The Hessian of Axially Symmetric Functions on SE(3) and Application in 3D Image Analysis

M.H.J. Janssen[1], T.C.J. Dela Haije[1], F.C. Martin[1], E.J. Bekkers[1], R. Duits[1]

[1]Eindhoven University of Technology, The Netherlands,
Department of Mathematics and Computer Science,
M.H.J.Janssen@tue.nl

Abstract. We propose a method for computation of the Hessian of axially symmetric functions on the roto-translation group $SE(3)$. Eigendecomposition of the resulting Hessian is then used for curvature estimation of tubular structures, similar to how the Hessian matrix of 2D or 3D image data can be used for orientation estimation. This paper focuses on a new implementation of a Gaussian regularized Hessian on the roto-translation group. Furthermore we show how eigenanalysis of this Hessian gives rise to exponential curve fits on data on position and orientation (e.g. orientation scores), whose spatial projections provide local fits in 3D data. We quantitatively validate our exponential curve fits by comparing the curvature of the spatially projected fitted curve to ground truth curvature of artificial 3D data. We also show first results on real MRA data. Implementations are available at: http://lieanalysis.nl/orientationscores.html.

1 Introduction

The Hessian matrix of 2D or 3D image data is commonly used to analyze the local structure of the data. Eigendecomposition of the Hessian matrix is common for orientation estimation and ridge detection, and eigenvalues have been used in features such as vesselness [8]. We aim to extend such techniques to functions on positions and orientations $\mathbb{R}^3 \times S^2$ which can be identified with axially symmetric functions on the 3D roto-translation group $SE(3)$. This paper focuses on the implementation of the Hessian of real-valued axially symmetric functions on $SE(3)$.

Data on positions and orientations can be obtained in several different settings. Here we focus on orientation scores, where data is obtained from 3D data by an invertible wavelet-type transformation [9]. Other options include liftings [12, 2], and diffusion-weighted MRI [10]. In general $U(\mathbf{x}, \mathbf{n}) : \mathbb{R}^3 \times S^2 \to \mathbb{R}$ is to be considered a probability density of finding a local oriented structure (i.e. an elongated structure) at position $\mathbf{x} \in \mathbb{R}^3$ with orientation $\mathbf{n} \in S^2$.

Just as in 2D and 3D data, eigendecomposition of the Hessian produces the directions of principle curvature with corresponding curvature values. When considering 3D data the eigenvector with smallest eigenvalue is often used to find the orientation of a line structure. When considering the direction of a structure

in the extended space of positions and orientations this does not only give us information about spatial orientation of the structure but also about the change in orientation which is directly related to curvature. This technique has already been used for curvature estimation of blood vessels in 2D data [1], and here we extend this technique to 3D.

Local curve optimization that accounts for local curvature via spiral fits has been proposed in [11], where a more complex model of neighboring curves is fitted. We have a simpler curve fit model (e.g. we do not account for fanning out). Our model has the advantage that we obtain curve fits as exact solutions of Euler-Lagrange equations and use the full-distribution on $\mathbb{R}^3 \times S^2$ instead of only relying on the principle eigenvector of the diffusion tensor.

Regarding the implementation of the Hessian matrix, we rely on Gaussian derivatives. For angular derivatives, finite derivatives were used in other works [5]. On the sphere S^2, limited amount of samples due to computation time and non uniform sampling can cause problems such as bias towards sampled orientations and additional numerical blur. Furthermore, adding regularization is not trivial when using finite differences and such implementations do not allow us to calculate derivatives at orientations which do not lie on the sampling grid. Especially this last shortcoming is problematic in practice. To solve this problem we express our spherical data in spherical harmonics after which the computation of the Gaussian Hessian is exact.

Theory on how eigenanalysis of the Hessian relates to exponential curve fits can be found in [6], where orientation estimation was extended to the space of positions and orientations $\mathbb{R}^3 \times S^2$. There a Hessian on the extended space $SE(3)$ was used but no details on implementation were given. The key objective of this article is to present a new algorithm for calculating the Hessian of axially symmetric functions on $SE(3)$ via spherical harmonics. This is useful for exponential curve fits and curvature estimation of elongated structures (e.g. vessels) in 3D data. Furthermore, we address validation of the exponential curve fits and the induced curvature measurements. First we provide background theory in Section 2, followed by a discussion of the discrete implementation of the Hessian in Section 3. Then we include the algorithm for curvature estimation in Section 4. Finally we quantitatively validate our method on artificial 3D data with ground truth curvatures and show first results on real MRA data in Section 5.

2 Theory

2.1 Embedding of $\mathbb{R}^3 \times S^2$ Data in SE(3)

When processing data on positions and orientations it is often necessary to equip the domain with a structure that links the data across different orientation channels, in such a way that a notion of alignment between local orientations is taken into account. This is achieved by embedding data on positions and orientations into the roto-translation group $SE(3) = \mathbb{R}^3 \rtimes SO(3)$, with group product $g_1 g_2 = (\mathbf{x}_1, \mathbf{R}_1)(\mathbf{x}_2, \mathbf{R}_2) = (\mathbf{R}_1 \mathbf{x}_2 + \mathbf{x}_1, \mathbf{R}_1 \mathbf{R}_2)$, with $g_1, g_2 \in SE(d)$.

Let $U : \mathbb{R}^3 \rtimes S^2 \to \mathbb{R}$ denote a function on the coupled space of positions and orientations. Then, its embedding $\tilde{U} : SE(3) \to \mathbb{R}$ is given by:

$$\tilde{U}(\mathbf{x}, \mathbf{R}) := U(\mathbf{x}, \mathbf{R}\mathbf{e}_z) \tag{1}$$

for all $\mathbf{x} \in \mathbb{R}^3$ and all rotations $\mathbf{R} \in SO(3)$. This embedding results in the axial symmetry in $\tilde{U}$ as a replacement $\mathbf{R} \mapsto \mathbf{R}\mathbf{R}_{\mathbf{e}_z,\alpha}$ does not affect $\tilde{U}$ in Eq. (1).

Throughout the document we will use the z-y-z Euler angles convention $\mathbf{R} = \mathbf{R}_{e_z,\gamma}\mathbf{R}_{e_y,\beta}\mathbf{R}_{e_z,\alpha}$, with $\alpha \in [-\pi, \pi)$, $\beta \in [0, \pi]$ and $\gamma \in [-\pi, \pi)$. Just as in [6] we will use $\mathbf{R}_\mathbf{n}$ to denote any rotation such that $\mathbf{R}_\mathbf{n}\mathbf{e}_z = \mathbf{n}$. Due to the data symmetry induced by embedding according to Eq. (1), the choice of $\mathbf{R}_\mathbf{n}$ does not matter. Therefore, we can choose for each orientation a specific rotation. We set $\alpha = 0$ and define $\mathbf{R}^0_\mathbf{n} = \mathbf{R}_{e_z,\gamma}\mathbf{R}_{e_y,\beta}$, where β and γ are standard spherical coordinates for the orientation: $\mathbf{n}(\beta, \gamma) = (\sin\beta\cos\gamma, \sin\beta\sin\gamma, \cos\beta)$.

2.2 The Hessian Matrix on SE(3)

By Eq. (1) we relate data U on positions and orientations to data $\tilde{U}$ on $SE(3)$. This is helpful to: 1) keep track of rotation and translation covariance as this boils down to left-invariance on the group. 2) work with moving frames of reference for each position and each orientation. Next, we explain the notion of left-invariance, followed by explicit formulas for the left-invariant vector fields on $SE(3)$ which are then used in the definition of the Hessian.

A vector field $g \mapsto \mathcal{A}|_g$ on $SE(3)$ is left-invariant if for all differentiable curves $\gamma : \mathbb{R} \mapsto SE(3)$ one has

$$\gamma'(0) = \mathcal{A}|_{\gamma(0)} \Rightarrow \forall_{g \in SE(3)}\ (g\gamma)'(0) = \mathcal{A}|_{g\gamma(0)}, \tag{2}$$

see Fig. 1 (a) for a geometric explanation. The left-invariant vector fields are obtained via push-forward of the left multiplication $\mathcal{A}_i|_g = (L_g)_* \mathcal{A}_i|_e$ of the Lie algebra basis $\mathcal{A}_i|_\mathfrak{e}$. On $SE(3)$ the left-invariant vector fields are given by

$$\begin{pmatrix} \mathcal{A}_1 \\ \mathcal{A}_2 \\ \mathcal{A}_3 \end{pmatrix} = \mathbf{R}^T \begin{pmatrix} \partial_x \\ \partial_y \\ \partial_z \end{pmatrix}, \quad \begin{pmatrix} \mathcal{A}_4 \\ \mathcal{A}_5 \\ \mathcal{A}_6 \end{pmatrix} = \begin{pmatrix} \cos\alpha\cot\beta\ \partial_\alpha + \sin\alpha\ \partial_\beta - \frac{\cos\alpha}{\sin\beta}\partial_\gamma \\ -\sin\alpha\cot\beta\ \partial_\alpha + \cos\alpha\ \partial_\beta + \frac{\sin\alpha}{\sin\beta}\partial_\gamma \\ \partial_\alpha, \end{pmatrix}, \tag{3}$$

for $\beta \neq 0, \beta \neq \pi$, with $\mathbf{x} = (x, y, z)$ and recall that $\mathbf{R} = \mathbf{R}_{\mathbf{e}_z,\gamma}\mathbf{R}_{\mathbf{e}_y,\beta}\mathbf{R}_{\mathbf{e}_z,\alpha}$.

For defining the regularized Hessian matrix we use the following regularization kernel:

$$\tilde{G}_\mathbf{s}(g) := G^{\mathbb{R}^3}_{s_p}(\mathbf{x})\, G^{S^2}_{s_o}(\mathbf{R}\mathbf{e}_z), \tag{4}$$

where $G^{\mathbb{R}^3}_{s_p}$ is the diffusion kernel on $\mathbb{R}^3$ and $G^{S^2}_{s_o}$ the diffusion kernel on the sphere S^2, and where $\mathbf{s} = (s_p, s_o)$, $s_p, s_o \geq 0$, are the spatial and angular scales of regularization respectively. Then, the 6×6 non-symmetric regularized Hessian[1]

[1] In general the Hessian depends on the imposed connection on the tangent bundle $T(SE(3))$, where $\mathbf{H} = (\nabla^*_{\mathcal{A}_i} d\tilde{U})(\mathcal{A}_j)$. Here we follow [6, App. 4] and choose ∇ as the left Cartan connection, since it is the correct connection for left-invariant processing.

matrix at $g = (\mathbf{x}, \mathbf{R})$ is defined by

$$(\mathbf{H}^{\mathbf{s}}\tilde{U})(g) := (\mathbf{H}\tilde{V})(g) := [\mathcal{A}_j\mathcal{A}_i(\tilde{V})(g)]_{i,j=1}^{6}\,, \quad \text{with } \tilde{V} = \tilde{G}_{\mathbf{s}} * \tilde{U}, \tag{5}$$

and where i, j denote the row and column index respectively. Here $*$ denotes convolution on the group $SE(3)$.

2.3 Exponential Curves on SE(3) and Spatial Projection

An exponential curve in $SE(3)$ is a curve obtained by the exponential mapping from Lie algebra to Lie group. For $g \in SE(3)$ and $t \in \mathbb{R}$ we write

$$\gamma_g^{\mathbf{c}}(t) = (\mathbf{x}(t), \mathbf{R}(t)) = g\, e^{t\sum_{i=1}^{6} c^i A_i}, \tag{6}$$

with $A_i = \mathcal{A}_i|_{\mathfrak{e}}$ denoting a basis of the Lie algebra $T_{\mathfrak{e}}(SE(d))$ and $\mathbf{c}^T = (\mathbf{c}^{(1)}, \mathbf{c}^{(2)})^T = (c^1, \ldots, c^6)^T \in \mathbb{R}^6$ be a given column vector, where $\mathbf{c}^{(1)} = (c^1, c^2, c^3) \in \mathbb{R}^3$ denotes the spatial velocity components and $\mathbf{c}^{(2)} = (c^4, c^5, c^6) \in \mathbb{R}^3$ denotes the rotational velocity components. The exponential curve has the crucial property that the components of the tangent vector expressed in the left-invariant basis are constant over the entire parametrization:

$$\dot{\gamma}_g^{\mathbf{c}}(t) = \sum_{i=1}^{6} c^i\ \mathcal{A}_i|_{\gamma_g^{\mathbf{c}}(t)}\,, \tag{7}$$

see Fig. 1 (b). For formulae of the exponential curves in $SE(3)$ see [4]. Their spatial parts are circular spirals with constant curvature and torsion magnitude:

$$\|\boldsymbol{\kappa}\| = \frac{\|\mathbf{c}^{(1)} \times \mathbf{c}^{(2)}\|}{\|\mathbf{c}^{(1)}\|^2} \text{ and } \|\boldsymbol{\tau}\| = \frac{|\mathbf{c}^{(1)} \cdot \mathbf{c}^{(2)}| \cdot \|\boldsymbol{\kappa}\|}{\|\mathbf{c}^{(1)}\|}. \tag{8}$$

2.4 Exponential Curve Fits

The exponential curve fitting procedure minimizes

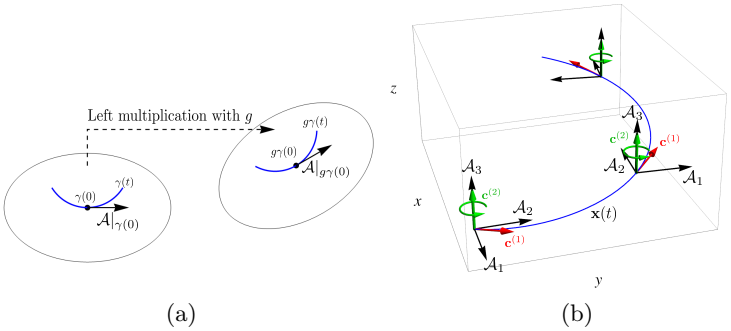
$$\begin{aligned} \mathbf{c}^*(g) &= \underset{\mathbf{c}\in\mathbb{R}^6, \|\mathbf{c}\|_\mu = 1}{\arg\min} \left\| \frac{d}{dt} \nabla\tilde{V}(\gamma_g^{\mathbf{c}}(t))\Big|_{t=0} \right\|_\mu^2, \\ \|\mathbf{c}\|_\mu &:= \|\mathbf{M}_\mu \mathbf{c}\|, \quad \mathbf{M}_\mu = \begin{pmatrix} \mu\mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_3 \end{pmatrix} \in \mathbb{R}^{6\times 6}, \end{aligned} \tag{9}$$

where $\mu > 0$ is a parameter used to balance spatial and angular distances. Its relation to the Hessian is that the normalized eigenvector $\mathbf{M}_\mu \mathbf{c}^*(g)$ with smallest eigenvalue of symmetric Hessian product matrix

$$\mathbf{M}_\mu^{-1}(\mathbf{H}\tilde{V}(g))^T \mathbf{M}_\mu^{-2} \mathbf{H}\tilde{V}(g) \mathbf{M}_\mu^{-1} \tag{10}$$

provides the solution $\mathbf{c}^*(g)$ of optimization problem (9), see [6, Theorem 3] which shows the $SE(2)$ version of this problem. In contrast to [6], we do not use *external*

regularization. In [6] this result for $SE(2)$ was not extended to $SE(3)$ because the optimization problem does not have a unique solution due to a non-trivial null-space of the matrix caused by the symmetry in our data. In practice however one needs to rely on a two-step method as explained in [6], where we first fit the optimal spatial tangent and then fit the curvature. The fact that the two-step method avoids the problems caused by the null-space was overlooked in [6]. Details follow in Section 4.



(a) (b)

Fig. 1. (a) Schematic visualization of left-invariant vector fields, Eq.(2). A vector viewed as a tangent to a curve is used to construct a vector field by moving the curve to a different group element by group multiplication. (b) An exponential curve $\gamma_g^{\mathbf{c}}(t)$, Eq. (6). The spatial part is a spiral (blue line). Its tangent $\dot{\gamma}_g^{\mathbf{c}}(t)$, Eq. (7), has constant components w.r.t. the left-invariant vector fields $\mathcal{A}_i$ which act as a local frame of reference. Here $\mathbf{c}^{(1)}$ are components of the spatial tangent (red arrow) and $\mathbf{c}^{(2)}$ components of the angular tangent, shown here as the axis of rotation (green arrow).

3 Implementation on Discrete Data

In practice we have discrete data on positions and orientations (e.g. a discrete orientation score) defined on a 3D rectangular grid parametrized by $\mathbf{x} = (x, y, z)$ with $x = 1, 2, \ldots, N_x$, $y = 1, 2, \ldots, N_y$ and $z = 1, 2, \ldots, N_z$, which is defined for a set of discrete orientations $\mathbf{n}_i = \mathbf{n}(\beta_i, \gamma_i)$ labeled by $i = 1, 2, \ldots, N_o$.

We will now focus on computing $\mathbf{H}\tilde{V}(g)$ for a group element $g = (\mathbf{x}, \mathbf{R}_{\mathbf{n}}^0)$. Note that $\mathbf{n} = \mathbf{R}\mathbf{e}_z$ may not be on the spherical grid requiring interpolation. Next we do this interpolation via spherical harmonics.

3.1 Signal Approximation with Spherical Harmonic

First we will approximate the signal by expressing it in a spherical harmonic basis. The spherical harmonics are given by

$$Y_l^m(\beta, \gamma) = \sqrt{\frac{2l+1}{4\pi}} \sqrt{\frac{(l-m)!}{(l+m)!}} e^{im\gamma} P_l^m(\cos\beta), \tag{11}$$

where P_l^m is the associated Legendre function and with integer order $l \geq 0$ and integer phase factor $-l \leq m \leq l$. When doing computations it is more convenient to use an alternative indexing j for the basis functions, and we define:

$$\begin{aligned} Y_j = Y_m^l, \qquad & \text{with } j := j(l,m) = (l^2 + l + 1) + m, \\ & l_j = \lfloor \sqrt{j-1} \rfloor, \quad m_j = j - 1 - (l_j)^2 - l_j. \end{aligned} \tag{12}$$

We approximate the data U at a discrete position $\mathbf{x}$ by

$$U(\mathbf{x}, \mathbf{n}(\beta,\gamma)) = \sum_{j=1}^{J} c_j(\mathbf{x})\, Y_j(\beta,\gamma), \tag{13}$$

where $J = j(L,L)$, with L the maximum spherical harmonic order considered chosen such that $J < N_o$ and J close to N_o. Let $\mathbf{u}(\mathbf{x}) = (U(\mathbf{x},\mathbf{n}_1), \ldots, U(\mathbf{x},\mathbf{n}_{N_o}))^T$ be the vector containing the data at position $\mathbf{x}$, $\mathbf{c}(\mathbf{x}) = (c_1(\mathbf{x}), \ldots, c_J(\mathbf{x}))^T$ the coefficients of our approximation at $\mathbf{x}$, and $\mathbf{M}$ the matrix

$$\mathbf{M} = \begin{pmatrix} Y_1(\beta_1,\gamma_1) & Y_2(\beta_1,\gamma_1) & \cdots & Y_J(\beta_1,\gamma_1) \\ \vdots & \vdots & \ddots & \vdots \\ Y_1(\beta_{N_o},\gamma_{N_o}) & Y_2(\beta_{N_o},\gamma_{N_o}) & \cdots & Y_J(\beta_{N_o},\gamma_{N_o}) \end{pmatrix} \in \mathbb{C}^{N_o \times J}, \tag{14}$$

then we have for each position $\mathbf{x}$ the overdetermined system $\mathbf{u} = \mathbf{M}\mathbf{c}$ and we can use a least square fitting procedure to find the coefficients $\mathbf{c}$, i.e. $\mathbf{c}(\mathbf{x}) = (\mathbf{M}^\dagger\mathbf{M})^{-1}\mathbf{M}^\dagger\mathbf{u}(\mathbf{x})$. We now have a description of the data for all (β,γ) but still for a discrete grid of positions.

3.2 Angular Regularization

Because of the choice of regularization kernel $\tilde{G}$ in Eq. (4) we can separately perform angular and isotropic spatial regularization and it does not matter which is performed first. First we apply angular regularization. Since spherical harmonics are eigenfunctions of the Laplace operator, i.e. $\Delta_{S^2} Y_m^l = -l(l+1)Y_m^l$, the regularized signal is given by

$$(G_{s_o}^{S^2} *_{S^2} U)(\mathbf{x}, \mathbf{n}(\beta,\gamma)) = \sum_{j=1}^{J} c_j^{s_o}(\mathbf{x})\, Y_j(\beta,\gamma), \quad \text{with } c_j^{s_o} = c_j e^{-l_j(l_j+1)s_o}. \tag{15}$$

3.3 Spatial Regularization and Spatial Gaussian Derivatives

For all spatial derivatives we use Gaussian derivatives which are evaluated on the spherical harmonic coefficients, for example

$$(\partial_x G_{s_p}^{\mathbb{R}^3} *_{\mathbb{R}^3} U)(\mathbf{x},\beta,\gamma) = \sum_{j=1}^{J} (\partial_x G_{s_p}^{\mathbb{R}^3} *_{\mathbb{R}^3} c_j)(\mathbf{x})\, Y_j(\beta,\gamma). \tag{16}$$

Analogous procedures are used for all spatial derivatives on the coefficients $c_j^{s_o}$. For a group element $g = (\mathbf{x}, \mathbf{R}_{\mathbf{n}(\beta,\gamma)})$ we have the following spatial gradient and Hessian of regularized data $\tilde{V}$ in x-y-z coordinates

$$\nabla_{\mathbb{R}^3}\tilde{V}(g) = \sum_{j=1}^{J} \mathbf{c}_j^{\nabla_{\mathbb{R}^3}}(\mathbf{x})\, Y_j(\beta,\gamma), \quad \text{with } \mathbf{c}_j^{\nabla_{\mathbb{R}^3}} = \begin{pmatrix} \partial_x G_{s_p}^{\mathbb{R}^3} *_{\mathbb{R}^3} c_j^{s_o} \\ \partial_y G_{s_p}^{\mathbb{R}^3} *_{\mathbb{R}^3} c_j^{s_o} \\ \partial_z G_{s_p}^{\mathbb{R}^3} *_{\mathbb{R}^3} c_j^{s_o} \end{pmatrix}, \tag{17}$$

$$(\mathbf{H}_{\mathbb{R}^3}\tilde{V})(g) = \sum_{j=1}^{J} \mathbf{C}_j^{H_{\mathbb{R}^3}}(\mathbf{x})\, Y_j(\beta,\gamma), \text{ with } \mathbf{C}_j^{H_{\mathbb{R}^3}} = [\partial_{x_i}\partial_{x_j} G_{s_p}^{\mathbb{R}^3} *_{\mathbb{R}^3} c_j^{s_o}]_{i,j=1}^{3}, \tag{18}$$

where $(x_1, x_2, x_3) = (x, y, z)$.

For computing the angular derivatives (e.g. $\mathcal{A}_4\mathcal{A}_4\tilde{V}$) we apply the same amount of spatial regularization:

$$\tilde{V}(g) = \sum_{j=1}^{J} c_j^{\tilde{V}}(\mathbf{x})\, Y_j(\beta,\gamma), \qquad \text{with } c_j^{\tilde{V}} = G_{s_p}^{\mathbb{R}^3} *_{\mathbb{R}^3} c_j^{s_o}. \tag{19}$$

3.4 Computation of the Hessian

For computation of the Hessian we use the commutator relations and the fact that by the invariance due to construction (1) we have $\mathcal{A}_6\tilde{V} = 0$ such that

$$\mathbf{H}\tilde{V} = \begin{pmatrix} \mathcal{A}_1^2 & \mathcal{A}_2\mathcal{A}_1 & \mathcal{A}_3\mathcal{A}_1 & \mathcal{A}_1\mathcal{A}_4 & \mathcal{A}_1\mathcal{A}_5 & \mathcal{A}_2 \\ \mathcal{A}_1\mathcal{A}_2 & \mathcal{A}_2^2 & \mathcal{A}_3\mathcal{A}_2 & \mathcal{A}_2\mathcal{A}_4 & \mathcal{A}_2\mathcal{A}_5 & -\mathcal{A}_1 \\ \mathcal{A}_1\mathcal{A}_3 & \mathcal{A}_2\mathcal{A}_3 & \mathcal{A}_3^2 & \mathcal{A}_3\mathcal{A}_4 - \mathcal{A}_2 & \mathcal{A}_3\mathcal{A}_5 + \mathcal{A}_1 & 0 \\ \mathcal{A}_1\mathcal{A}_4 & \mathcal{A}_2\mathcal{A}_4 & \mathcal{A}_3\mathcal{A}_4 & \mathcal{A}_4^2 & \mathcal{A}_5\mathcal{A}_4 & \mathcal{A}_5 \\ \mathcal{A}_1\mathcal{A}_5 & \mathcal{A}_2\mathcal{A}_5 & \mathcal{A}_3\mathcal{A}_5 & \mathcal{A}_4\mathcal{A}_5 & \mathcal{A}_5^2 & -\mathcal{A}_4 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tilde{V} \tag{20}$$

recall Eq. (5). Due to the non-zero commutators we have for example $(\mathcal{A}_4\mathcal{A}_3)\tilde{V} = (\mathcal{A}_3\mathcal{A}_4 - \mathcal{A}_2)\tilde{V}$ which explains the $(3,4)$ entry in the matrix. The derivatives can be split into three categories: spatial derivatives $(\mathcal{A}_3^2, \mathcal{A}_1, \mathcal{A}_2)$, angular derivatives $(\mathcal{A}_4\mathcal{A}_5 = \mathcal{A}_5\mathcal{A}_4, \mathcal{A}_4\mathcal{A}_4, \mathcal{A}_5\mathcal{A}_5)$ and mixed derivatives $(\mathcal{A}_3\mathcal{A}_4, \mathcal{A}_3\mathcal{A}_5)$.

Spatial Derivatives For each $g = (\mathbf{x}, \mathbf{R}_{\mathbf{n}}^0)$ the spatial Hessian (rows and columns 1 to 3 of Eq. (20)) is given by

$$(\mathbf{H}^{spat}\tilde{V})(g) = (\mathbf{R}_{\mathbf{n}}^0)^T (\mathbf{H}_{\mathbb{R}^3}\tilde{V})(g)\mathbf{R}_{\mathbf{n}}^0, \tag{21}$$

where the spatial Hessian in x-y-z coordinates $\mathbf{H}_{\mathbb{R}^3}\tilde{V}$ at g is calculated from the coefficients $\mathbf{C}_j^{H_{\mathbb{R}^3}}(\mathbf{x}) \in \mathbb{R}^{3\times 3}$ according to Eq. (18). Likewise, we have

$$\begin{pmatrix} \mathcal{A}_1\tilde{V}(g) \\ \mathcal{A}_2\tilde{V}(g) \\ \mathcal{A}_3\tilde{V}(g) \end{pmatrix} = (\mathbf{R}_{\mathbf{n}}^0)^T \nabla_{\mathbb{R}^3}\tilde{V}(g), \tag{22}$$

for the first order spatial derivatives. More specifically we have $\mathcal{A}_3 = \mathbf{n} \cdot \nabla_{\mathbb{R}^3}$.

Angular Derivatives For rotations of the form $\mathbf{R}_{\mathbf{n}}^0$ the angular derivatives can be expressed in partial derivatives with respect to standard spherical coordinates (β, γ), recall Eq. (3) for $\alpha = 0$ and using Eq. (19):

$$\begin{aligned}
\mathcal{A}_4 \tilde{V}(\mathbf{x}, \mathbf{R}_{\mathbf{n}}^0) &= \frac{-1}{\sin\beta} \partial_\gamma \tilde{V}(\mathbf{x}, \mathbf{R}_{\mathbf{n}}^0) = \sum\nolimits_{j=1}^{J} c_j^{\tilde{V}}(\mathbf{x}) \frac{-1}{\sin\beta} \partial_\gamma Y_j(\beta, \gamma), \\
\mathcal{A}_5 \tilde{V}(\mathbf{x}, \mathbf{R}_{\mathbf{n}}^0) &= \partial_\beta \tilde{V}(\mathbf{x}, \mathbf{R}_{\mathbf{n}}^0) = \sum\nolimits_{j=1}^{J} c_j^{\tilde{V}}(\mathbf{x}) \partial_\beta Y_j(\beta, \gamma),
\end{aligned} \tag{23}$$

$$\begin{aligned}
\mathcal{A}_4\mathcal{A}_4 \tilde{V}(\mathbf{x}, \mathbf{R}_{\mathbf{n}}^0) &= \sum\nolimits_{j=1}^{J} c_j^{\tilde{V}}(\mathbf{x}) \left(\frac{1}{\sin^2\beta} \partial_\gamma \partial_\gamma + \frac{1}{\tan\beta} \partial_\beta \right) Y_j(\beta, \gamma), \\
\mathcal{A}_4\mathcal{A}_5 \tilde{V}(\mathbf{x}, \mathbf{R}_{\mathbf{n}}^0) &= \sum\nolimits_{j=1}^{J} c_j^{\tilde{V}}(\mathbf{x}) \left(\frac{-1}{\sin\beta} \partial_\gamma \partial_\beta + \frac{\cos\beta}{\sin^2\beta} \partial_\gamma \right) Y_j(\beta, \gamma), \\
\mathcal{A}_5\mathcal{A}_5 \tilde{V}(\mathbf{x}, \mathbf{R}_{\mathbf{n}}^0) &= \sum\nolimits_{j=1}^{J} c_j^{\tilde{V}}(\mathbf{x}) \partial_\beta \partial_\beta Y_j(\beta, \gamma),
\end{aligned} \tag{24}$$

recall that $\mathcal{A}_5\mathcal{A}_4 = \mathcal{A}_4\mathcal{A}_5$. Then, using Lemma 2 and Lemma 4 from [7] we eliminate the pole at $\beta = 0$ and obtain:

$$\begin{aligned}
\mathcal{A}_4 Y_l^m &= a_l^m \tilde{Y}_{l-1}^{m,-1} + a_l^{-m} \tilde{Y}_{l-1}^{m,1}, \\
\mathcal{A}_5 Y_l^m &= b_l^m \tilde{Y}_l^{m,1} - b_l^{-m} \tilde{Y}_l^{m,-1}, \\
\mathcal{A}_4\mathcal{A}_4 Y_l^m &= -l(l+1) Y_l^m - \mathcal{A}_5\mathcal{A}_5 Y_l^m, \\
\mathcal{A}_4\mathcal{A}_5 Y_l^m &= h_l^m Y_{l-1}^m - g_l^{-m} \tilde{Y}_{l-1}^{m,-2} + g_l^m \tilde{Y}_{l-1}^{m,2}, \\
\mathcal{A}_5\mathcal{A}_5 Y_l^m &= k_l^m Y_l^m + j_l^{-m} \tilde{Y}_l^{m,-2} + j_l^m \tilde{Y}_l^{m,2},
\end{aligned} \tag{25}$$

with $\tilde{Y}_l^{m,p} = e^{-ip\gamma} Y_l^{m+p}$ and the following factors:

$$\begin{aligned}
a_l^m &= \frac{i}{2} \sqrt{\frac{2l+1}{2l-1}} \sqrt{(l+m)(l+m-1)}, & b_l^m &= \frac{1}{2} \sqrt{(l-m)(l+m+1)}, \\
h_l^m &= a_l^m b_{l-1}^{m-1} - a_l^{-m} b_{l-1}^{-m-1}, & g_l^m &= a_l^{-m} b_{l-1}^{m+1}, \\
k_l^m &= -b_l^{-m} b_l^{m-1} - b_l^m b_l^{-m-1}, & j_l^m &= b_l^m b_l^{m+1}.
\end{aligned}$$

Mixed Derivatives Finally, the mixed derivatives are computed by

$$\begin{pmatrix} \mathcal{A}_1\mathcal{A}_i \tilde{V}(g) \\ \mathcal{A}_2\mathcal{A}_i \tilde{V}(g) \\ \mathcal{A}_3\mathcal{A}_i \tilde{V}(g) \end{pmatrix} = (\mathbf{R}_{\mathbf{n}}^{0\,T} \nabla_{\mathbb{R}^3} \mathcal{A}_i \tilde{V})(g) = (\mathbf{R}_{\mathbf{n}}^{0\,T} \mathcal{A}_i \nabla_{\mathbb{R}^3} \tilde{V})(g), \quad i = 4, 5, \tag{26}$$

with $\mathcal{A}_i \nabla_{\mathbb{R}^3} = (\mathcal{A}_i \partial_x, \mathcal{A}_i \partial_y, \mathcal{A}_i \partial_z)^T$. We combine Eq. (26) with Eqs. (17) and (23), and direct calculations yield:

$$\begin{aligned}
\mathbf{R}_{\mathbf{n}}^{0\,T} \mathcal{A}_4 \nabla_{\mathbb{R}^3} \tilde{V}(g) &= \mathbf{R}_{\mathbf{n}}^{0\,T} \sum_{j=1}^{J} \mathbf{c}_j^{\nabla_{\mathbb{R}^3}}(\mathbf{x}) \tfrac{-1}{\sin\beta} \tfrac{\partial}{\partial\gamma} Y_j(\beta, \gamma), \\
\mathbf{R}_{\mathbf{n}}^{0\,T} \mathcal{A}_5 \nabla_{\mathbb{R}^3} \tilde{V}(g) &= \mathbf{R}_{\mathbf{n}}^{0\,T} \sum_{j=1}^{J} \mathbf{c}_j^{\nabla_{\mathbb{R}^3}}(\mathbf{x}) \tfrac{\partial}{\partial\beta} Y_j(\beta, \gamma).
\end{aligned} \tag{27}$$

Remark 1. In all implementations we use real valued spherical harmonics. The formulas for the derivatives are easily adapted for this using $\mathcal{A}_i\mathcal{A}_j \operatorname{Re}(Y_m^l) = \operatorname{Re}(\mathcal{A}_i\mathcal{A}_j Y_m^l)$.

4 Algorithm

We use the two-step method from [6] relying on a separate spatial and angular optimization using two submatrices of the symmetrized Hessian product. In the first step the spatial tangent is found and in the second step the curvature is estimated. In the second step we evaluate the Hessian at a different group element. Therefore we first apply some preparation steps such as approximating the signal using spherical harmonics, performing angular regularization and preparing spatial derivatives (Algorithm 1). Then we have the following two steps:

1) Determine spatial tangent First compute $\mathbf{H}\tilde{V}(g)$ using Algorithm 2, then compute rows and columns 1 to 3 of the symmetric Hessian product Eq.(10). Set $\mathbf{c}^{(1)}(g)$ equal to the eigenvector with smallest eigenvalue of this 3×3 matrix.

2) Determine curvature We determine the curvature at a different group element $g_{new} = (\mathbf{x}, \mathbf{R}_{\mathbf{n}_{new}})$, where $\mathbf{n}_{new} = \mathbf{R}_{\mathbf{n}}\mathbf{c}^{(1)}$, is the orientation given by the spatial tangent $\mathbf{c}^{(1)}(g)$ from step 1. Then compute $\mathbf{H}\tilde{V}(g_{new})$ using Algorithm 2, and compute rows and columns 3 to 5 of the symmetric Hessian product Eq.(10). Find the auxiliary coefficients $\mathbf{c}_{new}(g_{new}) = (0, 0, c^3(g_{new}), c^4(g_{new}), c^5(g_{new}), 0)^T$ where the components of the tangent vector $(\mu c^3, c^4, c^5)$ are set equal to the eigenvector with smallest eigenvalue of the 3×3 matrix.

We use the curvature determined at g_{new} for our curve fit at g (relying on neighboring exponential curves [6]). The coefficients of our fitted curve at g are given by

$$\mathbf{c}^*(g) = \begin{pmatrix} \mathbf{R}_{\mathbf{n}}^T\mathbf{R}_{\mathbf{n}_{new}} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{\mathbf{n}}^T\mathbf{R}_{\mathbf{n}_{new}} \end{pmatrix} \mathbf{c}_{new}(g_{new}). \tag{28}$$

This gives the final, torsion-free, exponential curve fit $t \mapsto \gamma_g^{\mathbf{c}^*(g)}(t)$ in $SE(3)$. Implementations of the left-invariant derivatives are available in the *Mathematica* package *Lie Analysis*: Available at http://lieanalysis.nl/orientationscores.html.

5 Experiments

For the experiments 3 artificial datasets were constructed. The first is a spiral with increasing curvature which contains a wide range of curvature values. The second and third contain random tube patterns. First, data on positions and orientations was constructed from our 3D data via the orientation score transform using cake-wavelets [9] with $N_o = 42$. Then the curvature was estimated using the exponential curve fit algorithm from Section 4 with

input : $\mathbf{u}$: $N_x \times N_y \times N_z \times N_o$-array containing the input data.
L : maximum order of spherical harmonics; initialize $J = j(L, L)$.
s_o, s_p: scale of angular and spatial regularization.
output: $\mathbf{c}^{\nabla_{\mathbb{R}^3}}$: $N_x \times N_y \times N_z \times J \times 3$-array containing the spherical harmonic coefficients of the gradient.
$\mathbf{c}^{\mathbf{H}_{\mathbb{R}^3}}$: $N_x \times N_y \times N_z \times J \times 3 \times 3$-array containing the spherical harmonic coefficients of the spatial Hessian.
$\mathbf{c}^{\tilde{V}}$: $N_x \times N_y \times N_z \times J$-array containing the spherical harmonic coefficients of $\tilde{V}$.
Approximation by spherical harmonics;
$\mathbf{M}$:=Eq. (14); $\mathbf{M}^+ := \left(\mathbf{M}^\dagger \mathbf{M}\right)^{-1} \mathbf{M}^\dagger$;
for *all positions* $\mathbf{x}$ **do**
  $\mathbf{c}(\mathbf{x}, \cdot) = \mathbf{M}^+ \mathbf{U}(\mathbf{x}, \cdot)$;
end
Angular Regularization (Eq. (15));
for *all positions* $\mathbf{x}$ *and all* j **do**
  $\mathbf{c}(\mathbf{x}, j) = \mathbf{c}(\mathbf{x}, j) e^{-l_j(l_j+1)s_o}$;
end
Spatial Derivatives and Spatial Regularization;
for *all* j **do**
  $\mathbf{c}^{\nabla_{\mathbb{R}^3}}(\cdot, j)$:= Eq. (17) (using Mathematica's GaussianFilter);
  $\mathbf{c}^{\mathbf{H}_{\mathbb{R}^3}}(\cdot, j)$:= Eq. (18) (using Mathematica's GaussianFilter);
  $\mathbf{c}^{\tilde{V}}(\cdot, j)$:= Eq. (19) (using Mathematica's GaussianFilter);
end

Algorithm 1: Preparation

input : $\mathbf{c}^{\nabla_{\mathbb{R}^3}}, \mathbf{c}^{\mathbf{H}_{\mathbb{R}^3}}, \mathbf{c}^{\tilde{V}}$: see output Algorithm 1.
$\mathbf{n}$: $N_x \times N_y \times N_z \times N_o$-array containing the input orientations.
output: $\mathbf{H}$: $N_x \times N_y \times N_z \times N_o \times 6 \times 6$-array containing the Hessian.
Calculate Hessian;
for *all positions* $\mathbf{x}$ *and all orientations* i **do**
  Initialize Rotation;
  $\mathbf{R}(\mathbf{x}, i) := \mathbf{R}^0_{\mathbf{n}(\mathbf{x},i)}$;
  Spatial Derivatives;
  $\mathbf{H}_{\mathbb{R}^3}$:=Eq. (18) with input $\mathbf{c}^{\mathbf{H}_{\mathbb{R}^3}}(\mathbf{x}, \cdot)$ and $\mathbf{n}(\mathbf{x}, i)$;
  $\mathbf{H}_{spat} := \mathbf{R}^T(\mathbf{x}, i) \mathbf{H}_{\mathbb{R}^3} \mathbf{R}(\mathbf{x}, i)$ (Eq. (21));
  $(\mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3)$:=Eq. (22) with input $\mathbf{c}^{\nabla_{\mathbb{R}^3}}(\mathbf{x}, \cdot)$ and $\mathbf{R}(\mathbf{x}, i)$;
  Angular Derivatives;
  $(\mathcal{A}_4, \mathcal{A}_5, \mathcal{A}_4\mathcal{A}_4, \mathcal{A}_4\mathcal{A}_5, \mathcal{A}_5\mathcal{A}_5)$:=Eqs. (23),(24) with input $\mathbf{c}^{\tilde{V}}(\mathbf{x}, \cdot)$ and $\mathbf{n}(\mathbf{x}, i)$;
  Mixed Derivatives;
  $\begin{pmatrix} \mathcal{A}_1\mathcal{A}_4 & \mathcal{A}_1\mathcal{A}_5 \\ \mathcal{A}_2\mathcal{A}_4 & \mathcal{A}_2\mathcal{A}_5 \\ \mathcal{A}_3\mathcal{A}_4 & \mathcal{A}_3\mathcal{A}_5 \end{pmatrix}$:=Eq. (27) with input $\mathbf{c}^{\nabla_{\mathbb{R}^3}}(\mathbf{x}, \cdot)$ and $\mathbf{n}(\mathbf{x}, i)$;
  Combine;
  $(\mathbf{H})(\mathbf{x}, i)$:= Eq. (20);
end

Algorithm 2: Hessian

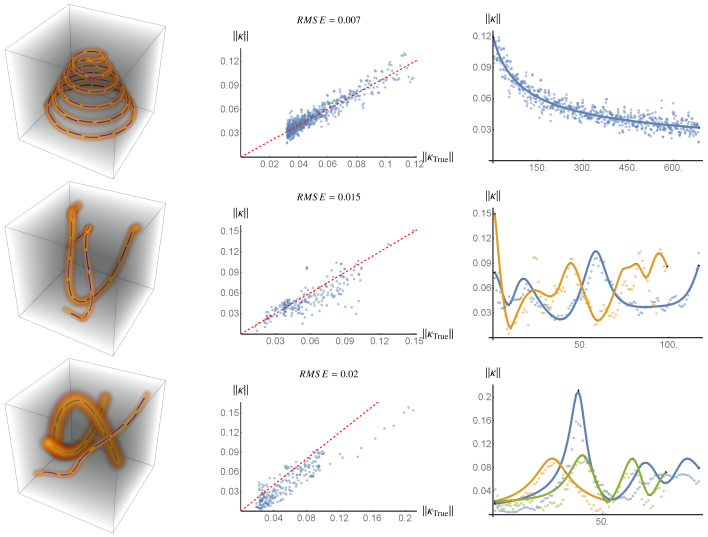


Fig. 2. Fit results on artificial datasets for $\sigma_{\text{Noise}} = 0.1$. Left: The data before adding noise. On a subset of points the spatial part of the fitted curves are shown (blue lines). Middle: Scatter plot of ground truth vs estimated curvatures. Right: Ground truth curvature at the selected points (lines) and estimated curvature (points) for the different structures represented by the different colors. RMSE: root-mean-square error.

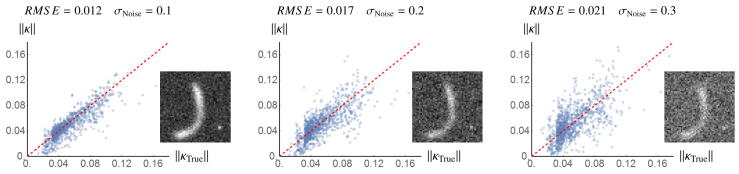


Fig. 3. Scatter plot of ground truth vs estimated curvatures for different amounts of noise. In these plots, the data from all four datasets is combined. Also a 2D slice through dataset 3 is shown for the different noise levels.

$(s_p, s_o) = (2, 0.08), \mu = 0.2$ and $L = 5$, followed by Eq. (8). We compared estimated curvatures to the ground truth curvatures, see Fig. 2. A high correlation is observed between ground truth and estimated curvature. In the left column we also show in each figure the spatial projection of the fitted exponential curves.

To test the robustness to noise of our method, we added different levels of Gaussian noise to our data, see Fig. 3, with σ_{Noise} ranging from 0.1 to 0.3. The data has a Gaussian tube profile which has a maximum value of one.

Finally, we show some preliminary results of curvature estimation on magnetic resonance angiography data which can be valuable in 3D vessel biomarkers [3]. Visual inspection shows that the measured curvature is high at locations where the vessels are highly curved, see Fig. 4.

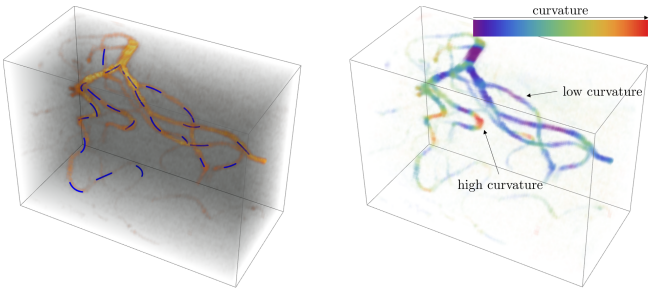


Fig. 4. Voxel-wise exponential curve fits for curvature estimation in 3D MR-anigography data. Left: The input patch (taken from subject "Normal-002" of the public database [3]). On a selection of points the spatial part of the fitted curves are shown (blue lines). Right: A plot of the corresponding curvature values at each voxel in color.

6 Conclusion

We presented a method for calculating a Gaussian regularized Hessian of axially symmetric functions on the roto-translation group $SE(3)$ obtained from data on positions and orientations. After expressing our data in spherical harmonics, all computations are exact. Furthermore, we used this Hessian in curvature estimation of tubular structures in 3D data after first applying an orientation score transform to the 3D data. Experimental results on artificial data show high accuracy of our curvature estimation and promising results on real medical data.

Acknowledgements. The research leading to these results has received funding from the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013) / ERC grant *Lie Analysis*, agr. nr. 335555. Tom Dela Haije gratefully acknowledges the Netherlands Organization for Scientific Research (NWO, No. 617.001.202) for financial support.

References

1. Bekkers, E.J., Zhang, J., Duits, R., ter Haar Romeny, B.M.: Curvature Based Biomarkers for Diabetic Retinopathy via Exponential Curve Fits in SE(2). Proc. Ophthalmic Med. Image Anal. Int. Workshop, 113–120 (2015)
2. Boscain, U., Chertovskih, R.A., Gauthier, J.P., Remizov, A.O.: Hypoelliptic Diffusion and Human Vision: A Semidiscrete New Twist. SIAM J. Imag. Sci. 7(2), 669–695 (2014)
3. Bullitt, E., Zeng, D., Gerig, G., Aylward, S., Joshi, S., Smith, J.K., Lin, W., Ewend, M.G.: Vessel Tortuosity and Brain Tumor Malignancy: A Blinded Study. Acad Radiol 12(10), 1232–1240 (2005)
4. Chirikjian, G.S., Kyatkin, A.B.: Engineering Applications of Noncommutative Harmonic Analysis: With Emphasis on Rotation and Motion Groups. (2000)
5. Creusen, E.J., Duits, R., Dela Haije, T.C.J.: Numerical Schemes for Linear and Non-linear Enhancement of DW-MRI. In: SSVM, 14–25. LNCS, No. 6667 (2011)
6. Duits, R., Janssen, M.H.J., Hannink, J., Sanguinetti, G.R.: Locally Adaptive Frames in the Roto-Translation Group and Their Applications in Medical Imaging. JMIV 56(3), 367–402 (2016)

7. Eshagh, M.: Alternative expressions for gravity gradients in local north-oriented frame and tensor spherical harmonics. Acta Geophys. 58(2), 215–243 (2009)
8. Frangi, A.F., Niessen, W.J., Vincken, K.L., Viergever, M.A.: Multiscale vessel enhancement filtering. MICCAI, 130–137. LNCS No. 1496 , (1998)
9. Janssen, M.H.J., Duits, R., Breeuwer, M.: Invertible Orientation Scores of 3D Images. In: SSVM, 563–575, LNCS No. 9087. (2015)
10. Savadjiev, P., Campbell, J.S.W., Pike, G.B., Siddiqi, K.: 3D curve inference for diffusion MRI regularization and fibre tractography. MIA 10(5), 799–813 (2006)
11. Savadjiev, P., Strijkers, G.J., Bakermans, A.J., Piuze, E., Zucker, S.W., Siddiqi, K.: Heart wall myofibers are arranged in minimal surfaces to optimize organ function. PNAS 109(24), 9248–9253 (2012)
12. Zweck, J., Williams, L.R.: Euclidean Group Invariant Computation of Stochastic Completion Fields Using Shiftable-Twistable Functions. JMIV 21(2), 135–154 (2004)