$\nabla_\zeta^{\text{left}}$ (1.15)	$\nabla_\zeta^{\text{gauge}}$ (1.34)
Divergence $\text{div}(\mathbf{v})$	$\sum_{i=1}^3 \mathcal{A}_i v^i$ (1.18)	$\sum_{i=1}^3 \mathcal{B}_i v^i$ (1.35)
Norm $\ \cdot\ _{\mathcal{G}_\zeta}$	$\ \cdot\ _{\mathcal{G}_\zeta^{\text{left}}}$	$\ \cdot\ _{\mathcal{G}_\zeta^{\text{gauge}}}$
Volume form $d\mu$	$d\mu^{\text{left}}$ (corollary 1.34)	$\beta^1 \wedge \beta^2 \wedge \beta^3$

Table 2.1: Definition of symbols used in this chapter relative to the choice of underlying geometry, relevant definitions and equations included in parentheses.

2.2 Coherence Enhancing Diffusion

To provide a benchmark against which we can compare our new methods we first introduce the state-of-art method *Coherence Enhancing Diffusion* which steers diffusion based on orientation confidence[23].

The idea here being that we use the isotropy factor α_f from definition 1.56 to apply isotropic diffusion where the data is isotropic and apply anisotropic diffusion where the data is anisotropic. Let D_S, D_A and ζ define the metric per definition 1.29.

Definition 2.1 (Coherence Enhancing Diffusion (left invariant)). Given $U \in BV(\Omega)$ with $\Omega \overset{\text{ap}}{\subset} \mathbb{M}_2$ (an aperture per definition 1.41) then we call $W : \Omega \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ obeying

$$\begin{cases} \frac{\partial W}{\partial t}(g, t) = D_S \mathcal{A}_{11} W(\cdot, t) + \zeta^2 D_S \mathcal{A}_2 (\alpha_f \mathcal{A}_2 W(\cdot, t)) + D_A \mathcal{A}_3 (\alpha_f \mathcal{A}_3 W(\cdot, t)) & t \geq 0, \\ W(g, 0) = U(g) & g \in \Omega, \end{cases}$$

coherence enhancing diffusion on the left invariant frame started at U .

Observe that W is smooth since it is the solution to a parabolic PDE with differentiable coefficients. Or if we are working in the gauge frame:

Definition 2.2 (Coherence Enhancing Diffusion (gauge)). Given $U \in BV(\Omega)$ with $\Omega \overset{\text{ap}}{\subset} \mathbb{M}_2$ then we call $W : \Omega \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ obeying

$$\begin{cases} \frac{\partial W}{\partial t}(g, t) = \mathcal{B}_{11} W(\cdot, t) + \mathcal{B}_2 (\alpha_f \mathcal{B}_2 W(\cdot, t)) + \mathcal{B}_3 (\alpha_f \mathcal{B}_3 W(\cdot, t)) & t \geq 0, \\ W(g, 0) = U(g) & g \in \Omega, \end{cases}$$

coherence enhancing diffusion on the gauge frame started at U .

2.3 Total Variation Flow

So far most of what we have constructed relies on functions being at least continuously differentiable. And while in practice images being discrete means they can be approximated arbitrarily close by smooth functions we might want to include discontinuities as a means to model step-edges without losing too much in terms of differentiation. For that reason we turn to functions of bounded variation.

Intertwined with such functions is the concept of *total variation* itself or within the context of $\mathbb{M}_2$: *total roto-translation variation*. Total variation relates to noisy images in the sense that adding noise to an image function increases its total variation. This gives us an avenue of attack in denoising images by decreasing their total variation, an approach that has been applied successfully to 2D images [25, 26] and has been extended to lifted images [5]. The novelty of our approach in [17] is formulating a more general PDE approach that works equally well in higher dimensions.

2.3.1 Weak Formulation

We start with the function space L^1 so that integrals exists, for these functions we can calculate their total variation.

Definition 2.3 (Total variation). Let $U \in L^1(\Omega)$ then:

$$TV(U, \Omega) := \sup \left\{ \int_{\Omega} U(g) \operatorname{div} \mathbf{v}(g) d\mu(g) \mid \mathbf{v} \text{ a } C_0^1 \text{ vector field} : \forall g \in \Omega : \|\mathbf{v}\|_{\mathcal{G}_\varepsilon} \leq 1 \right\}.$$

Recall that this formulation depends on our choice of metric, either we go with the left invariant frames and use definition 1.29 or we go with the gauge frames and use definition 1.51.

We now restrict ourselves to those L^1 functions that have finite total variation and call these functions of bounded variation.

Definition 2.4 (Space of functions of bounded variation).

$$BV(\Omega) := \{U \in L^1(\Omega) \mid TV(U, \Omega) < \infty\}.$$

For the purpose we have in mind we will amend the previous definitions, starting with the space over which we take the supremum that we have not explicitly named above.

Definition 2.5 (Regularized smoothing space). Let Ω be a compact subset of $\mathbb{M}_2$, $C_0^\infty(\Omega)$ the space of smooth functions and $\chi_0(\Omega)$ the space of smooth vector fields both zero on the boundary then:

$$Sm(\Omega) := \left\{ (\psi, \mathbf{v}) \in C_0^\infty(\Omega) \times \chi_0(\Omega) \mid \forall g \in \Omega : \|\mathbf{v}(g)\|_{\mathfrak{g}_k}^2 + |\psi(g)|^2 \leq 1 \right\}.$$

Definition 2.6 (Total roto-translation variation). Let $U \in BV(\Omega)$ and $\varepsilon \geq 0$ then:

$$TV_\varepsilon(U, \Omega) := \sup_{(\psi, \mathbf{v}) \in Sm(\Omega)} \int_{\Omega} \varepsilon \psi(g) + U(g) \operatorname{div} \mathbf{v}(g) \, d\mu(g).$$

The key idea here is that we bound TV_ε from below with the inclusion of the $\varepsilon\psi$ term, we can see that if we take a vector field with zero divergence and $\psi \equiv 1$ then the integral evaluates to $\varepsilon\mu(\Omega)$. From which we conclude the following corollary.

Corollary 2.7.

$$0 < \varepsilon\mu(\Omega) \leq TV_\varepsilon(U, \Omega),$$

Additionally if $U \in BV(\Omega)$:

$$0 \leq TV_0(U, \Omega) \leq TV_\varepsilon(U, \Omega) \leq TV_0(U, \Omega) + \varepsilon\mu(\Omega).$$

When we set $\varepsilon = 0$ we trivially recover the classical notion of total variation.

2.3.2 Strong Formulation

We can now go back and take a U that is continuously differentiable and see how to interpret total variation for more well behaved functions.

Lemma 2.8. With $\Omega \overset{\text{ap.}}{\subset} \mathbb{M}_2$ let $U \in C^1(\Omega, \mathbb{R})$ and $\zeta, \varepsilon > 0$ then we have:

$$TV_\varepsilon(U, \Omega) = \int_{\Omega} \sqrt{\|\nabla_\zeta U(g)\|_{\mathfrak{g}_\zeta}^2 + \varepsilon^2} \, d\mu(g).$$

Proof.

$$\begin{aligned} TV_\varepsilon(U, \Omega) &= \sup_{(\psi, \mathbf{v}) \in Sm(\Omega)} \int_{\Omega} \varepsilon \psi(g) + U(g) \operatorname{div} \mathbf{v}(g) \, d\mu(g) \\ &= \sup_{(\psi, \mathbf{v}) \in Sm(\Omega)} \int_{\Omega} \varepsilon \phi(g) - \mathbf{v}(g)U \, d\mu(g) + \int_{\Omega} \operatorname{div}(U\mathbf{v})(g) \, d\mu(g) \quad (\text{lemma 1.40}) \end{aligned}$$

We can apply Gauss's theorem since the boundary is piecewise smooth and $U\mathbf{v}$ is C^1 , the resulting term is zero since $\mathbf{v}$ vanishes at the boundary.

$$\begin{aligned} &= \sup_{(\psi, \mathbf{v}) \in Sm(\Omega)} \int_{\Omega} \varepsilon \phi(g) - \mathbf{v}(g)U \, d\mu(g) + \oint_{\partial\Omega} (U(g)\mathbf{v}(g), \mathbf{N}(g))_{\mathfrak{g}_\zeta} \, d\sigma(g) \\ &= \sup_{(\psi, \mathbf{v}) \in Sm(\Omega)} \int_{\Omega} \varepsilon \phi(g) - (\nabla_\zeta U(g), \mathbf{v}(g))_{\mathfrak{g}_\zeta} \, d\mu(g) \end{aligned}$$

We can look at the integrand as an inner product in the space $\mathbb{R} \times T_g \mathbb{M}_2$ defined as follows: $\langle (a_1, \mathbf{v}_1), (a_2, \mathbf{v}_2) \rangle_{\mathbb{R} \times T_g \mathbb{M}_2} := a_1 a_2 + \langle \mathbf{v}_1, \mathbf{v}_2 \rangle_{\mathfrak{g}_\zeta}$, we rewrite:

$$= \sup_{(\psi, \mathbf{v}) \in Sm(\Omega)} \int_{\Omega} ((\varepsilon, \nabla_{\zeta} U), (\psi(g), -\mathbf{v}(g)))_{\mathbb{R} \times T_g \mathbb{M}_2} d\mu(g).$$

From the *Cauchy-Schwarz* inequality we know that to maximize the integrand we need to select $(\psi, -\mathbf{v})$ as being collinear with $(\varepsilon, \nabla_{\zeta} U)$ while $\|\mathbf{v}(g)\|_{\mathfrak{g}_\zeta}^2 + |\psi(g)|^2 = 1$, hence the C^1 optimal would be:

$$(\phi(g), -\mathbf{v}(g)) = \frac{1}{\sqrt{\|\nabla_{\zeta} U(g)\|_{\mathfrak{g}_\zeta}^2 + \varepsilon^2}} (\varepsilon, \nabla_{\zeta} U(g)).$$

Since ψ and $\mathbf{v}$ are smooth they can arbitrarily closely approximate the optimal above so substituting this C^1 optimum gives exactly the supremum:

$$\begin{aligned} TV_{\varepsilon}(U, \Omega) &= \int_{\Omega} \frac{((\varepsilon, \nabla_{\zeta} U(g)), (\varepsilon, \nabla_{\zeta} U(g)))_{\mathbb{R} \times T_g \mathbb{M}_2}}{\sqrt{\|\nabla_{\zeta} U(g)\|_{\mathfrak{g}_\zeta}^2 + \varepsilon^2}} d\mu(g) \\ &= \int_{\Omega} \frac{\varepsilon^2 + (\nabla_{\zeta} U(g), \nabla_{\zeta} U(g))_{\mathfrak{g}_\zeta}}{\sqrt{\|\nabla_{\zeta} U(g)\|_{\mathfrak{g}_\zeta}^2 + \varepsilon^2}} d\mu(g) \\ &= \int_{\Omega} \frac{\|\nabla_{\zeta} U(g)\|_{\mathfrak{g}_\zeta}^2 + \varepsilon^2}{\sqrt{\|\nabla_{\zeta} U(g)\|_{\mathfrak{g}_\zeta}^2 + \varepsilon^2}} d\mu(g) \\ &= \int_{\Omega} \sqrt{\|\nabla_{\zeta} U(g)\|_{\mathfrak{g}_\zeta}^2 + \varepsilon^2} d\mu(g). \end{aligned}$$

■

From this lemma we can see how total variation measures how much a C^1 function changes by integrating the norm of the gradient over the domain. The reason we added the ε term also becomes clearer as it keeps the integrand strictly positive at every $g \in \mathbb{M}_2$ and so makes dividing by this quantity well defined even when the gradient vanishes at some point.

Now looking at our modified total variation as a quantity to optimize, or at least reduce, we can formulate a variational problem. In order to establish an optimization algorithm the first variation in particular will be of importance.

Lemma 2.9 (First variation of TV). *With $\Omega \overset{\text{ap.}}{\subset} \mathbb{M}_2$ let $U \in C^2(\Omega)$ and $\zeta, \varepsilon > 0$, then:*

$$\partial TV_{\varepsilon}(U) = -\operatorname{div} \left(\frac{\nabla_{\zeta} U}{\sqrt{\|\nabla_{\zeta} U\|_{\mathfrak{g}_\zeta}^2 + \varepsilon^2}} \right).$$

Proof. Let δ be a smooth function vanishing at the boundary on which we apply an equivalent

formulation of the first variation:

$$\begin{aligned}
\langle \partial TV_\epsilon(U), \delta \rangle_{L^2(\Omega)} &= \lim_{h \downarrow 0} \frac{TV_\epsilon(U + h\delta) - TV_\epsilon(U)}{h} \\
&= \lim_{h \downarrow 0} \frac{1}{h} \int_{\Omega} \sqrt{\|\nabla_{\zeta}(U + h\delta)(g)\|_{\mathfrak{G}_{\zeta}}^2 + \epsilon^2} \\
&\quad - \sqrt{\|\nabla_{\zeta}U(g)\|_{\mathfrak{G}_{\zeta}}^2 + \epsilon^2} d\mu(g) \\
&= \lim_{h \downarrow 0} \frac{1}{h} \int_{\Omega} \sqrt{\|\nabla_{\zeta}U(g)\|_{\mathfrak{G}_{\zeta}}^2 + 2h (\nabla_{\zeta}U(g), \nabla_{\zeta}\delta(g))_{\mathfrak{G}_{\zeta}} + h^2 \|\nabla_{\zeta}\delta(g)\|_{\mathfrak{G}_{\zeta}}^2 + \epsilon^2} \\
&\quad - \sqrt{\|\nabla_{\zeta}U(g)\|_{\mathfrak{G}_{\zeta}}^2 + \epsilon^2} d\mu(g)
\end{aligned}$$

(Taylor expansion of the integrand around $h = 0$)

$$\begin{aligned}
&= \lim_{h \downarrow 0} \frac{1}{h} \int_{\Omega} \frac{h (\nabla_{\zeta}U(g), \nabla_{\zeta}\delta(g))_{\mathfrak{G}_{\zeta}}}{\sqrt{\|\nabla_{\zeta}U(g)\|_{\mathfrak{G}_{\zeta}}^2 + \epsilon^2}} + O(h^2) d\mu(g) \\
&= \int_{\Omega} \left(\frac{\nabla_{\zeta}U(g)}{\sqrt{\|\nabla_{\zeta}U(g)\|_{\mathfrak{G}_{\zeta}}^2 + \epsilon^2}}, \nabla_{\zeta}\delta(g) \right)_{\mathfrak{G}_{\zeta}} d\mu(g)
\end{aligned}$$

(lemma 1.40)

$$= \int_{\Omega} \operatorname{div} \left(\frac{\delta \nabla_{\zeta}U}{\sqrt{\|\nabla_{\zeta}U\|_{\mathfrak{G}_{\zeta}}^2 + \epsilon^2}} \right) (g) - \delta(g) \operatorname{div} \left(\frac{\nabla_{\zeta}U}{\sqrt{\|\nabla_{\zeta}U\|_{\mathfrak{G}_{\zeta}}^2 + \epsilon^2}} \right) (g) d\mu(g)$$

(Gauss's theorem on the first term which then vanishes since $\delta = 0$ on the boundary)

$$\begin{aligned}
&= \oint_{\partial\Omega} \frac{\delta(g) (\nabla_{\zeta}U(g), \mathbf{N}(g))_{\mathfrak{G}_{\zeta}}}{\sqrt{\|\nabla_{\zeta}U(g)\|_{\mathfrak{G}_{\zeta}}^2 + \epsilon^2}} d\sigma(g) - \int_{\Omega} \delta(g) \operatorname{div} \left(\frac{\nabla_{\zeta}U}{\sqrt{\|\nabla_{\zeta}U\|_{\mathfrak{G}_{\zeta}}^2 + \epsilon^2}} \right) (g) d\mu(g) \\
&= - \int_{\Omega} \delta(g) \operatorname{div} \left(\frac{\nabla_{\zeta}U}{\sqrt{\|\nabla_{\zeta}U\|_{\mathfrak{G}_{\zeta}}^2 + \epsilon^2}} \right) (g) d\mu(g) \\
&= \left(- \operatorname{div} \left(\frac{\nabla_{\zeta}U}{\sqrt{\|\nabla_{\zeta}U\|_{\mathfrak{G}_{\zeta}}^2 + \epsilon^2}} \right), \delta \right)_{L^2(\Omega)}
\end{aligned}$$

■

The first variation of TV does not scale with the function, indeed let $\lambda > 0$ then $TV_0(\lambda U) = TV_0(U)$. At first sight this isn't a desirable property for an image processing tool since we would like a method to be invariant under monotonic codomain transformations. There is however also an opportunity here vis-à-vis stability and time-scaling that we will encounter later.

2.3.3 Enhancement PDE

With the results of the previous sections in hand we can formulate an evolution equation that starts from the given function and reduces its total variation as time progresses.

Definition 2.10 (Total Variation Flow). Given $U \in BV(\Omega)$ with $\Omega \stackrel{\text{ap}}{\subset} \mathbb{M}_2$ and $\varepsilon, \zeta > 0$ then we call $W^\varepsilon : \mathbb{M}_2 \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ (being smooth) obeying

$$\begin{cases} \frac{\partial W^\varepsilon}{\partial t}(g, t) = \operatorname{div} \left(\frac{\nabla_\zeta W^\varepsilon(\cdot, t)}{\sqrt{\|\nabla_\zeta W^\varepsilon(\cdot, t)\|_{\mathfrak{G}_\zeta}^2 + \varepsilon^2}} \right)(g) & t \geq 0, \\ \nabla_{\mathbb{R}^2} W^\varepsilon(g, 0) \cdot \mathbf{N}(\mathbf{x}) = 0 & g = (\mathbf{x}, \mathbf{n}) \in \partial\Omega, \\ W^\varepsilon(g, 0) = U(g) & g \in \Omega. \end{cases} \quad (2.1)$$

the total variation flow started at U , or equivalently the gradient flow of TV_ε started at U . Where we use Neumann boundary conditions with $\mathbf{N}$ as the normal to the spatial boundary.

Given that we have defined the gradient and divergence in terms of the left-invariant differential operators $(A_i)_{i=1}^3$ we see that our formulation of TVF is equivariant under group action, it does not matter where we locate the data in the manifold, the resulting flow is the same only at a different location. Or formally for all $g \in \mathbb{M}_2$ let W^ε be the flow started at U and W_g^ε the flow started at $\mathcal{L}_g U$, then:

$$\mathcal{L}_g W^\varepsilon(\cdot, t) = W_g^\varepsilon(\cdot, t). \quad (2.2)$$

The next result serves to bound the effect of our introduction of ε as a regularization mechanism and assures us that our modifications do not qualitatively alter the gradient flow.

Theorem 2.11 (Strong L_2 -convergence). Let $U \in BV(\Omega)$ and let W^ε be the gradient flow of TV_ε started at U and $\zeta, \varepsilon \geq 0$. For $t \geq 0$ we have

$$W^\varepsilon(\cdot, t) \xrightarrow[\mathcal{L}_2(\Omega)]{\varepsilon \downarrow 0} W^0(\cdot, t).$$

And furthermore

$$\|W^\varepsilon(\cdot, t) - W^0(\cdot, t)\|_{L^2(\Omega)} \leq 16 (\|U\|_{L^2(\Omega)} (TV_0(U) + \delta) \delta t^2)^{1/5} = O\left(\sqrt[5]{\varepsilon t^2}\right),$$

where $\delta = \varepsilon \mu(\Omega)$.

Proof. We aim to satisfy the conditions of [17, Thm. 2] when we let $H = L^2(\Omega)$, $F = TV_0$ and $G = TV_\varepsilon$.

- Clearly $L^2(\Omega)$ is a Hilbert space.
- F and G are proper since total variation is non-negative and not infinite for all $L^2(\Omega)$ functions.
- From corollary 2.7 it follows that: $\forall u \in H : |F(u) - G(u)| < \varepsilon \mu(\Omega) = \delta$.
- F and G are convex. Let $\lambda \in (0, 1)$ and $u_1, u_2 \in H$, then

$$\begin{aligned} G(\lambda u_1 + (1 - \lambda)u_2) &= \sup_{(\phi, \mathbf{v}) \in Sm(\Omega)} \int_{\Omega} \varepsilon \phi + (\lambda u_1 + (1 - \lambda)u_2) \operatorname{div} \mathbf{v} d\mu \\ &= \sup_{(\phi, \mathbf{v}) \in Sm(\Omega)} \left(\lambda \int_{\Omega} \varepsilon \phi + u_1 \operatorname{div} \mathbf{v} d\mu + (1 - \lambda) \int_{\Omega} \varepsilon \phi + u_2 \operatorname{div} \mathbf{v} d\mu \right) \\ &\leq \lambda \sup_{(\phi, \mathbf{v}) \in Sm(\Omega)} \int_{\Omega} \varepsilon \phi + u_1 \operatorname{div} \mathbf{v} d\mu + (1 - \lambda) \sup_{(\phi, \mathbf{v}) \in Sm(\Omega)} \int_{\Omega} \varepsilon \phi + u_2 \operatorname{div} \mathbf{v} d\mu \\ &= \lambda G(u_1) + (1 - \lambda)G(u_2). \end{aligned}$$

The above steps also work for F , so both are convex.

- F and G are lower semi-continuous. Let $u_0 \in H$ then

$$\begin{aligned} \liminf_{u \rightarrow u_0} G(u) &= \liminf_{u \rightarrow u_0} G(u) - G(u_0) + G(u_0) \\ &= \liminf_{u \rightarrow u_0} \left(\sup_{(\phi, \mathbf{v}) \in Sm(\Omega)} \int_{\Omega} \varepsilon \phi + u \operatorname{div} \mathbf{v} d\mu - \sup_{(\phi, \mathbf{v}) \in Sm(\Omega)} \int_{\Omega} \varepsilon \phi + u_0 \operatorname{div} \mathbf{v} d\mu \right) \\ &\quad + G(u_0) \end{aligned}$$

(let $(\phi_n, \mathbf{v}_n)_{n \in \mathbb{N}} \subset Sm(\Omega)$ so that $\int_{\Omega} \varepsilon \phi_n + u_0 \operatorname{div} \mathbf{v}_n d\mu \rightarrow G(u_0)$ as $n \rightarrow \infty$)

$$\begin{aligned} &\geq \liminf_{u \rightarrow u_0} \left(\lim_{n \rightarrow \infty} \int_{\Omega} (u - u_0) \operatorname{div} \mathbf{v}_n d\mu \right) + G(u_0) \\ &= \liminf_{u \rightarrow u_0} \left(\lim_{n \rightarrow \infty} (u - u_0, \operatorname{div} \mathbf{v}_n)_{L^2(\Omega)} \right) + G(u_0) \\ &= \liminf_{u \rightarrow u_0} \left(u - u_0, \lim_{n \rightarrow \infty} \operatorname{div} \mathbf{v}_n \right)_{L^2(\Omega)} + G(u_0) = G(u_0). \end{aligned}$$

And the same for F when $\varepsilon = 0$, so both are lower semi-continuous.

Now take $E = \max \{F(U), G(U)\} \leq TV_0(U) + \delta$ and $M = \|U\|_{L^2(\Omega)}$ then applying [17, Thm. 2] we get:

$$\begin{aligned} \|W^\varepsilon(\cdot, t) - W^0(\cdot, t)\|_{L^2(\Omega)} &\leq 16 (ME\delta t^2)^{1/6} + \|U - U\|_{L^2(\Omega)} \\ &\leq 16 (\|U\|_{L^2(\Omega)} (TV_0(U) + \delta) \delta t^2)^{1/6}, \end{aligned}$$

for all $t \in [0, \|U\|_{L^2(\Omega)}^6 (TV_0(U) + \delta)^6 / \delta^9]$.

Clearly if $U \in BV(\Omega)$ then as $\varepsilon \rightarrow 0$ the bound goes to zero and the time interval on which it is valid goes to infinity. $\blacksquare$

2.3.4 Isotropy Controlled Flow

We can make a further refinement to the TVF method by incorporating the isotropy factor. Recall that the spatial isotropy factor α_f takes values in $(0, 1]$ where a value close to 0 indicates that the data is anisotropic while a value close to 1 indicates that the data is isotropic. We can make the same modification to TVF that *Coherence Enhancing Diffusion* made to diffusion (without necessarily using gauge frames).

Definition 2.12 (Coherence Enhancing Total Variation Flow). Given $U \in BV(\Omega)$ with $\Omega \stackrel{\text{ap}}{\subset} \mathbb{M}_2$ and $\varepsilon, \zeta \geq 0$ then we call $W^\varepsilon : \mathbb{M}_2 \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ obeying

$$\begin{cases} \frac{\partial W^\varepsilon}{\partial t}(g, t) = \left(\operatorname{div} \circ \begin{pmatrix} 1 & \alpha_f(g) \\ & \alpha_f(g) \end{pmatrix} \right) \left(\frac{\nabla_\zeta W^\varepsilon(\cdot, t)}{\sqrt{\|\nabla_\zeta W^\varepsilon(\cdot, t)\|_{\mathfrak{g}_\zeta}^2 + \varepsilon^2}} \right)(g) & t \geq 0, \\ \nabla_{\mathbb{R}^2} W^\varepsilon(g, 0) \cdot \mathbf{N}(\mathbf{x}) = 0 & g = (\mathbf{x}, \mathbf{n}) \in \partial\Omega, \\ W^\varepsilon(g, 0) = U(g) & g \in \Omega. \end{cases} \quad (2.3)$$

the coherence enhancing total variation flow started at U , or equivalently the gradient flow of TV_ε started at U . Where we use Neumann boundary conditions with $\mathbf{N}$ as the normal to the spatial boundary.

Again gradient and divergence used here are either the left-invariant ones from section 1.1.5 (where the metric tensor is diagonalized in the left invariant frame) or the gauge frame ones from 1.2.3 (where the metric tensor is diagonalized in the gauge frame). The rescaling operator $\begin{pmatrix} 1 & \alpha_f(g) \\ & \alpha_f(g) \end{pmatrix}$ should be understood as being relative to one of those diagonalized metrics.

2.4 Mean Curvature Flow

Total Variation Flow not being invariant under monotonic codomain transformation leads us to consider a close relative that does have this property: *Mean Curvature Flow* (MCF).

2.4.1 Mean Curvature

Consider a curve in $\mathbb{R}^2$, then its mean curvature is defined in terms of its unit normal $\mathbf{n}$ as $(-\operatorname{div} \mathbf{n})$. We say that such a curve follows the mean curvature flow if all points $\mathbf{x}$ of the curve move as:

$$\partial_t \mathbf{x} = -\operatorname{div}(\mathbf{n}) \mathbf{n}. \quad (2.4)$$

Remark 2.13. Some sources define mean curvature multiplied with a constant factor $1/2$ or $1/n$ for higher dimensions, we do not include it in our definition.

Usually this flow will lead to singularities, such as in figure 2.1, after which the formulation (2.4) is no longer valid since the normal vector is not defined in singular points. This problem can be overcome by looking at the curve (or (hyper)surface) as the level set of an underlying function. Defined in such a way we can construct an evolution equation in such a way that the level set curves move along their mean curvature flow, see [27] for the original results or [28] for a more comprehensive overview.

Suppose a smooth function F on $\mathbb{R}^n$ then the evolution equation

$$\begin{cases} \partial_t F = \|\nabla F\| \operatorname{div} \left(\frac{\nabla F}{\|\nabla F\|} \right), \\ F(0) = F_0. \end{cases} \quad (2.5)$$

gives rise to the function's level sets following the mean curvature flow.

Remark that if we take $\lambda > 0$ then unlike for TVF we have $\partial_t(\lambda F) = \lambda \partial_t F$.

For an example of this construction see figure 2.1, where we let (2.4) act first on a polynomial on $\mathbb{R}^2$ and then on the same function which has had a significant amount of noise added, we plot the same level set at the same points in time in both cases. It should be clear from the figure why we would consider mean curvature flow for the purpose of denoising.

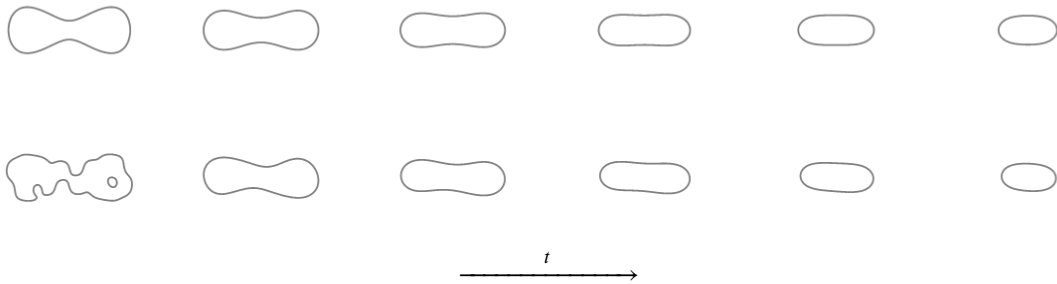


Figure 2.1: Mean curvature flow on a level set in 2D, top: starting from the polynomial $(x-1)^2(x+1)^2 + y^2$ for a value of 1, bottom: the same but with added sinusoidal noise.

Just like total variation flow, mean curvature flow admits weak solutions[27] which allows for its application to non-smooth imaging functions[6].

2.4.2 Enhancement PDE

In order for the PDE to be defined even when the gradient vanishes and to ensure numerical stability we add a small regularization constant ε in the same vein as [27, 6] but inline with our

construction of total variation flow. This brings us to our definition of mean curvature flow on $\mathbb{M}_2$.

Definition 2.14 (Mean Curvature Flow). Given $U \in BV(\Omega)$ with $\Omega \overset{\text{ap}}{\subset} \mathbb{M}_2$ and $\varepsilon, \zeta \geq 0$ then we call $W^\varepsilon : \mathbb{M}_2 \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ obeying

$$\begin{cases} \frac{\partial W^\varepsilon}{\partial t}(g, t) = \sqrt{\|\nabla_\zeta W^\varepsilon(\cdot, t)\|_{\mathfrak{g}_\zeta}^2 + \varepsilon^2} \operatorname{div} \left(\frac{\nabla_\zeta W^\varepsilon(\cdot, t)}{\sqrt{\|\nabla_\zeta W^\varepsilon(\cdot, t)\|_{\mathfrak{g}_\zeta}^2 + \varepsilon^2}} \right)(g) & t \geq 0, \\ \nabla_{\mathbb{R}^2} W(\mathbf{x}, \mathbf{n}, 0) \cdot \mathbf{N}(\mathbf{x}) = 0 & g = (\mathbf{x}, \mathbf{n}) \in \partial\Omega, \\ W^\varepsilon(g, 0) = U(g) & g \in \Omega. \end{cases}$$

the mean curvature flow started at U .

Just like TVF, our MCF formulation is equivariant under group action on $\mathbb{M}_2$.

Remark 2.15. Similarly to what we have done for TVF we can construct a *Coherence Enhancing Mean Curvature Flow* by replacing div by the operator $\operatorname{div} \circ \begin{pmatrix} 1 & & \\ & \alpha_f & \\ & & \alpha_f \end{pmatrix}$.

2.5 Implementation Notes

2.5.1 Numerical Stability

Our implementation depends on the *Lie Analysis*[29] package for Mathematica, which we use to calculate orientation scores. While the Lie Analysis package also provides gauge frames, we use our own implementation, which we started as an exercise but we noted that for our choice of parameters and the images we considered it performed slightly better. We tracked this down to the scale at which we calculate the eigenvectors¹, in theory this does not matter but numerically it does.

We discretize the right hand side of our evolution equations with first-order finite differences and then integrate using forward Euler. Forward Euler being only conditionally stable means we need to take care in choosing a time step. For diffusion we can determine a bound on Δt analytically via the Gershgorin circle theorem[30, Ch 4.2], but our equations are more difficult to analyze and we determined suitable Δt heuristically as we performed our experiments.

An important additional technique we already briefly mentioned needed to make the numerical implementation stable is smoothing. We apply Gaussian smoothing in two places:

1. We smooth the Hessian matrix component-wise before determining the first gauge frame vector.
2. We smooth the gradient norm.

Considering that we usually have much higher spatial resolution ($\gg 100$) than angular resolution (usually 8) we adapt the smoothing to reflect this by applying more (discrete) spatial smoothing than (discrete) angular smoothing. Additionally we employ different padding rules: spatially we need to introduce new data on the edges to maintain the size of the orientation score, which we do with fixed padding, angularly there is no need to extrapolate data because of the periodicity of S^1 , we simply pad with the data on the opposite side of the array.

¹Compare the two scales given in (1.28)

2.5.2 Timescales & Total Variation Scaling

Given the significant differences in the three techniques we presented we should not expect these to operate on the same timescale, and indeed they do not as can be seen in table 2.2.

Technique	Time to lowest L_1 error
CEDOS	0.25
Mean Curvature Flow	0.70
Total Variation Flow	0.02

Table 2.2: Comparing timescales by time to lowest L_1 error on a test image.

For the purpose of presentation and ease of comparison we will want to pretend that they are and plot results as such while cautioning the reader not to directly compare different methods at a particular time.

From table 2.2 we can see that the CEDOS and MCF minima are of the same order and we can plot any data on the same timescale without issues. TVF however acts on a much smaller timescale but instead of modifying the timescale after the fact we can use that for $\lambda > 0$ we have $TV(\lambda f) = TV(f)$ which means that the size of the integration step is invariant to scaling. If we then let W_1 be the total variation flow started at f and W_λ the flow started at λf we obtain:

$$W_1(\cdot, t) = \frac{1}{\lambda} W_\lambda(\cdot, \lambda t). \quad (2.6)$$

From which follows that if W_1 obtains its minimum error at time t then $\lambda^{-1} W_\lambda$ obtains the same minimum error at λt . Hence we simply need to choose λ appropriately to bring the temporal behavior of TVF inline with CEDOS and MCF. In our experiments we have chosen $\lambda = 50$.

2.6 Experiments

We center our experiments on the test image shown in figure 2.3a, to which we add zero-mean Gaussian noise with 0.3 standard deviation and clip back to $[0, 1]$.

2.6.1 Isotropic without Coherence Enhancement

Our first set of experiments uses isotropic ($\zeta = 1$) MCF and TVF without coherence enhancement to denoise the test image. We use both the left invariant and gauge frame variant and benchmark against CEDOS (or rather just DOS without the coherence enhancement, meaning simply diffusion on the orientation score). The key parameters that were used in this section are summarized in table 2.3.

Parameter	Purpose	Value
D_S	Spatial weighting	1
D_A	Angular weighting	0.01
ζ	Lateral weighting	1
c	Isotropy factor drop-off	0.2
λ	TVF scaling	50
ε	Gradient norm regularization	0.001

Table 2.3: Key parameters for isotropic experiments.

For a quantitative analysis we employ the relative L_1 and L_2 norms. Let $f_0 : \Omega \rightarrow [0, 1]$ be the source image and $f_t : \mathbb{R}_{\geq 0} \times \Omega \rightarrow [0, 1]$ the image resulting from the inverse orientation score

transform of the flow started at f_0 with added noise then we will look at the following quantity:

$$\frac{\|f_t - f_0\|_p}{\|f_0\|_p}, \quad (2.7)$$

where $p \in \{1, 2\}$.

We plot these quantities in figure 2.2. It is clear that TVF produces the smallest error and MCF is unable to outperform CEDOS in this case. We can also see that while adding gauge frames does not improve the minimum error much it does drastically diminish the image degradation due to over-smoothing for MCF and TVF.

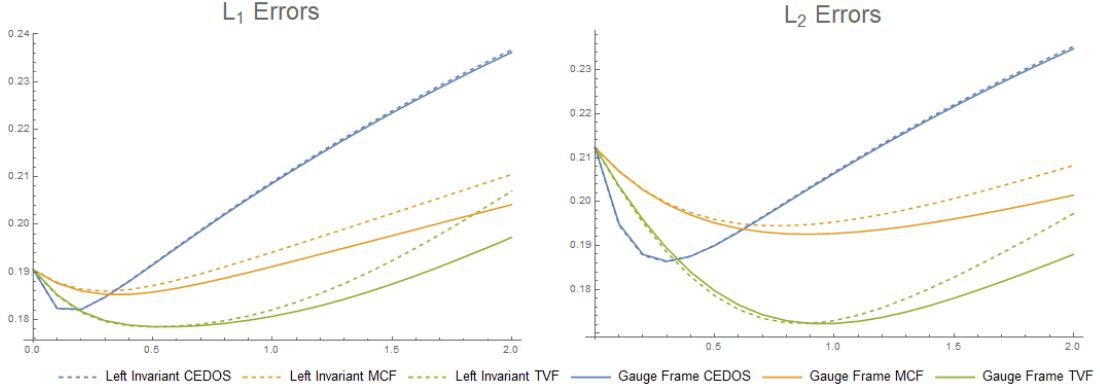
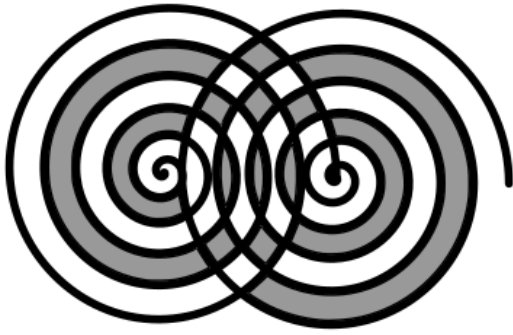


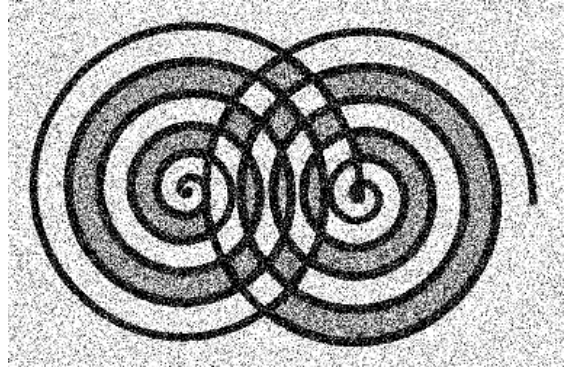
Figure 2.2: Relative errors for isotropic setup without coherence enhancement

For a qualitative comparison we smooth the test image for twice the time needed to reach the minimal L_2 error for each method so as to make the comparison as fair as possible. The resulting images are plotted in figure 2.3 along with the original image and the noisy image.

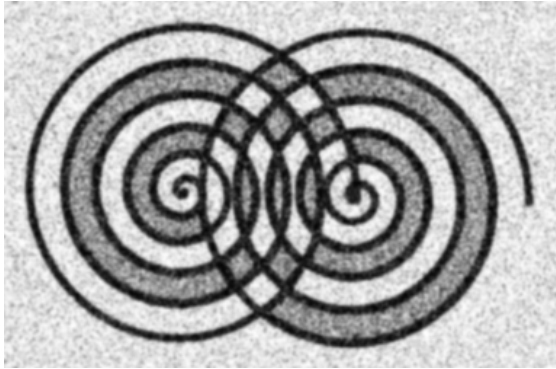
We can conclude that the qualitative comparison is inline with the error analysis, TVF clearly has dealt very well with the noise, almost fully clearing the plateaus while keeping the edges distinct. CEDOS and MCF maintain edges as well but they do not clear the plateaus nearly as well.



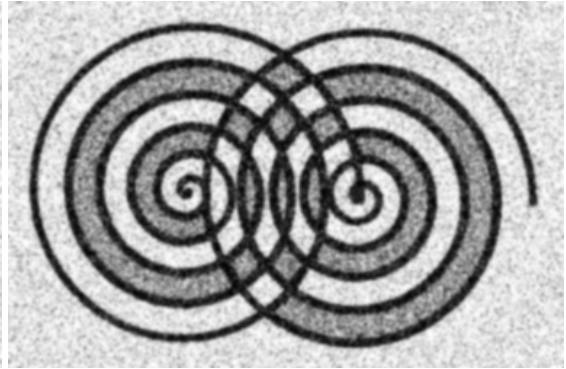
(a) Source image.



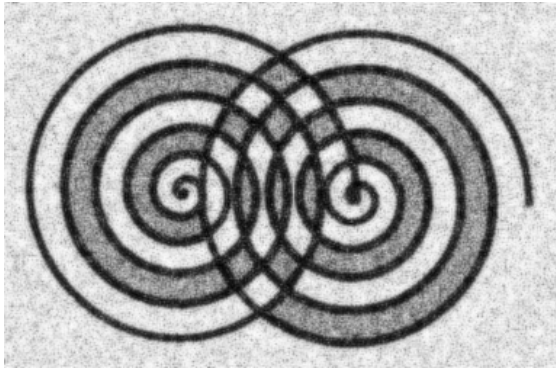
(b) Noisy image.



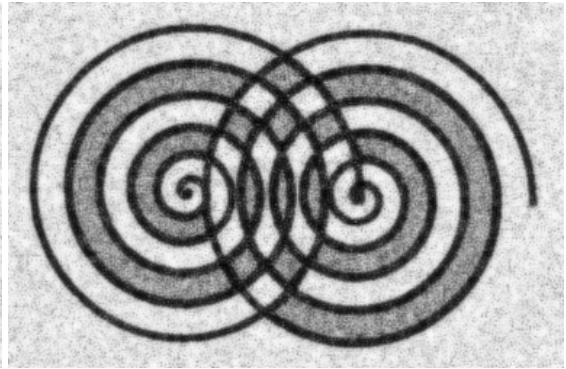
(c) CEDOS at $t = 0.6$.



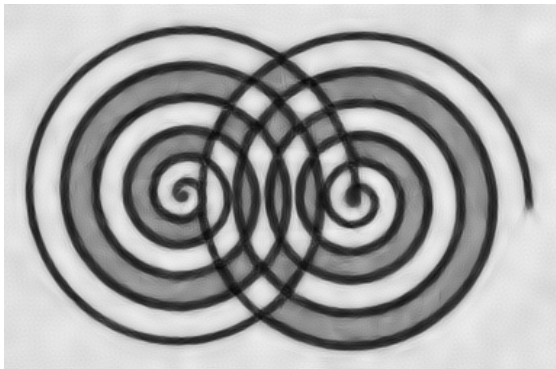
(d) Gauge frame CEDOS at $t = 0.6$.



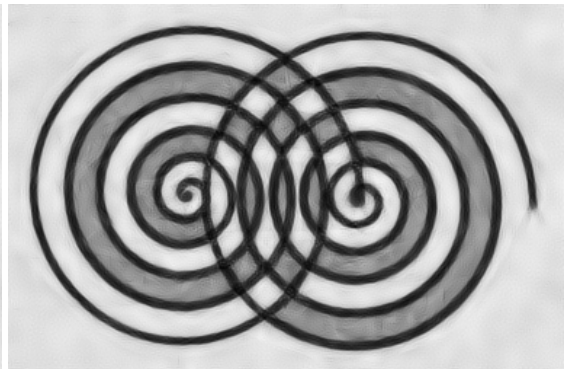
(e) MCF at $t = 1.6$.



(f) Gauge frame MCF at $t = 1.8$.



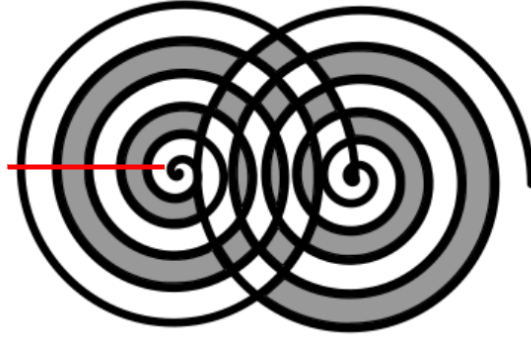
(g) TVF at $t = 1.8$.



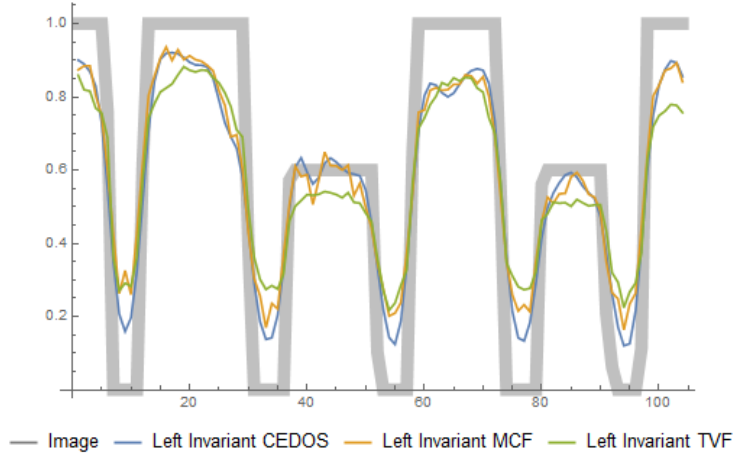
(h) Gauge frame TVF at $t = 1.8$.

Figure 2.3: Qualitative comparison of isotropic flows at twice the time needed to reach the minimal L_2 error.

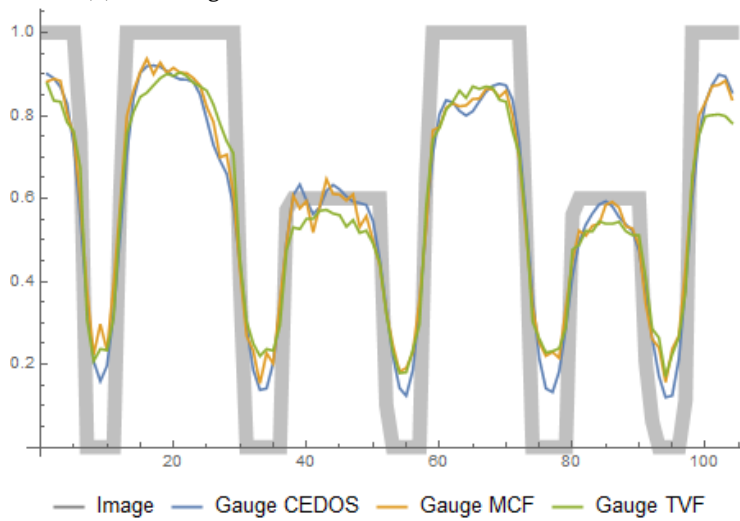
We can take a closer look at the edge vs. plateau behavior by looking at cross sections, we plot one in figure 2.4. We see that while TVF cleans up the plateaus better it does lose a bit more contrast than CEDOS or MCF, we will get back to this in 2.6.4.



(a) Location of the cross section.



(b) Resulting cross sections with left invariant methods.



(c) Resulting cross sections with gauge frames.

Figure 2.4: Cross section of isotropic flows without coherence enhancement at twice the time needed to reach the minimal L_2 error.

2.6.2 Isotropic with Coherence Enhancement

We repeat the previous experiment but this time we enable coherence enhancement, the parameters stay the same as in table 2.3. We plot the relative errors in figure 2.5.

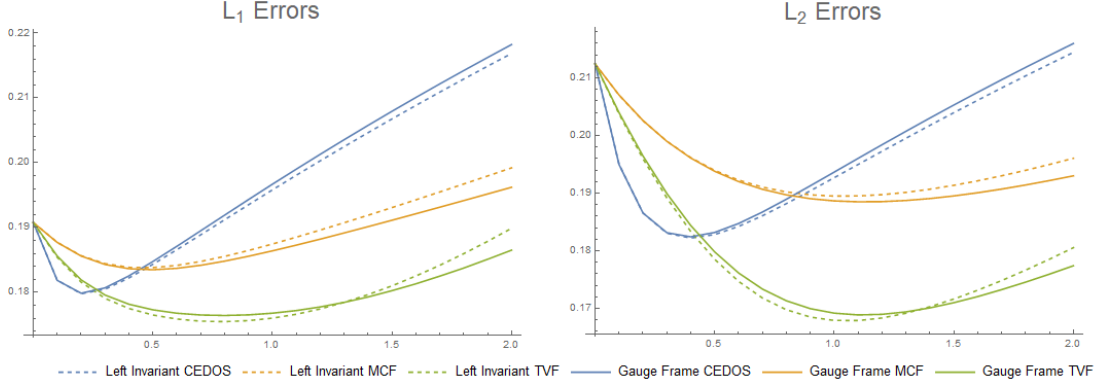


Figure 2.5: Relative errors for isotropic setup with coherence enhancement

The overall error behavior is the same with coherence enhancement as without coherence enhancement with a slight improvement minima. However this time the effect of switching to gauge frames is much less positive, CEDOS becomes overall worse, MCF only shows a reduced benefit and while TVF shows improved over-smoothing behavior its minimal error is increased.

Visually the results are indistinguishable from the non-coherence enhancement results shown in figure 2.3. We compare the error minima of our experiments so far in table 2.4.

Method	Without coherence enhancement		With coherence enhancement	
	L1	L2	L1	L2
Left invariant CEDOS	0.1826	0.1865	0.1798	0.1822
Gauge frame CEDOS	0.1827	0.1866	0.1799	0.1824
Left invariant MCF	0.1864	0.1949	0.1838	0.1895
Gauge frame MCF	0.1857	0.1930	0.1835	0.1885
Left Invariant TVF	0.1789	0.1727	0.1755	0.1679
Gauge frame TVF	0.1790	0.1727	0.1765	0.1689

Table 2.4: Comparing minimal relative L_1 and L_2 errors for isotropic methods.

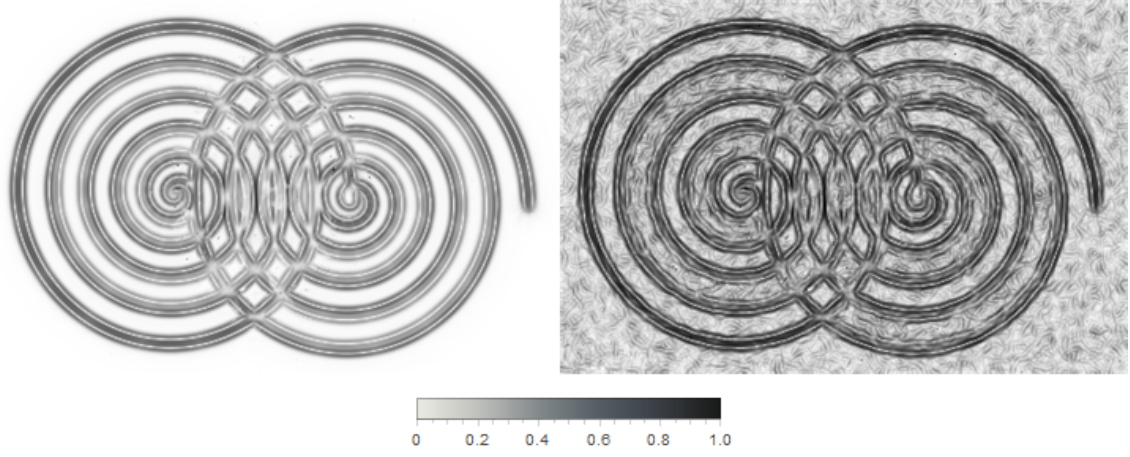
To explain the disappointing performance we take a look at the effect of noise on gauge frame fitting by looking at the gauge frame derived isotropy factor and the local curvature estimate. The isotropy factor we already know, the local (absolute) curvature we can estimate as follows[23, Eq. 32]: let $\mathcal{B}_1 = \sum_{i=1}^3 b^i \mathcal{A}_i$ be the first gauge frame vector then we calculate:

$$\frac{|b^3|}{\sqrt{(b^1)^2 + (b^2)^2}}. \quad (2.8)$$

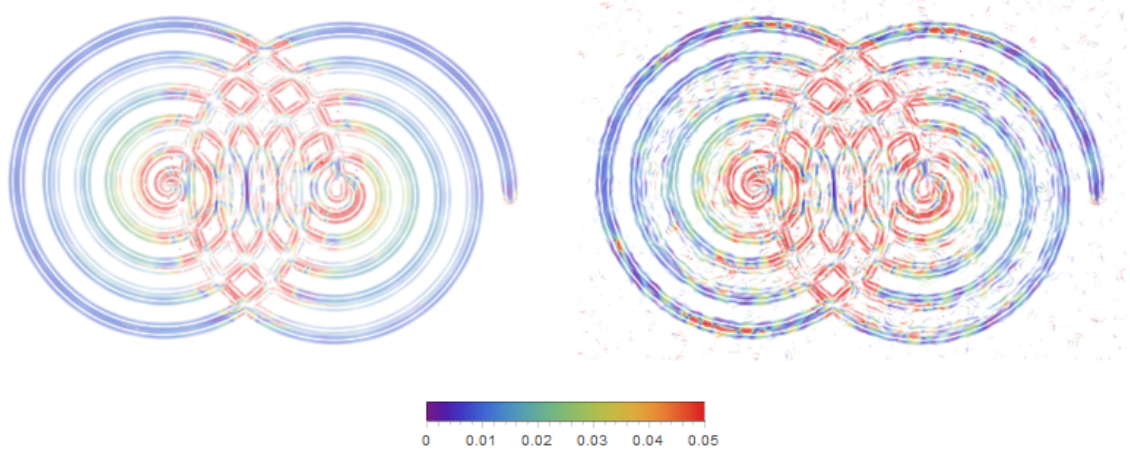
For the purpose of plotting we then use the isotropy factor to apply alpha so that we only show the local curvature at locations where we have a high degree of anisotropy.

The result of this calculation is shown in figure 2.6. We can see that noise significantly affects the result, while the isotropy factor map just becomes noisy the curvature shows large patches where the curvature estimate is significantly off, particularly on the outer parts of the spirals.

We can reason that using coherence enhancement or gauge frames is beneficial but *both* is not because using both we carry too much of the noise over into the final product.



(a) Gauge frame isotropy factor (minimum projection) of the source image (left) and the noisy image (right).



(b) Gauge frame derived curvature estimate of the source image (left) and the noisy image (right).

Figure 2.6: Looking at the effect of noise on gauge frame fitting.

2.6.3 Anisotropic

To see the effect of anisotropic flow we set $\zeta = 0.25$ and repeat the experiment, first without coherence enhancement. We plot the relative errors in figure 2.7.

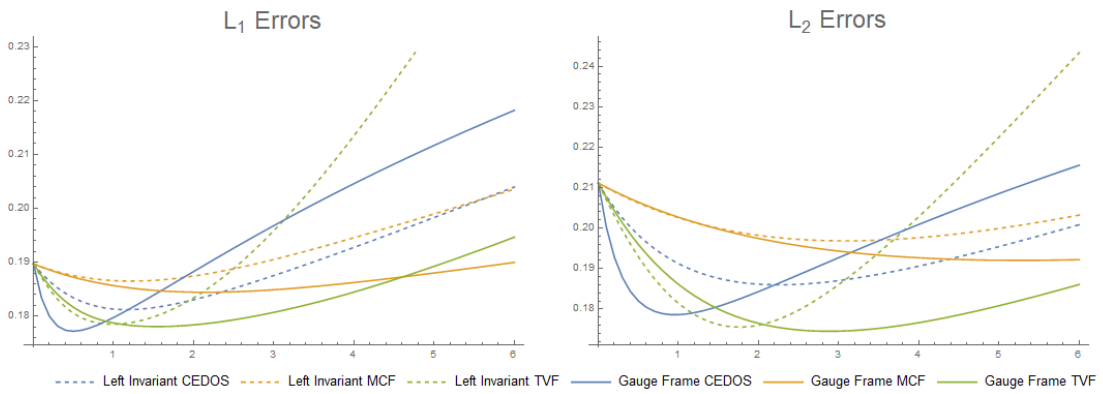


Figure 2.7: Relative errors for anisotropic ($\zeta = 0.25$) setup without coherence enhancement

The error plots have become more confused with the introduction of anisotropy, the six methods react quite differently to being laterally restricted. However looking at the over-smoothed images in figure 2.8 we can still see that TVF is far ahead of CEDOS and MCF when it comes to removing noise while still preserving edges.

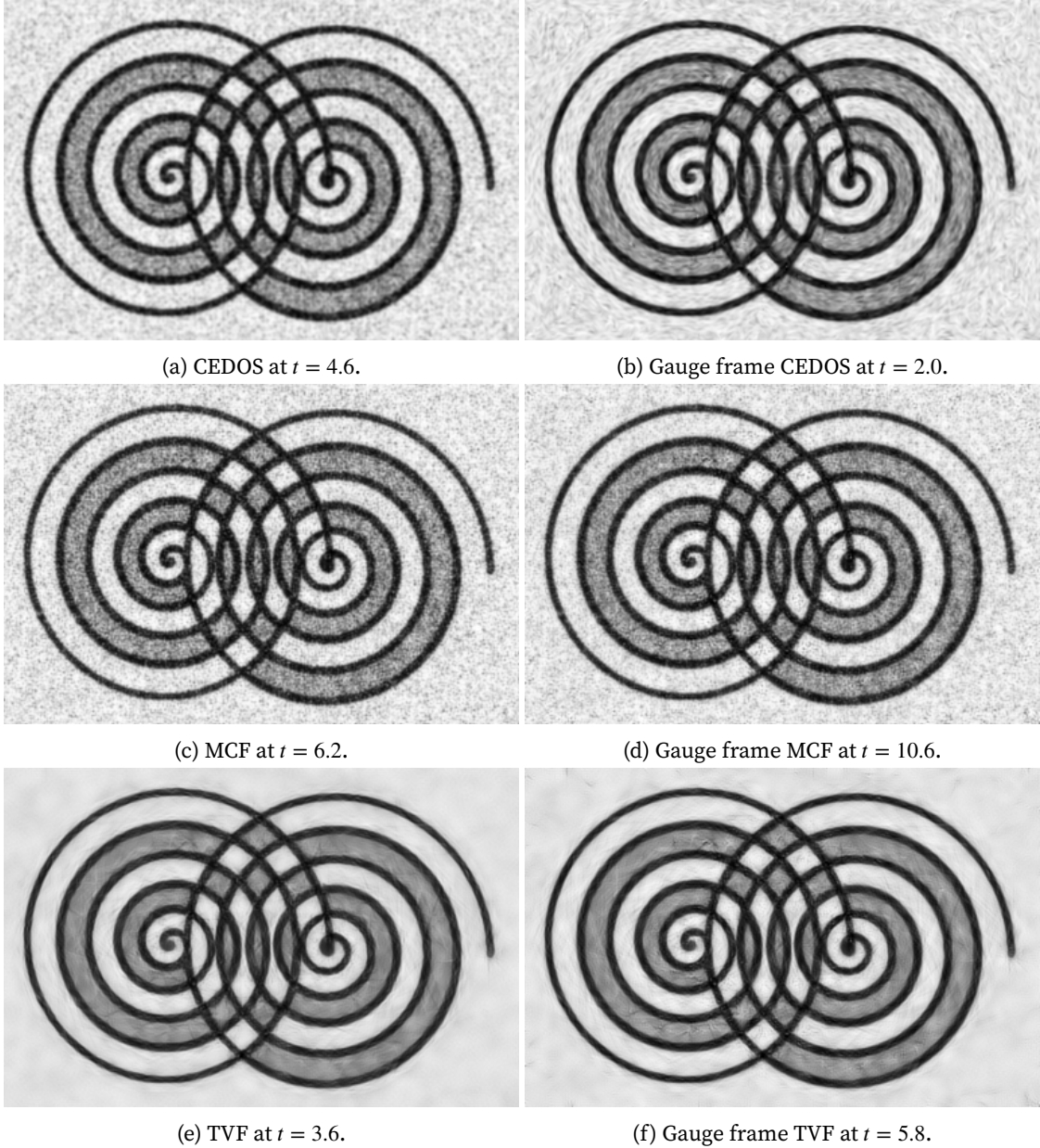


Figure 2.8: Qualitative comparison of isotropic flows ($\zeta = 0.25$) at twice the time needed to reach the minimal L_2 error.

Switching on coherence enhancement yields some quantitative improvement, as can be seen in figure 2.9, but yields no qualitative difference with the non-coherence enhancement images already shown in figure 2.8.

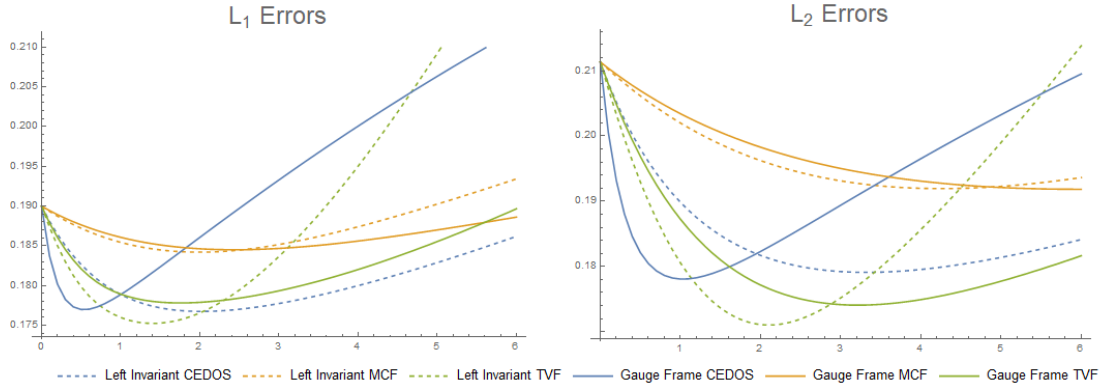


Figure 2.9: Relative errors for anisotropic ($\zeta = 0.25$) setup with coherence enhancement

We summarize the minimal relative errors in table 2.5. Comparing these numbers to table 2.4 leaves us with little reason to consider anisotropic flows, since only CEDOS has benefited from the lateral restriction, for the rest of the experiments we will only consider $\zeta = 1$.

Method	Without coherence enhancement		With coherence enhancement	
	L1	L2	L1	L2
Left invariant CEDOS	0.1812	0.1860	0.1768	0.1791
Gauge frame CEDOS	0.1772	0.1786	0.1770	0.1780
Left invariant MCF	0.1865	0.1969	0.1842	0.1919
Gauge frame MCF	0.1844	0.1921	0.1845	0.1918
Left Invariant TVF	0.1785	0.1756	0.1752	0.1710
Gauge frame TVF	0.1781	0.1745	0.1778	0.1741

Table 2.5: Comparing minimal relative L_1 and L_2 errors for anisotropic ($\zeta = 0.25$) methods.

2.6.4 Post-processing

Our experiments so far single out TVF as the most promising method and in the following sections we will endeavor to make optimal use of it.

In the cross sections we saw in figure 2.4 we already noticed that our method decreased contrast, this is a deficiency we can improve in post-processing. We do this by simply rescaling the result to $[0, 1]$. We compare the errors of TVF with and without post-processing in figure 2.10.

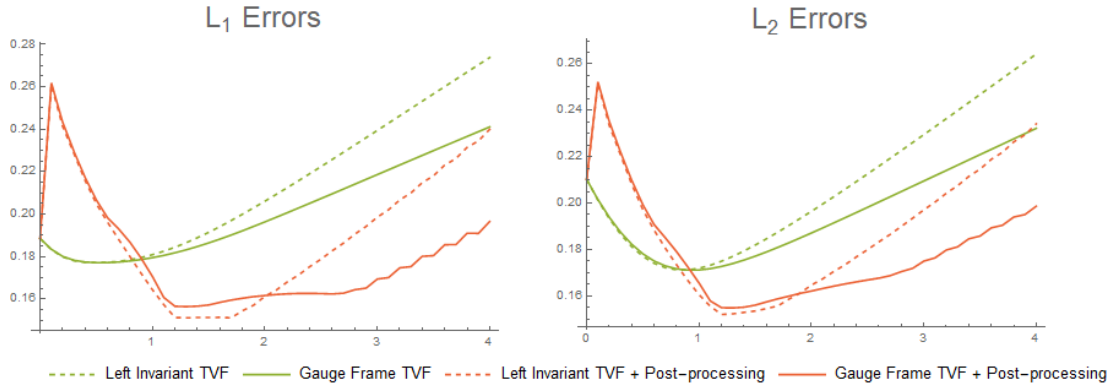


Figure 2.10: Relative errors for TVF (isotropic without coherence enhancement) with and without rescale post-processing.

Qualitatively this rescaling gives a predictable improvement as can be seen in figure 2.11.

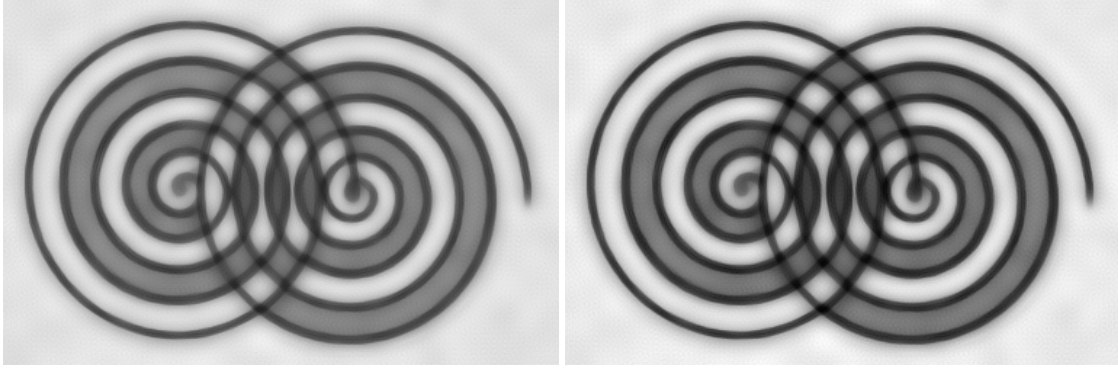


Figure 2.11: Left invariant TVF without (left) and with (right) rescale post-processing at $t = 4$.

We can conclude that this simple post-processing step is worth performing both from a quantitative and qualitative perspective.

2.6.5 3-Stage TVF Method

In section 2.6.2 we saw the detrimental effect noise had on gauge frame fitting which also affected coherence enhancement, but there is no reason we should fit gauge frames at the start. Combined with the efficacy of a post-processing step this leads us to propose the following three stage method:

1. pre-smooth the orientation score using left invariant TVF,
2. fit gauge frames and smooth using gauge frame TVF with coherence enhancement,
3. rescale post-processing on the produced image.

Going by our previous experiments we select a pre-smoothing time $t_1 = 1$ and call the subsequent main smoothing time t_2 . The relative errors of this method are plotted in figure 2.12.

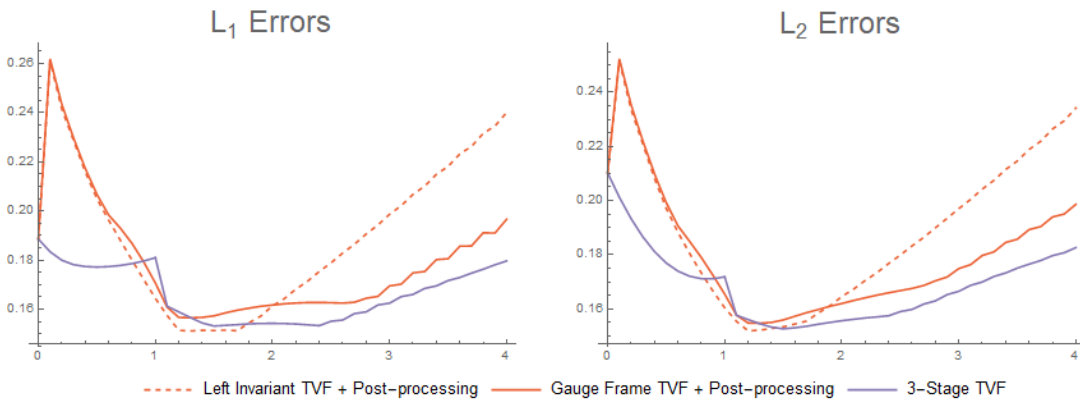


Figure 2.12: Relative errors of 3-stage TVF versus post-processed TVF.

While the minimal error is not improved by pre-smoothing, the range over which the error stays low is quite wide compared to plain post-processed TVF. Additionally the rate of image degradation due to over-smoothing is again improved which is a desirably property since in practice we will have to guess a good smoothing time.

In figure 2.13 the reader can see the final result of this method at $t = 2$.

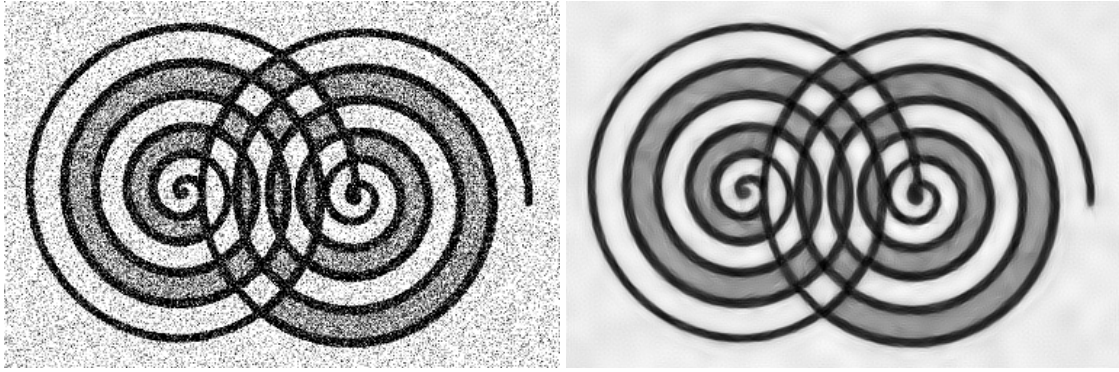
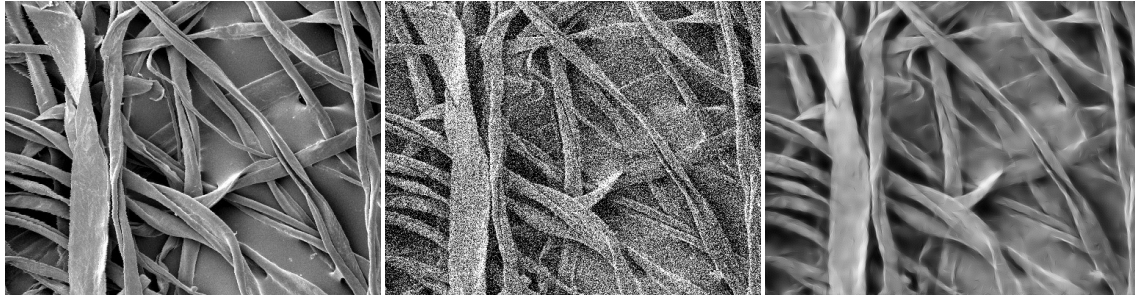


Figure 2.13: Contrasting the final result of the 3-stage TVF method at $t_1 = t_2 = 1$ (right) with the original noisy image (left).



(a) Collagen



(b) Mona Lisa

Figure 2.14: Contrasting the final result of the 3-stage TVF method at $t_1 = t_2 = 1$ (right) with the source image (left) and the source image with added noise (center).

Chapter 3

Machine Learning

Convolutional neural networks (CNNs) are a specialized kind of neural network that have been very successfully employed in such imaging applications as segmentation[31, 32], which is the application we will be looking at. In the context of image analysis segmentation means identifying each pixel of the image as belonging to a particular class. For example, as we will be doing with the DRIVE dataset, we can segment an image of the retina into 2 classes: blood vessels and background.

The work in this chapter replicates and makes some extensions to [9]. While most of the contents will focus on describing the structure of group convolution networks and their performance we will also take time to draw a connection between CNNs and PDEs.

3.1 Spatial Convolution Neural Network

We start off by giving a theoretical description of the conventional CNNs that are currently ubiquitous in imaging applications. We will implement such a network to benchmark our proposed enhancements against.

3.1.1 Layered Computation

As the name suggest CNNs rely on convolutions, which on $\mathbb{R}^2$ are formulated as:

$$(k *_{\mathbb{R}^2} f)(\mathbf{x}) := \int_{\mathbb{R}^2} k(\mathbf{x} - \mathbf{y}) f(\mathbf{y}) d\mathbf{y}, \quad (3.1)$$

with $k, f \in L^2(\mathbb{R}^2)$, where we refer to f as the image or input and k as the kernel.

We can extend this idea to multiple input and output channels (also called *features* in the context of machine learning): suppose $c_{in}, c_{out} \in \mathbb{N}$ so that $\mathbf{F}^{in} \in (L^2)^{c_{in}}$. Using a set of kernels of the form $\mathbf{K} \in (L^2)^{c_{in} \times c_{out}}$ we compute:

$$F_j^{out} = \sum_{i=1}^{c_{in}} K_{i,j} * F_i^{in} \quad (\forall j \in [1, c_{out}]), \quad (3.2)$$

which yields $\mathbf{F}^{out} \in (L^2)^{c_{out}}$. As a shorthand for this operation we introduce the following operator.

Definition 3.1 (Network Convolution Operator). Given $c_1, c_2 \in \mathbb{N}$, $\mathbf{F} \in (L^2(\mathbb{R}^2))^{c_1}$ and $\mathbf{K} \in (L^2(\mathbb{R}^2))^{c_1 \times c_2}$ we define:

$$\mathbb{C}_{\mathbf{K}} \mathbf{F} := \left(\sum_{i=1}^{c_1} K_{i,j} * F_i \right)_{j=1}^{c_2}.$$

The second ingredient of a CNN is the activation function that sits between convolutions, there is a myriad of options in choosing an activation function however in our experiments we only employ one:

Definition 3.2 (Rectified Linear Unit (ReLU)). Let $x \in \mathbb{R}$ then we define:

$$\phi(x) := \max(0, x).$$

We will also use the activation function as an operator where we commit the following abuse of notation:

$$(\phi \mathbf{F})(\cdot) := (\phi(F_1(\cdot)), \dots, \phi(F_n(\cdot))).$$

Remark 3.3. The fact that a ReLU is not differentiable can be a problem, for the purpose of analysis it can be approximated by the smooth function $\varepsilon \cdot \log(1 + e^{x/\varepsilon})$, which converges to a ReLU when $\varepsilon \rightarrow 0$.

A sequential N_l -layer segmentation CNN with N_c classes can then be written symbolically as:

$$f^{seg} = \underset{i \in \{1, 2, \dots, N_c\}}{\operatorname{argmax}} (\pi_i \circ \mathcal{C}_{\mathbf{K}^{(N_l)}} \circ \dots \circ \phi \circ \mathcal{C}_{\mathbf{K}^{(2)}} \circ \phi \circ \mathcal{C}_{\mathbf{K}^{(1)}}) f^{in}, \quad (3.3)$$

where π_i selects the i -th component, $\mathbf{K}^{(1)}$ has to have 1 input channels and $\mathbf{K}^{(N_l)}$ has to have N_c output channels, the result being $f^{seg} \in \mathbb{R}^2 \rightarrow \{1, 2, \dots, N_c\}$.

While the number of input channels of the first layer is dictated by the provided input and the number of output channels of the last layer is dictated by the classification problem, the number of intermediate channels or features can be freely chosen. We denote these parameters as $N^{(l)}$ for $l \in \{1, \dots, N_l - 1\}$ so that the set of kernels of layer l is shaped as:

$$\mathbf{K}^{(l)} \in (L^2(\mathbb{R}^2))^{N^{(l-1)} \times N^{(l)}}, \quad (3.4)$$

where $N^{(0)} = 1$ and $N^{(N_l)} = N_c$.

If we name the input of the network $\mathbf{F}^{(0)}$ then the intermediate output of each layer is given as :

$$\mathbf{F}^{(l)} = (\phi \circ \mathcal{C}_{\mathbf{K}^{(l)}}) \mathbf{F}^{(l-1)}, \quad (3.5)$$

which implies $\mathbf{F}^{(l)} \in (L^2(\mathbb{R}^2))^{N^{(l)}}$.

3.1.2 Loss & Optimization

The final ingredient needed to construct a neural network is a loss function to compare actual output against desired output. We do not want to compare the final output given by (3.3) since *argmax* is not differentiable. Again there is a large amount of choice here but we will use *softmax cross-entropy* [33], a common choice for training classification networks and readily differentiable.

This loss function starts from the output of the last layer $\mathbf{F}^{(N_l)}$ whose components (recall $\mathbf{F}^{(N_l)} \in (L^2(\mathbb{R}^2))^{N_c}$) assigns to each point, for each category, a score (in $\mathbb{R}$ commonly called *logits*) indicating the likelihood of that point belonging to that category. We can turn these into a pointwise probability distribution with the *softmax* function:

$$p_i(\mathbf{x}) := \frac{e^{F_i^{(N_c)}(\mathbf{x})}}{\sum_{j=1}^{N_c} e^{F_j^{(N_c)}(\mathbf{x})}} \quad i \in \{1, 2, \dots, N_c\}. \quad (3.6)$$

Notice that for all $\mathbf{x}$ that $\sum_{i=1}^{N_c} p_i(\mathbf{x}) = 1$ and $p_i(\mathbf{x}) \geq 0$, so $(p_i)_{i=1}^{N_c}$ is a proper discrete probability distribution at every point.

Remark 3.4. Since the softmax function is strictly monotone we can write

$$f^{seg} = \operatorname{argmax}_{i \in \{1, 2, \dots, N_c\}} p_i,$$

which would be identical to (3.3).

Given a ground-truth segmentation $f^* : \mathbb{R}^2 \rightarrow \{1, 2, \dots, N_c\}$ we can associate with it the following trivial pointwise probability distribution to compare with:

$$q_i(\mathbf{x}) := \mathbb{1} \{f^*(\mathbf{x}) = i\}, \quad (3.7)$$

which is a probability distribution since for all $\mathbf{x}$ we have $q_i(\mathbf{x}) \geq 0$ and $\sum_{i=1}^{N_c} q_i(\mathbf{x}) = 1$.

To measure the difference between these two distributions we use cross-entropy (also called relative entropy see [34, Ch. 2.3]):

$$H(q(\mathbf{x}), p(\mathbf{x})) := - \sum_{i=1}^{N_c} q_i(\mathbf{x}) \log p_i(\mathbf{x}), \quad (3.8)$$

which can be simplified since the target distribution $q(\mathbf{x})$ is given as (3.7):

$$H(q(\mathbf{x}), p(\mathbf{x})) = - \log p_{f^*(\mathbf{x})}(\mathbf{x}), \quad (3.9)$$

which gives us a pointwise loss that we can turn into a global loss function by integration.

Definition 3.5 (Softmax Cross-entropy Loss). Given an input image $f \in L^2(\Omega_f)$ and ground-truth segmentation $f^* : \Omega_f \rightarrow \{1, 2, \dots, N_c\}$ the softmax cross-entropy loss of the network is given as:

$$\ell_{sce}(f, f^*) := - \int_{\Omega_f} \log p_{f^*(\mathbf{x})}(\mathbf{x}) d\mathbf{x},$$

where p_i is calculated as the softmax over the output of the last layer of the network acting on f per (3.6).

If we took any probability distribution $p_i(\mathbf{x})$ the integral above can diverge but with $p_i(\mathbf{x})$ given by (3.6) and $F_i^{(N_c)} \geq 0$ due to the ReLU we have:

$$1 \geq p_i(\mathbf{x}) \geq \frac{1}{\sum_{j=1}^{N_c} e^{F_j^{(N_c)}(\mathbf{x})}} > 0 \quad (\text{a.e.}) \quad (3.10)$$

Combining this with the fact that in practice the domain Ω_f will be bounded means that the integral converges.

Now we can formulate our optimization problem. Given a training set of input images and ground-truth segmentations $(f_i, f_i^*)_{i=1}^{N_{train}}$ our goal is to choose the set of all kernels of our network $\mathbf{K}$ (assume we have N_K individual kernels) in such a way that the average loss over the training set is minimized, or formally:

$$\mathbf{K}^* := \operatorname{argmin}_{\mathbf{K} \in (L^2)^{N_K}} \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} \ell_{sce}(f_i, f_i^*). \quad (3.11)$$

Obviously this problem is not generally amenable to analytic solutions and we will have to turn to numerical approximations, details on this subject are beyond the scope of this report and we refer to [35] for details.

Convolutions being translation equivariant (as in it does not matter whether we translate the input function and then convolve or convolve and then translate the output, the result is the same) causes the CNN to also be translation equivariant, but it is not by itself rotation equivariant, while the segmentation problem is. Current practice is to train the network with the same images rotated at different angles to gain a form of trained rotation equivariance, this is less than optimal and we will see about making rotation equivariance a feature of the network itself.

3.2 Designed Equivariance

First of all we will make precise what we mean by equivariance in our context. Recall the definitions of the left regular representations $\mathcal{U}_g$ in def. 1.19 and $\mathcal{L}_g$ in def. 1.18.

Definition 3.6 (Roto-translation equivariance).

- (a) Let $\mathcal{X}$ be an operator from a function space on $\mathbb{R}^2$ to a function space on $\mathbb{R}^2$, then it is roto-translation equivariant if:

$$\mathcal{U}_g \circ \mathcal{X} = \mathcal{X} \circ \mathcal{U}_g \quad \forall g \in \mathbb{M}_2.$$

- (b) Let $\mathcal{Y}$ be an operator from a function space on $\mathbb{R}^2$ to a function space on $\mathbb{M}_2$, then it is roto-translation equivariant if:

$$\mathcal{L}_g \circ \mathcal{Y} = \mathcal{Y} \circ \mathcal{U}_g \quad \forall g \in \mathbb{M}_2.$$

- (c) Let $\mathcal{Z}$ be an operator from a function space on $\mathbb{M}_2$ to a function space on $\mathbb{R}^2$, then it is roto-translation equivariant if:

$$\mathcal{U}_g \circ \mathcal{Z} = \mathcal{Z} \circ \mathcal{L}_g \quad \forall g \in \mathbb{M}_2.$$

Remark that the orientation score transform obeys (b), per corollary 1.53.

Two variants of this definition that are also worth mentioning are:

- translation equivariance: in that case we only require that the equalities in the above definition hold only for all g in the translation subgroup of $\mathbb{M}_2$,
- roto-translation invariance: in the case of a classification problem we require the output to stay the same if the input roto-translates, i.e. if $\mathcal{X}$ is a classification of a function space on $\mathbb{R}^2$ we require $\mathcal{X} = \mathcal{X} \circ \mathcal{U}_g$ for all $g \in \mathbb{M}_2$ and we call this property *invariance* rather than equivariance.

Given that spatial CNNs are structurally translation invariant/equivariant we wonder if we can structure a network in such a way as to have more complicated equivariances such as roto-translation equivariance. The fact that activation functions are applied point-wise makes them invariant to any sort of domain transformation, so we need to focus on the linear operators in between.

Abstractly let us model the linear part of the layer by a bounded linear operator $K : L^2(X) \rightarrow L^2(Y)$ for some manifolds X and Y and let us choose a group G , that can act transitively on both the input domain X and output domain Y , which we want K to be equivariant with respect to. More precisely if A and B are left-regular representations of G on X resp. Y then we want $K \circ A_g = B_g \circ K$ for all $g \in G$.

The following theorem summarizes classical results on equivariant linear bounded operators from the field of harmonic analysis, which we here place in the context of machine learning via equivariant neural networks. Taking on a slightly different viewpoint, similar results are also discussed in detail in the context of deep learning in recent works [36, 37].

Theorem 3.7. *Let X and Y be manifolds on which the Lie group G acts transitively. Let μ_X and μ_Y be Radon measures on X resp. Y such that μ_X is invariant under group action:*

$$\forall g \in G, x \in X : d\mu_X(g \odot x) = d\mu_X(x).$$

Let $K : L^2(X, d\mu_X) \rightarrow L^2(Y, d\mu_Y)$ be a bounded linear operator such that $\sup_{\|f\|=1} \|Kf\|_{L^\infty} < \infty$. Finally let A and B be linear left-regular representations of G on X resp. Y such that $K \circ A_g = B_g \circ K$ then the following hold for all $f \in L^2(X)$.

1. K is a linear integral operator with unique kernel k :

$$\exists! k \in L^1(Y \times X, d\mu_X \times d\mu_Y) : (Kf)(y) = \int_X k(y, x) f(x) d\mu_X(x) \quad (\forall y \in Y).$$

2. After choosing a reference point $y_0 \in Y$ (so that $\forall y \in Y \exists g_y \in G$ such that $y = g_y \odot y_0$) K is a convolution operator almost everywhere on Y :

$$\exists! k_0 \in L^1(X, d\mu_X) : (Kf)(y) = \int_X k_0(g_y^{-1} \odot x) f(x) d\mu_X(x) \quad (\text{almost everywhere in } Y).$$

3. Moreover this k_0 must be invariant with respect to the stabilizer (in G) of the reference point y_0 :

$$\forall h \in \text{Stab}_G(y_0) : k_0(x) = k_0(h \odot x).$$

Proof.

Given that $\sup_{\|f\|=1} \|Kf\|_{L^\infty} < \infty$ it follows that K is a bounded operator from $L^2(X, d\mu_X)$ to $L^\infty(Y, d\mu_Y)$ and so an integral operator[38, Thm. 1.3], which proves the first claim.

Next, assuming $K \circ A_g = B_g \circ K$ for all $g \in G$ we continue for all $f \in L^2(X, d\mu_X)$,

$$\begin{aligned} (K \circ A_g)(f) &= (B_g \circ K)(f) \\ \Leftrightarrow \int_X k(y, x) f(g^{-1} \odot x) d\mu_X(x) &= \int_X k(g^{-1} \odot y, x) f(x) d\mu_X(x) \\ &\quad (\text{change of variables on the left } x' = g^{-1} \odot x) \\ \Leftrightarrow \int_X k(y, g \odot x') f(x') d\mu_X(g \odot x') &= \int_X k(g^{-1} \odot y, x) f(x) d\mu_X(x) \\ &\quad (\text{invariance of the measure } d\mu_X) \\ \Leftrightarrow \int_X k(y, g \odot x') f(x') d\mu_X(x') &= \int_X k(g^{-1} \odot y, x) f(x) d\mu_X(x) \\ \Leftrightarrow k(y, g \odot x) &= k(g^{-1} \odot y, x) \quad (\text{a.e. for all } (x, y) \in X \times Y) \\ \Leftrightarrow k(g \odot y, g \odot x) &= k(y, x) \quad (\text{a.e. for all } (x, y) \in X \times Y). \end{aligned}$$

Now using that for all $y \in Y$ we can find a $g_y \in G$ so that $y = g_y \odot y_0$ we find:

$$k(y, x) = k(g_y \odot y_0, x) = k(y_0, g_y^{-1} \odot x).$$

So when we define $k_0(x) := k(y_0, x)$ we can write:

$$k(y, x) = k_0(g_y^{-1} \odot x),$$

which proves the second claim.

Finally if we take $h \in \text{Stab}_G(y_0)$ we fill in:

$$k_0(h \odot x) = k(y_0, h \odot x) = k(h^{-1} \odot y_0, x) = k(y_0, x) = k_0(x),$$

which proves the last claim. ■

Now that we understand the structure of the linear operations in general we can see what this translates to¹ for two-dimensional networks that we want to make roto-translation equivariant. The natural domain for such operations is $\mathbb{M}_2$ but the input and outputs live on $\mathbb{R}^2$, so we need three types of layers, first a *lifting* layer from $\mathbb{R}^2$ to $\mathbb{M}_2$ then a series of layers from $\mathbb{M}_2$ to $\mathbb{M}_2$ and finishing with a *projection* layer from $\mathbb{M}_2$ back to $\mathbb{R}^2$. Applying theorem 3.7 we find the following result.

¹Pun not intended.

Corollary 3.8.

A CNN consisting of the following layers is roto-translation equivariant.

- (a) A lifting layer where $X = \mathbb{R}^2$, $Y = \mathbb{M}_2$, $y_0 = (\mathbf{0}, 0)$, $g_y = (\mathbf{x}, \theta)$ and $k_0 \in L^2(\mathbb{R}^2)$ so that the output needs to be calculated from an input $f \in L^2(\mathbb{R}^2)$ as:

$$(\mathbf{x}, \theta) \mapsto \int_{\mathbb{R}^2} k_0(R_\theta^{-1}(\mathbf{x} - \mathbf{x}')) f(\mathbf{x}') d\mathbf{x}'.$$

- (b) Convolutions layers where $X = Y = \mathbb{M}_2$, $y_0 = (\mathbf{0}, 0)$, $g_y = (\mathbf{x}, 0)$ and $k_0 \in L^2(\mathbb{M}_2)$ so that from an input $F \in L^2(\mathbb{M}_2)$ the output needs to be calculated as:

$$(\mathbf{x}, \theta) \mapsto \int_{\mathbb{R}^2} \int_{(-\pi, \pi)} k_0(R_{\theta'}^{-1}(\mathbf{x} - \mathbf{x}'), \theta - \theta') F(\mathbf{x}', \theta') d\theta' d\mathbf{x}'.$$

- (c) A projection layer where $X = \mathbb{M}_2$, $Y = \mathbb{R}^2$, $y_0 = \mathbf{0}$, $g_y = (\mathbf{x}, 0)$ and $k_0 \in L^2(\mathbb{M}_2)$ so that from an input $F \in L^2(\mathbb{M}_2)$ the output needs to be calculated as:

$$\mathbf{x} \mapsto \int_{\mathbb{R}^2} \int_{(-\pi, \pi)} k_0(R_{\theta'}^{-1}(\mathbf{x} - \mathbf{x}'), -\theta') F(\mathbf{x}', \theta') d\theta' d\mathbf{x}',$$

and, while this was trivial in the two previous cases since the only stabilizer was the identity, we now have an additional constraint on our kernel:

$$k_0(R_\theta \mathbf{x}, \theta) = k_0(\mathbf{x}, 0).$$

Using this theoretical result we can redesign our network.

3.3 Group Convolution Neural Network

Now we apply the results of corollary 3.8 to the multi feature design of our spatial CNN to make a comparable network that is roto-translation invariant from the start.

3.3.1 Lifting

The first layer of the network has a kernel set of the form $\mathbf{K}^{(1)} \in L^2(\mathbb{R}^2)^{N^{(1)}}$ which, when we let $g = (\mathbf{x}, \mathbf{n}) \in \mathbb{M}_2$, is used to compute the output of the layer as:

$$F_i^{(1)}(g) = \left(\left(\mathcal{R}_{\mathbf{n}} K_i^{(1)} \right) *_{\mathbb{R}^2} f \right) (\mathbf{x}) = \left(\mathcal{U}_g K_i^{(1)}, f \right)_{L^2(\mathbb{R}^2)}, \quad (3.12)$$

with $(\mathcal{R}_{\mathbf{n}} f)(\mathbf{y}) := f(R_{\mathbf{n}}^{-1} \mathbf{y})$ the rotation operator associated with $\mathbf{n}$. Consequently $\mathbf{F}^{(1)} \in (L^2(\mathbb{M}_2))^{N^{(1)}}$.

Remark 3.9. An alternative approach here would be to use the orientation score transform itself as the first fixed function layer of the network and start the actual trainable part of the network on the orientation score itself. We set aside this idea for now and proceed with a multi-feature trainable lifting layer.

Symbolically we will represent this layer by the lifting operator $\mathcal{C}^\dagger$.

Definition 3.10 (Lifting Operator). Given $f \in L^2(\mathbb{R}^2)$ and a kernel set $\mathbf{K} \in (L^2(\mathbb{R}^2))^N$ then the lifting operator is defined as:

$$\left(\mathcal{C}_{\mathbf{K}}^\dagger f \right) (\mathbf{x}) := \left(\int_{\mathbb{R}^2} K_i(R_{\mathbf{n}}^{-1}(\mathbf{x} - \mathbf{y})) f(\mathbf{y}) d\mathbf{y} \right)_{i=1}^N.$$

Now that we have feature maps that live on $\mathbb{M}_2$ we can generalize the convolution layer.

3.3.2 Group Convolution

In general we can define convolution on groups as follows.

Definition 3.11 (Group Convolution). Given a Lie group G and $K, F \in L^2(G)$, then the convolution of K and F is given by:

$$K *_G F(g) := \int_G K(h^{-1} \odot g) F(h) d\mu(h),$$

where $\mu : G \rightarrow \mathbb{R}^+$ is the left-invariant Haar measure.

For brevity we will write $dh := d\mu(h)$ and we normalize the Haar measure such that when we switch to the helical atlas we can write $dh = dx dy d\theta$ as we will do below. Also note that the Haar is exactly that measure which satisfies the group invariance required by theorem 3.7.

Remark 3.12. Observe that spatial convolution as in (3.1) is a special case of group convolution using the 2-dimensional translation group.

In the case of $SE(2)$ and using the helical atlas from definition 1.7 this becomes:

$$\begin{aligned} K *_M F(\mathbf{x}, \theta) &= \int_{\mathbb{R}^2} \int_{(-\pi, \pi)} K\left((\mathbf{x}', \theta')^{-1} \odot (\mathbf{x}, \theta)\right) F(\mathbf{x}', \theta') d\theta' d\mathbf{x}' \\ &= \int_{\mathbb{R}^2} \int_{(-\pi, \pi)} K\left(R_{\theta'}^{-1}(\mathbf{x} - \mathbf{x}'), \theta - \theta'\right) F(\mathbf{x}', \theta') d\theta' d\mathbf{x}'. \end{aligned} \quad (3.13)$$

Note how the appearance of the rotation $R_{\theta'}^{-1}$ makes this operation distinct from convolution on $\mathbb{R}^3$, as the kernel not only translates over the function F but rotates at the same time as it moves along the orientation axis, this difference in kernel behavior is illustrated in figure 3.1. We also refer to this motion as a *shift-twist*: the kernel twists while it shifts over θ .

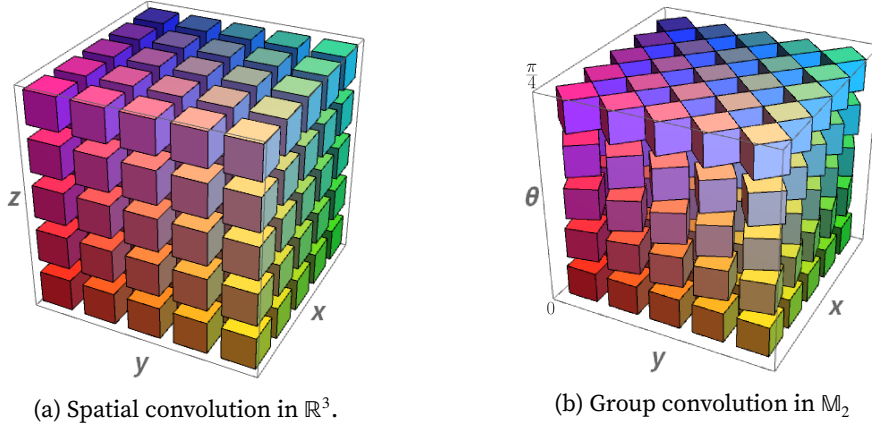


Figure 3.1: Comparing kernel traversal in $\mathbb{R}^3$ versus $\mathbb{M}_2$.

Now we can generalize out networks convolution layer to work on $\mathbb{M}_2$.

Definition 3.13 (Network Group Convolution Operator). Given $c_1, c_2 \in \mathbb{N}$, $\mathbf{F} \in (L^2(\mathbb{M}_2))^{c_1}$ and $\mathbf{K} \in (L^2(\mathbb{M}_2))^{c_1 \times c_2}$ we define:

$$\mathbb{C}_{\mathbf{K}} \mathbf{F} := \left(\sum_{i=1}^{c_1} K_{i,j} *_M F_i \right)_{j=1}^{c_2}.$$

3.3.3 Projection

Eventually we will need to output a function on $\mathbb{R}^2$, which we accomplish through a projection layer. Some care needs to be taken in how we do this to ensure that we maintain the networks equivariance, formally the projection operator $\mathcal{C}^\downarrow$ needs to obey:

$$\mathcal{C}^\downarrow \circ \mathcal{L}_g = \mathcal{U}_g \circ \mathcal{C}^\downarrow, \quad (3.14)$$

similar to the requirement in part (c) of corollary 3.8 and also corollary 1.53.

We saw in corollary 3.8 how we need to accomplish this for bounded linear operators. Instead of going that route we will be using a non-linear unbounded projection related to the *dilation* morphological filter. The reason for this is twofold:

1. it is a generalization of the projection layer successfully used in [9],
2. it is the first experiment with the larger idea of basing CNN design on PDEs (see appendix A for an overview of this idea).

We will not go into morphological operations in general in this report but develop this specific operation in isolation so that this report is self-contained.

We will first be looking at straightforward maximum projection.

Definition 3.14 (Maximum Projection). Let $\mathbf{F} \in (L^\infty(\mathbb{M}_2))^N$ be a set of feature maps, then we define:

$$(\mathcal{C}_{\max}^\downarrow \mathbf{F})(\mathbf{x}) := \left(\max_{\theta \in (-\pi, \pi)} F_i(\mathbf{x}, \theta) \right)_{i=1}^N.$$

Quite clearly maximum projection satisfies (3.14).

For our second option we can generalize maximum projection by considering the following map:

$$\mathbf{x} \mapsto \max_{\substack{\theta' \in (-\pi, \pi) \\ \mathbf{x}' \in \mathbb{R}^2}} K(\mathbf{x} - \mathbf{x}', \theta') + F(\mathbf{x}', \theta'). \quad (3.15)$$

Where we let $K : \mathbb{M}_2 \rightarrow (-\infty, 0]$. Observe how plain maximum projection is recovered by choosing

$$K(\mathbf{x}, \theta) = \begin{cases} 0 & \text{if } \mathbf{x} = \mathbf{0}, \\ -\infty & \text{else.} \end{cases} \quad (3.16)$$

The fact that the last expression does not depend on θ gives us a clue about the restrictions we need to place on our choice of K to satisfy equivariance, which we examine by expanding both sides of (3.14). Let $g = (\mathbf{x}_g, \theta_g)$, then on the right-hand side:

$$(\mathcal{U}_g \circ \mathcal{C}^\downarrow F)(\mathbf{x}) = \max_{\substack{\theta' \in (-\pi, \pi) \\ \mathbf{x}' \in \mathbb{R}^2}} K(R_{\theta_g}^{-1}(\mathbf{x} - \mathbf{x}_g) - \mathbf{x}', \theta') + F(\mathbf{x}', \theta'), \quad (3.17)$$

and on the left-hand side:

$$\begin{aligned} (\mathcal{C}^\downarrow \circ \mathcal{L}_g F)(\mathbf{x}) &= \max_{\substack{\theta'' \in (-\pi, \pi) \\ \mathbf{x}'' \in \mathbb{R}^2}} K(\mathbf{x} - \mathbf{x}'', \theta'') + (\mathcal{L}_g F)(\mathbf{x}'', \theta'') \\ &= \max_{\substack{\theta'' \in (-\pi, \pi) \\ \mathbf{x}'' \in \mathbb{R}^2}} K(\mathbf{x} - \mathbf{x}'', \theta'') + F(R_{\theta_g}^{-1}(\mathbf{x}'' - \mathbf{x}_g), \theta'' - \theta_g) \\ &\quad \left(\text{change of variables } \mathbf{x}' = R_{\theta_g}^{-1}(\mathbf{x}'' - \mathbf{x}_g) \quad \text{and} \quad \theta' = \theta'' - \theta_g \right) \\ &= \max_{\substack{\theta' \in (-\pi, \pi) \\ \mathbf{x}' \in \mathbb{R}^2}} K(\mathbf{x} - \mathbf{x}_g - R_{\theta_g} \mathbf{x}', \theta' + \theta_g) + F(\mathbf{x}', \theta'). \end{aligned} \quad (3.18)$$

A sufficient condition to ensure (3.17) equals (3.18) is:

$$K \left(R_{\theta_g}^{-1} (\mathbf{x} - \mathbf{x}_g) - \mathbf{x}', \theta' \right) = K \left(\mathbf{x} - \mathbf{x}_g - R_{\theta_g} \mathbf{x}', \theta' + \theta_g \right),$$

or after the change of variables $\mathbf{y} = R_{\theta_g}^{-1} (\mathbf{x} - \mathbf{x}_g) - \mathbf{x}'$ and setting $\theta' = 0$ this simplifies to:

$$K (\mathbf{y}, 0) = K \left(R_{\theta_g} \mathbf{y}, \theta_g \right). \quad (3.19)$$

Meaning that at each orientation layer the kernel K has to be a rotated version of itself at the reference orientation layer, and hence it suffices to specify K as a 2-dimensional function in order to fully specify an equivariant projection. We formalize this construction in the following definition.

Definition 3.15 (Dilated Projection). Let $\mathbf{F} \in (L^\infty(\mathbb{M}_2))^N$ be a set of feature maps and $\mathbf{K} \in (\mathbb{R}^2 \rightarrow (-\infty, 0])^N$ a set of planar kernels, then we define:

$$\left(\mathcal{C}_{\mathbf{K}}^\downarrow \mathbf{F} \right) (\mathbf{x}) := \left(\max_{\substack{\theta \in (-\pi, \pi) \\ \mathbf{y} \in \mathbb{R}^2}} K_i \left(R_\theta^{-1} (\mathbf{x} - \mathbf{y}) \right) + F_i (\mathbf{y}, \theta) \right)_{i=1}^N.$$

The following theorem about dilated projection ties it into morphological projection and so into the PDE interpretation of appendix A. However for a numerical implementation it is considerably more efficient to fuse these two operations, which is why we present them as such.

Theorem 3.16 (Dilation/Projection Concatenation). *The dilated projection operator $\mathcal{C}_{\mathbf{K}}^\downarrow$ is the concatenation of a left-invariant dilation on the group followed by the maximum projection over the orientations.*

Proof. Consider the maximum projection over the dilation with kernel $\bar{K}$:

$$\begin{aligned} \max_{\theta \in (-\pi, \pi)} \left(\bar{K} \square_{\mathbb{M}_2} F \right) (\mathbf{x}, \theta) &= \max_{\theta \in (-\pi, \pi)} \left(\max_{\substack{\theta' \in (-\pi, \pi) \\ \mathbf{x}' \in \mathbb{R}^2}} \bar{K} \left(R_{\theta'}^{-1} (\mathbf{x} - \mathbf{x}') \right), \theta - \theta' \right) + F(\mathbf{x}', \theta') \\ &= \max_{\substack{\theta' \in (-\pi, \pi) \\ \mathbf{x}' \in \mathbb{R}^2}} \left(\max_{\theta \in (-\pi, \pi)} \bar{K} \left(R_{\theta'}^{-1} (\mathbf{x} - \mathbf{x}') \right), \theta - \theta' \right) + F(\mathbf{x}', \theta') \\ &= \max_{\substack{\theta' \in (-\pi, \pi) \\ \mathbf{x}' \in \mathbb{R}^2}} \left(\max_{\theta \in (-\pi, \pi)} \bar{K} \left(R_{\theta'}^{-1} (\mathbf{x} - \mathbf{x}') \right), \theta \right) + F(\mathbf{x}', \theta'), \end{aligned}$$

now take $K(\mathbf{x}) := \max_{\theta' \in (-\pi, \pi)} \bar{K}(\mathbf{x}, \theta')$ which we substitute:

$$= \max_{\substack{\theta' \in (-\pi, \pi) \\ \mathbf{x}' \in \mathbb{R}^2}} \left\{ K \left(R_{\theta'}^{-1} (\mathbf{x} - \mathbf{x}') \right) + F(\mathbf{x}', \theta') \right\}.$$

■

Remark 3.17. In the proof above in the special case where $\bar{K}$ is binary (i.e., $\bar{K} : \mathbb{M}_2 \rightarrow \{-\infty, 0\}$) with 0 on a sub-group $H \subset \mathbb{M}_2$ and $-\infty$ otherwise, this morphological convolution $\square_{\mathbb{M}_2}$ coincides with the definition of (left-)coset pooling in G-CNNs given in [12].

The previous derivations also prove the following statement.

Proposition 3.18. *The dilated projection operator $\mathcal{C}_{\mathbf{K}}^\downarrow$ is equivariant with roto-translations, i.e.*

$$\mathcal{C}_{\mathbf{K}}^\downarrow \circ \mathcal{L}_g = \mathcal{U}_g \circ \mathcal{C}_{\mathbf{K}}^\downarrow.$$

Now we have a choice in how we wish to specify these projection kernels. One option is leaving them trainable and letting the optimizer worry about them as we did with the lifting layer. The second option is specifying a fixed kernel of our own design. For this report we will be taking the second route and develop an anisotropic projection layer.

The idea behind anisotropic projection is that for a given orientation we might want to probe more for a maximum along that orientation longitudinally then laterally. We can do this by equipping $\mathbb{R}^2$ with the following anisotropic norm:

Definition 3.19. Let $\mathbf{x} \in \mathbb{R}^2$ then:

$$\|\mathbf{x}\|_\zeta^2 := \sqrt{x_1^2 + \zeta^{-2}x_2^2}.$$

Which we can then use to specify the kernel K of a dilated projection as:

$$K(\mathbf{x}) = -\lambda \|\mathbf{x}\|_\zeta^p. \quad (3.20)$$

Where $\lambda > 0$ is used to scale the kernel and p specifies the sharpness.

Remark 3.20. This anisotropic norm works well because it is a good local approximation of the max projection of the group distance $d_{\mathcal{G}_\zeta}((\mathbf{x}, \theta))$ which we saw and approximated in section 1.1.7. After we set $D_S = 1$ we obtain:

$$\max_{\theta \in (-\pi, \pi)} c^1(x, y, \theta)^2 + \frac{c^2(x, y, \theta)^2}{\zeta^2} + \frac{\theta^2}{D_A} = x^2 + \frac{y^2}{\zeta^2} - \frac{(1 - \zeta^2)^2}{\zeta} \frac{x^2 y^2}{x^2 + \zeta^2 y^2 + 4 \frac{\zeta^2}{D_A^2}}. \quad (3.21)$$

For our application due to the use of batch-normalization and a ReLU the data is constrained in $[0, 1]$ meaning the shape of the projection kernel past the value of -1 (i.e. close to the origin for an appropriate choice of λ) has no effect and our anisotropic norm is a good estimate where it matters. We can see that if $x^2 + y^2 \leq 1$ that the last term above can be bounded:

$$\left| \frac{(1 - \zeta^2)^2}{\zeta} \frac{x^2 y^2}{x^2 + \zeta^2 y^2 + 4 \frac{\zeta^2}{D_A^2}} \right| \leq \frac{(1 - \zeta^2)^2 D_A^2}{4 \zeta^4}.$$

This approximation holds for sufficiently large ζ and sufficiently small D_A , in our network we use $\zeta = 1/2$ and $D_A = 1/10$ for which the approximation is off by at most $9/400$.

Figure 3.2 compares several estimates around the origin, after sampling on a small grid there is not much difference between various types of approximations.

The contours of the mapping $(\mathbf{x}, \theta) \mapsto K(R_\theta^{-1} \mathbf{x})$ then take on the appearance of cylinders that are first flattened (by $\zeta < 1$) and then twisted (by R_θ^{-1}), as can be seen in figure 3.3.

In the discrete setting we specify the kernel on a fixed $N_s \times N_s \times N_\theta$ grid as follows:

$$K_{i,j,k} = -\frac{1}{N_s} \left\| \left(\left(i - \frac{N_s + 1}{2} \right) \Delta x, \left(j - \frac{N_s + 1}{2} \right) \Delta y, k \Delta \theta \right) \right\|_\zeta^p. \quad (3.22)$$

The specific kernel we used in our experiment is visualized in figure 3.4.

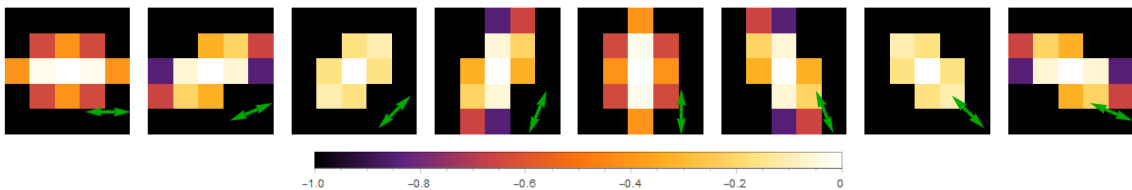


Figure 3.4: 5×5 discrete anisotropic projection kernel for $\lambda = 1/5$, $\zeta = 1/2$ and $p = 4$ on 8 orientations.

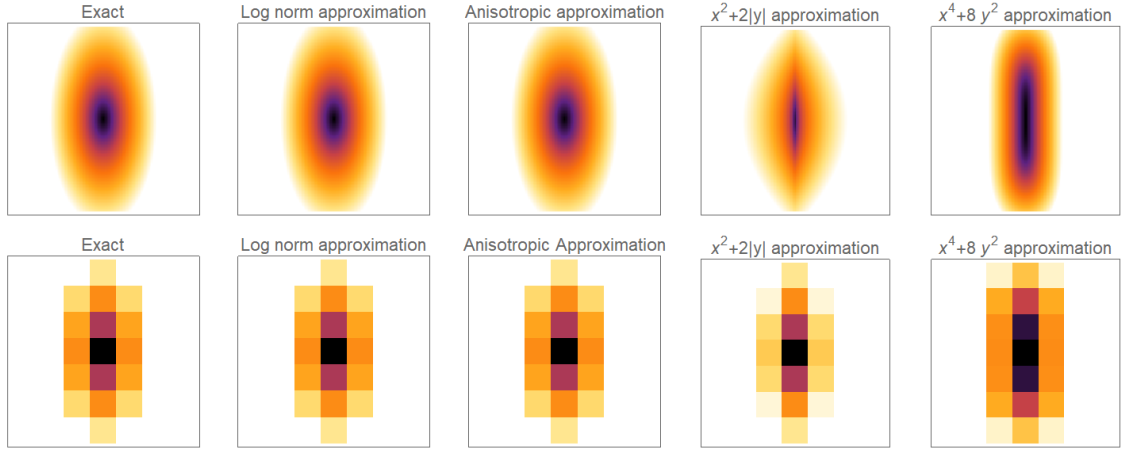


Figure 3.2: Exact projected norm (via numerical solution of the Eikonal PDE, see [39]) versus various approximations (Log norm via (3.21) and anisotropic norm via definition 3.19) both for the continuous case and a 7×7 sampling on $[-1, 1]^2$.

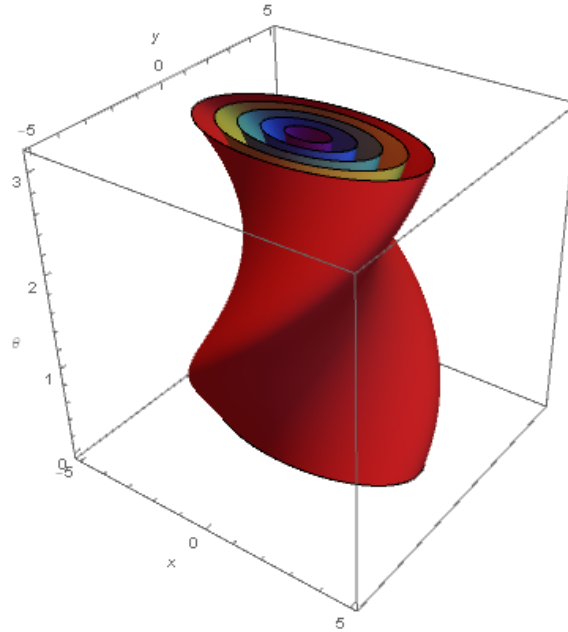


Figure 3.3: Anisotropic projection kernel for $\zeta = \frac{1}{2}$ and $p = 1$.

Remark 3.21. This projection relates to the morphological PDE of the type $u_t = \|\nabla u\|_{\mathcal{G}_\zeta}^\alpha$ where the solution is obtained by the dilation of the initial condition with the following kernel:

$$k_t(g) = -\left(\frac{2\alpha-1}{2\alpha}\right) t^{-\frac{1}{2\alpha-1}} \|g\|_{\mathcal{G}_\zeta}^{\frac{2\alpha}{2\alpha-1}}. \quad (3.23)$$

Our $p = 4$ translates to $\alpha = \frac{2}{3}$ in this formulation. This is one of the links between PDEs and CNNs we aim to explore in future work. See appendix A for more.

3.4 DRIVE Segmentation

We will be benchmarking our networks on the *DRIVE* dataset [40]. This public database contains images of the retina and is often used to test blood vessel segmentation algorithms on.

3.4.1 Networks

We will test three CNNs:

1. a spatial network,
2. a group convolution network with maximum projection,
3. a group convolution network with dilated projection.

The architecture of these networks is specified in figure 3.5, for fairness we ensured they have almost the same amount of parameters. Both group convolution network have 8 orientations, hence why we refer to them as SE(2,8) networks.

The retinal images are split into 128×128 patches and fed into the network in batches of 16. We train each network for 10 epochs with a momentum optimizer with the loss function consisting of the softmax cross-entropy per definition 3.5 plus an L^2 penalty over all trainable coefficients. Additionally batch-normalization [41] is used after every convolution layer.

3.4.2 Results

For a quantitative performance comparison we use two metrics: the loss function and the *area under the ROC curve* [42] (*AUC* for short). See figure 3.6 for a comparison of the three networks we proposed.

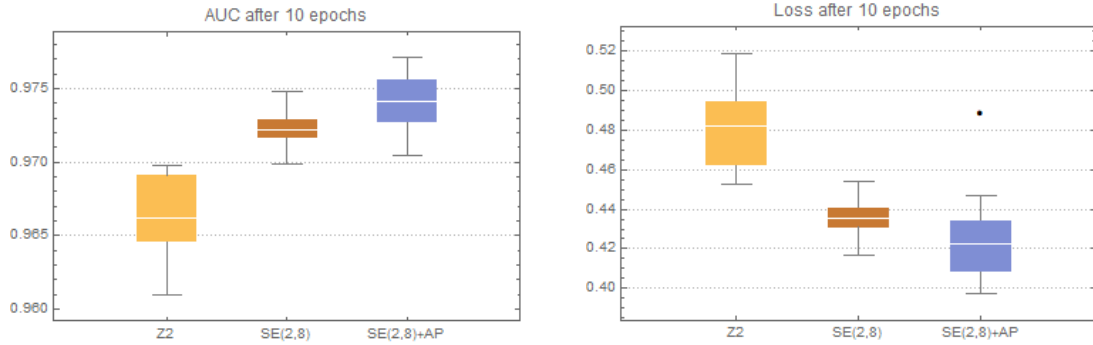


Figure 3.6: AUC (left) and loss function (right) performance comparison, per network we train 15 realizations to 10 epochs and then take the average loss and AUC over 8 validation images (that the network was not trained on).

Observe that while the group network with anisotropic projection yields the best results, the maximum projection network is more stable.

For a qualitative comparison we look at the produced segmentation on a test image (that was not trained on) and compare it with the human segmentation with a difference map, these are shown in figure 3.8 with a false classification summary in table 3.1. The test image alongside the human segmentation is shown in figure 3.7.

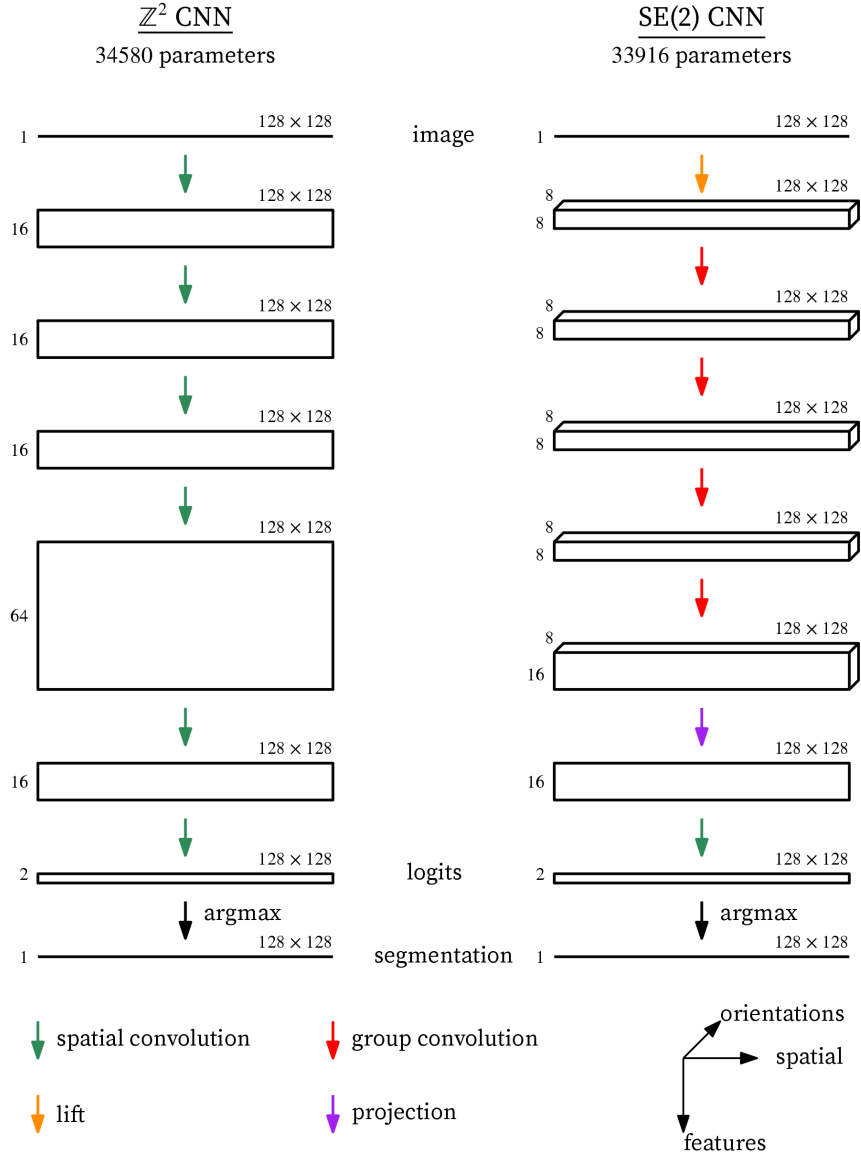


Figure 3.5: Segmentation network architectures.

Model	False positives	False negatives
$\mathbb{Z}^2$	7.22%	0.62%
SE(2,8) with max projection	5.87%	0.62%
SE(2,8) with anisotropic projection	5.79%	0.50%

Table 3.1: False classifications ratios.

All three networks display the following three deficiencies.

1. The right edge of the optical disk is identified as vein.
2. The smallest vessels are not or poorly segmented.
3. Vessels are drawn consistently too bold.

They are distinctive mainly in the amount of “spotting”, this is quite pronounced in the case of the spatial network, the first group projection network reduces this significantly and the anisotropic projection networks cleans up even more, particularly around the edges.

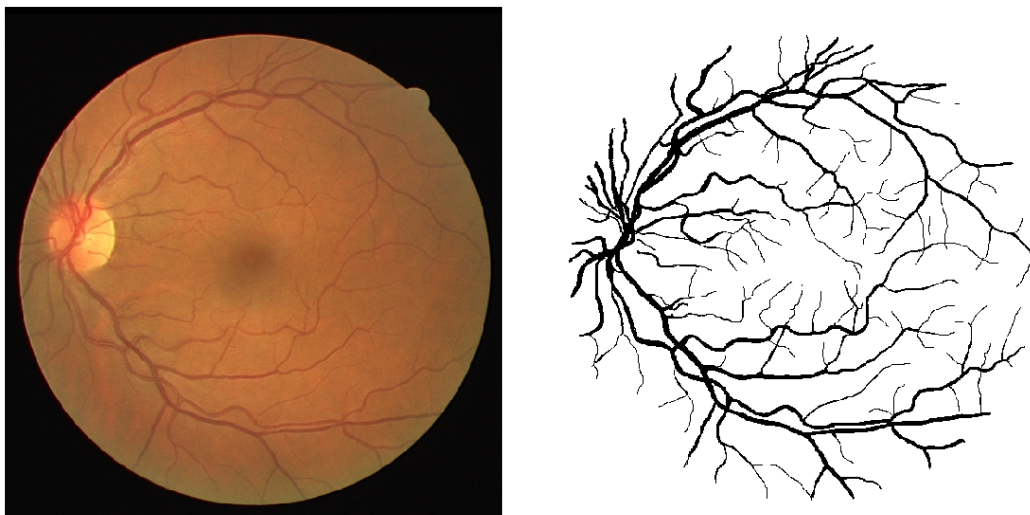
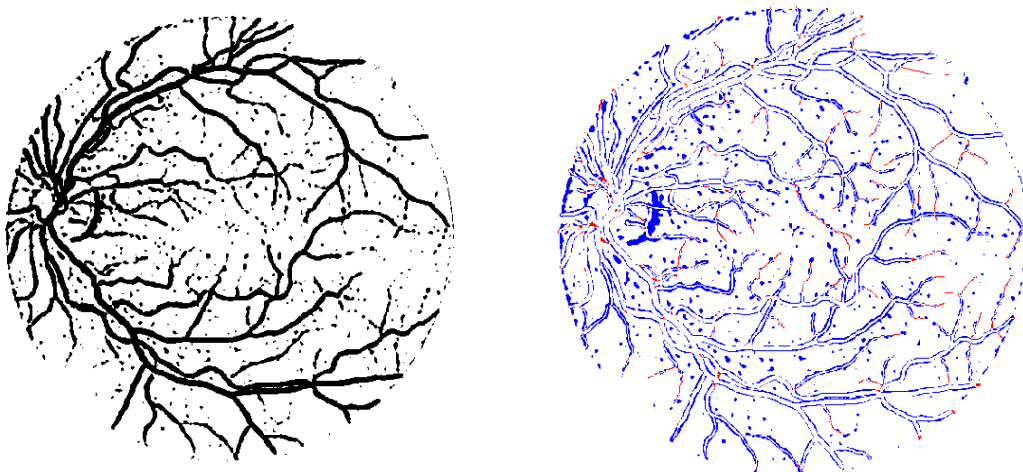
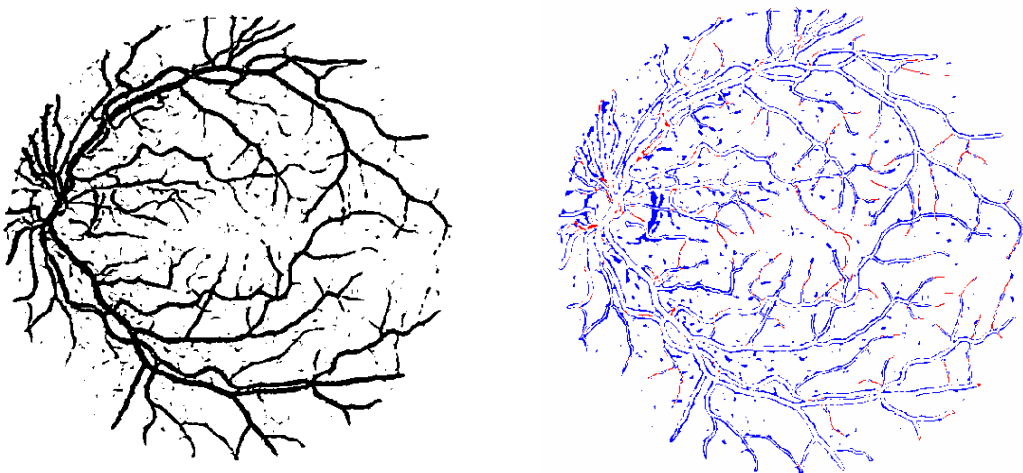


Figure 3.7: Source retinal image (left) with the human produced segmentation (right).

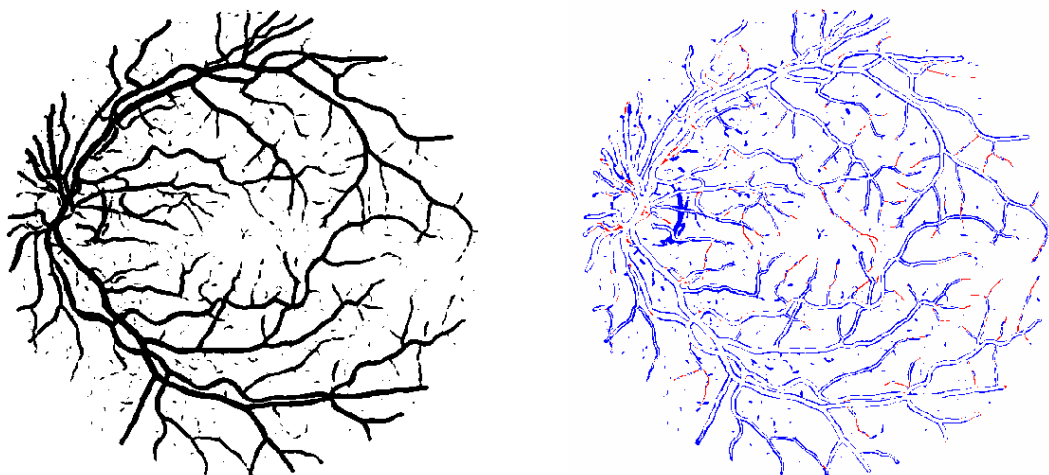
The difference maps suggest a two-fold improvement: longitudinal diffusion/dilation (to extend into the small vessels) and lateral erosion (to reduce the vessels width and sharpen the traces), we leave these ideas for future work.



(a) Z^2 CNN



(b) SE(2,8) CNN with maximum projection



(c) SE(2,8) CNN with anisotropic projection

Figure 3.8: Output of the networks on the left with difference map versus the human segmentation on the right (blue indicates false positive, red false negative).

Appendix A

PDE Guided CNN Design

Consider the following evolution PDE on the d -dimensional space of positions and orientations $\mathbb{M}_d$ equipped with metrics $G_{1/2/3}$ (which need not be distinct).

$$\begin{cases} \frac{\partial W}{\partial t}(p, t) &= \text{convection} \quad \text{fract. diffusion} \quad \text{dilation/erosion} \\ &= -\mathbf{c} \cdot \nabla_{G_1} W(p, t) + |\Delta_{G_2}|^\alpha W(p, t) \pm \|\nabla_{G_3} W(p, t)\|^{2\alpha}, \quad \text{for } p \in \mathbb{M}_d, t \geq 0, \\ W(p, 0) &= U(p), \quad \text{for } p \in \mathbb{M}_d. \end{cases} \quad (\text{A.1})$$

As the labels indicate, the three terms have three distinct effects on the solution: convection, diffusion (fractional or not), and dilation/erosion (depending on the sign). We can apply operator splitting here to look at the convection+diffusion term separately from the morphological term.

The solution to the convection and diffusion part can then be obtained by the convolution of the initial orientation score U with a kernel

$$W(p, t) = (K_t^{G_2, c, \alpha} *_{\mathbb{M}_2} U)(p), \quad (\text{A.2})$$

where exact kernels $K_t^{G_2, c, \alpha}$ have recently been derived (with more tangible approximations) in [2, 43], also generalized to $\mathbb{M}_3$ and $\alpha > 0$ in [44], again with tangible approximations [45].

The viscosity solution to the morphological part is obtained by the dilation or erosion of the initial condition with a morphological kernel. For the + variant (i.e. dilations) in (A.1) without convection and fractional diffusion we get

$$W(p, t) = (k_t \square U)(p) = \sup_{h \in SE(2)} \{k_t(h^{-1}p) + U(h)\}, \quad (\text{A.3})$$

with a negative dilation kernel $k_t(p)$ which for $\alpha = \frac{1}{2}$ is given by

$$k_t(p) = \begin{cases} 0 & \text{if } d_{G_3}(p, e) \leq t, \\ -\infty & \text{else,} \end{cases} \quad (\text{A.4})$$

where d_{G_3} denotes the Riemannian distance by metric tensor field G_3 .

Interestingly we may approximate k_t by

$$k_t^{\text{approx}}(p) = \begin{cases} 0 & \text{if } \|\log g\| \leq t, \\ -\infty & \text{else,} \end{cases} \quad (\text{A.5})$$

where $\|\log g\|$ denotes a logarithmic norm approximation similar to the one in section 1.1.7. See [46, App. E].

We can then numerically approximate the solution by alternately applying the linear convolution and morphological convolution for small time steps.

This iterative method translates to neural networks by putting each operator in a layer. The trainable parameters are then the transport vector $\mathbf{c}$ and the metrics $G_{1/2/3}$. The benefit being that all of these parameters are now left-invariant and geometrically meaningful PDE-coefficients.

The overall architecture of the network this necessitates is similar to the G-CNN we saw in Chapter 3 and it follows the same steps as the PDE based denoising methods from Chapter 2. Contrast the PDE-G-CNN architecture in figure A.1 with the G-CNN architecture in figure 3.5 and the denoising pipeline in figure 1.8.

The anisotropic projection network we experimented with in Chapter 3 was a first tentative test of this idea as our anisotropic projection is a concatenation of a dilation and a maximum projection (recall theorem 3.16). Considering that the results were positive indicates that this idea is worth exploring. Moreover, in previous geometric image processing techniques [45, 47, 15, 48] the PDE in (A.1) was beneficial over its counterparts acting only in the image domain.

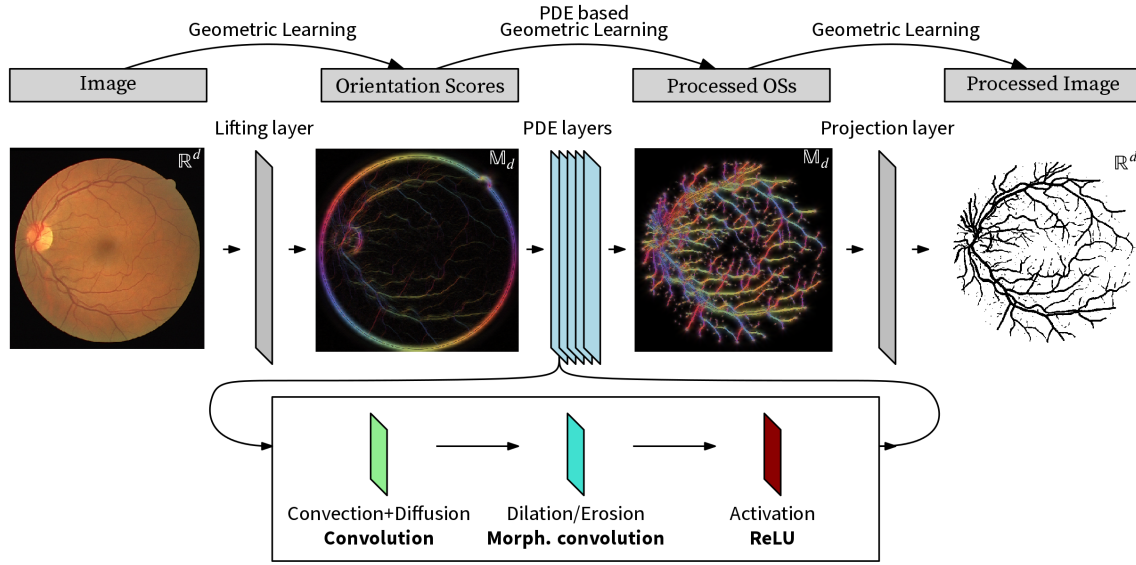


Figure A.1: PDE-G-CNN network architecture.

As we mentioned in the introduction this is a developing idea and there are some open questions, we mention a few.

- How does the ReLU fit in vis-a-vis negative values? (co-domain transform?)
- How does (batch) normalization fit into the PDE?
- Can multiple layers (with presumably differently trained parameters) still be captured by a single PDE or are we looking at a different PDE per layer?
 - If the former: can some limiting argument be made in that case?
 - If the later: are these PDEs still related to each other?
- How are multiple features dealt with?
 - Additionally how are variable number of features per layer dealt with?
- Can more PDEs than (A.1) be considered or can we formulate an axiomatic approach that forces us to use (A.1)?
- How to relate network depth to time evolution.

List of Figures

1	Crossing-preserving denoising	1
2	Invariance of classifying CNNs	2
1.1	Roto-translations can be composed to form a non-commutative group.	3
1.2	Relation between rotation vectors and operators.	4
1.3	Illustrating the structure of the helical atlas.	6
1.4	Types of movements in $\mathbb{M}_2$	6
1.5	Geometric interpretation of the $\mathbb{M}_2$ Lie brackets.	12
1.6	Norm estimates for $D_S = 1$, $D_A = 0.1$, $\zeta = 0.5$ for the Riemannian (left) and sub-Riemannian case (right).	18
1.7	Gauge frame fitting	19
1.8	Image processing pipeline.	22
1.9	Discrete cake wavelets.	23
1.10	Illustrating orientation score visualization.	23
1.11	Demonstrating orientation confidence in the left invariant frame.	24
2.1	Mean curvature flow of a level set.	32
2.2	Relative errors for isotropic setup without coherence enhancement	35
2.3	Qualitative comparison of isotropic flows.	36
2.4	Cross section of isotropic flows without coherence enhancement.	37
2.5	Relative errors for isotropic setup with coherence enhancement	38
2.6	Looking at the effect of noise on gauge frame fitting.	39
2.7	Relative errors for anisotropic ($\zeta = 0.25$) setup without coherence enhancement	39
2.8	Qualitative comparison of anisotropic flows.	40
2.9	Relative errors for anisotropic ($\zeta = 0.25$) setup with coherence enhancement	41
2.10	Relative errors for TVF (isotropic without coherence enhancement) with and without rescale post-processing.	41
2.11	Left invariant TVF without (left) and with (right) rescale post-processing at $t = 4$	42
2.12	Relative errors of 3-stage TVF versus post-processed TVF.	42
2.13	Final result of the 3-stage TVF method.	43
2.14	Example result of the 3-stage TVF method.	43
3.1	Comparing kernel traversal in $\mathbb{R}^3$ versus $\mathbb{M}_2$	50

3.4	5×5 discrete anisotropic projection kernel for $\lambda = \frac{1}{6}$, $\zeta = \frac{1}{2}$ and $p = 4$ on 8 orientations.	53
3.2	Exact projected norm versus various approximations.	54
3.3	Anisotropic projection kernel for $\zeta = \frac{1}{2}$ and $p = 1$	54
3.6	AUC (left) and loss function (right) performance comparison, per network we train 15 realizations to 10 epochs and then take the average loss and AUC over 8 validation images (that the network was not trained on).	55
3.5	Segmentation network architectures.	56
3.7	Source retinal image (left) with the human produced segmentation (right).	57
3.8	Output of the networks on the left with difference map versus the human segmentation on the right (blue indicates false positive, red false negative).	58
A.1	PDE-G-CNN network architecture.	60

List of Tables

2.1	Definition of symbols used in this chapter.	25
2.2	Comparing timescales.	34
2.3	Key parameters for isotropic experiments.	34
2.4	Comparing minimal relative L_1 and L_2 errors for isotropic methods.	38
2.5	Comparing minimal relative L_1 and L_2 errors for anisotropic ($\zeta = 0.25$) methods. .	41
3.1	False classifications ratios.	56

Bibliography

- [1] Cynthia Rudin. “Please stop explaining black box models for high stakes decisions”. In: *arXiv preprint arXiv:1811.10154* (2018).
- [2] Remco Duits and Erik Franken. “Left-invariant parabolic evolutions on (2) and contour enhancement via invertible orientation scores Part II: Nonlinear left-invariant diffusions on invertible orientation scores”. In: *Quarterly of applied mathematics* 68.2 (2010), pp. 293–331.
- [3] Remco Duits, Michiel HJ Janssen, Julius Hannink, and Gonzalo R Sanguinetti. “Locally adaptive frames in the roto-translation group and their applications in medical imaging”. In: *Journal of Mathematical Imaging and Vision* 56.3 (2016), pp. 367–402.
- [4] Jorg M Portegies, Rutger Henri Jacques Fick, Gonzalo R Sanguinetti, Stephan PL Meesters, Gabriel Girard, and Remco Duits. “Improving fiber alignment in HARDI by combining contextual PDE flow with constrained spherical deconvolution”. In: *PloS one* 10.10 (2015), e0138122.
- [5] Antonin Chambolle and Thomas Pock. “Total roto-translational variation”. In: *arXiv preprint arXiv:1709.09953* (2017). URL: <https://arxiv.org/abs/1709.09953>.
- [6] Giovanna Citti, Benedetta Franceschiello, Gonzalo Sanguinetti, and Alessandro Sarti. “Sub-Riemannian mean curvature flow for image processing”. In: *SIAM Journal on Imaging Sciences* 9.1 (2016), pp. 212–237.
- [7] Emre Baspinar and Giovanna Citti. “A diffusion driven curvature flow”. In: *arXiv preprint arXiv:1603.01156* (2016).
- [8] Emre Baspinar, Giovanna Citti, and Alessandro Sarti. “A geometric model of multi-scale orientation preference maps via Gabor functions”. In: *Journal of Mathematical Imaging and Vision* 60.6 (2018), pp. 900–912.
- [9] Erik J Bekkers, Maxime W Lafarge, Mitko Veta, Koen AJ Eppenhof, Josien PW Pluim, and Remco Duits. “Roto-translation covariant convolutional networks for medical image analysis”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2018, pp. 440–448. URL: <https://arxiv.org/abs/1804.03393>.
- [10] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. “Geometric deep learning: going beyond euclidean data”. In: *IEEE Signal Processing Magazine* 34.4 (2017), pp. 18–42.
- [11] Zhiwu Huang, Chengde Wan, Thomas Probst, and Luc Van Gool. “Deep learning on lie groups for skeleton-based action recognition”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 6099–6108.
- [12] Taco S Cohen and Max Welling. “Group equivariant convolutional networks”. In: *Int. Conf. on Machine Learning*. 2016, pp. 2990–2999.
- [13] E Weinan. “A proposal on machine learning via dynamical systems”. In: *Communications in Mathematics and Statistics* 5.1 (2017), pp. 1–11.
- [14] J.M. Lee. *Introduction to Smooth Manifolds*. 2nd ed. Graduate Texts in Mathematics. Springer, 2013. ISBN: 978-1-4419-9981-8. DOI: [10.1007/978-1-4419-9982-5_1](https://doi.org/10.1007/978-1-4419-9982-5_1).
- [15] E.J. Bekker. “Retinal Image Analysis using Sub-Riemannian Geometry in $SE(2)$ ”. PhD Thesis. TU Eindhoven, 2017.

- [16] B.M.N. Smets. “Improving on Vascular Tracking using Sub-Riemannian Geodesics”. Bachelor End Project. TU Eindhoven, 2017. URL: <https://bsmetsjr.bitbucket.io/publication/smets2017bep/>.
- [17] R. Duits, J. Portegies, E. St-Onge, and B.M.N. Smets. “Total Variation and Mean Curvature PDEs on $\mathbb{R}^d \rtimes S^{d-1}$ ”. Submitted 21 dec. 2019 to SSVM (Scale Space and Variational Methods) 2019. Accepted. To appear in LNCS Proceedings. URL: <https://arxiv.org/abs/1902.08145>.
- [18] André Bellaïche. “The tangent space in sub-Riemannian geometry”. In: *Sub-Riemannian geometry*. Springer, 1996, pp. 1–78.
- [19] Erik J Bekkers, Da Chen, and Jorg M Portegies. “Nilpotent approximations of sub-Riemannian distances for fast perceptual grouping of blood vessels in 2D and 3D”. In: *Journal of Mathematical Imaging and Vision* (2018), pp. 1–18.
- [20] Gerald B Folland. “A fundamental solution for a subelliptic operator”. In: *Bulletin of the American Mathematical Society* 79.2 (1973), pp. 373–376.
- [21] Aroldo Kaplan. “Fundamental solutions for a class of hypoelliptic PDE generated by composition of quadratic forms”. In: *Transactions of the American Mathematical Society* 258.1 (1980), pp. 147–153.
- [22] LMJ Florack, BM Ter Haar Romeny, JJ Koenderink, and MA Viergever. “Cartesian differential invariants in scale-space”. In: *Journal of Mathematical Imaging and Vision* 3.4 (1993), pp. 327–348.
- [23] Erik Franken and Remco Duits. “Crossing-preserving coherence-enhancing diffusion on invertible orientation scores”. In: *International Journal of Computer Vision* 85.3 (2009), p. 253.
- [24] Remco Duits, Maurice Duits, Markus van Almsick, and Bart ter Haar Romeny. “Invertible orientation scores as an application of generalized wavelet theory”. In: *Pattern Recognition and Image Analysis* 17.1 (2007), pp. 42–75.
- [25] Antonin Chambolle, Vicent Caselles, Daniel Cremers, Matteo Novaga, and Thomas Pock. “An introduction to total variation for image analysis”. In: *Theoretical foundations and numerical methods for sparse recovery* 9.263–340 (2010), p. 227.
- [26] Vicent Caselles, Antonin Chambolle, and Matteo Novaga. “The discontinuity set of solutions of the TV denoising problem and some extensions”. In: *Multiscale modeling & simulation* 6.3 (2007), pp. 879–894.
- [27] LC Evans, J Spruck, et al. “Motion of level sets by mean curvature. I”. In: *Journal of Differential Geometry* 33.3 (1991), pp. 635–681.
- [28] Robert Haslhofer. “Lectures on mean curvature flow”. In: *arXiv preprint arXiv:1406.7765* (2014). URL: <https://arxiv.org/abs/1406.7765>.
- [29] Frank Martin, Erik Bekkers, and Remco Duits. *Lie Analysis for Medical Image Analysis*. 2017. URL: <http://www.lieanalysis.nl/>.
- [30] Eric J Creusen, Remco Duits, and Tom CJ Dela Haije. “Numerical schemes for linear and non-linear enhancement of DW-MRI”. In: *International Conference on Scale Space and Variational Methods in Computer Vision*. Springer. 2011, pp. 14–25.
- [31] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully convolutional networks for semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.
- [32] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.
- [33] Rob A Dunne and Norm A Campbell. “On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function”. In: *Proc. 8th Aust. Conf. on the Neural Networks, Melbourne*. Vol. 181. Citeseer. 1997, p. 185.
- [34] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012. ISBN: 978-0-471-24195-9.

- [35] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016. ISBN: 978-0-262-03561-3.
- [36] Risi Kondor and Shubhendu Trivedi. “On the Generalization of Equivariance and Convolution in Neural Networks to the Action of Compact Groups”. In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. Stockholmsmässan, Stockholm Sweden: PMLR, July 2018, pp. 2747–2755. URL: <http://proceedings.mlr.press/v80/kondor18a.html>.
- [37] Taco Cohen, Mario Geiger, and Maurice Weiler. “A General Theory of Equivariant CNNs on Homogeneous Spaces”. In: *arXiv preprint arXiv:1811.02017* (2018).
- [38] Wolfgang Arendt and Alexander V Bukhvalov. “Integral representations of resolvents and semigroups”. In: *Forum Mathematicum*. Vol. 6. 6. Walter de Gruyter, Berlin/New York. 1994, pp. 111–136.
- [39] Remco Duits, Stephan PL Meesters, J-M Mirebeau, and Jorg M Portegies. “Optimal paths for variants of the 2D and 3D Reeds–Shepp car with applications in image analysis”. In: *Journal of Mathematical Imaging and Vision* (2018), pp. 1–33.
- [40] Joes Staal, Michael D Abràmoff, Meindert Niemeijer, Max A Viergever, and Bram Van Ginneken. “Ridge-based vessel segmentation in color images of the retina”. In: *IEEE transactions on medical imaging* 23.4 (2004), pp. 501–509. URL: <https://www.isi.uu.nl/Research/Databases/DRIVE/>.
- [41] Sergey Ioffe and Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *arXiv preprint arXiv:1502.03167* (2015).
- [42] Andrew P Bradley. “The use of the area under the ROC curve in the evaluation of machine learning algorithms”. In: *Pattern recognition* 30.7 (1997), pp. 1145–1159.
- [43] Remco Duits and Markus Van Almsick. “The explicit solutions of linear left-invariant second order stochastic evolution equations on the 2D Euclidean motion group”. In: *Quarterly of Applied Mathematics* 66.1 (2008), pp. 27–67.
- [44] Remco Duits, Erik Bekkers, and Alexey Mashtakov. “Fourier transform on the homogeneous space of 3D positions and orientations for exact solutions to linear PDEs”. In: *Entropy* 21.1 (2019), p. 38.
- [45] J Portegies. “PDEs on the Lie Group SE (3)”. PhD thesis. Ph. D. Thesis, Department of Mathematics and Computer Science, Eindhoven.
- [46] Erik J Bekkers, Remco Duits, Alexey Mashtakov, and Gonzalo R Sanguinetti. “A PDE approach to data-driven sub-Riemannian geodesics in SE (2)”. In: *SIAM Journal on Imaging Sciences* 8.4 (2015), pp. 2740–2770.
- [47] Erik M Franken. “Enhancement of crossing elongated structures in images”. In: *Eindhoven University of Technology. Eindhoven, The Netherlands* (2008).
- [48] R. Duits. *Roto-Translation Covariant Convolutional Networks for Medical Image Analysis*. Oct. 5, 2019. URL: <https://www.youtube.com/watch?v=yZ00D5FvDR4>.