Shape from sensors: Curve networks on surfaces from 3D orientations

Tibor Stanko, Stefanie Hahmann, Georges-Pierre Bonneau, Nathalie Saguin-Sprynski

To cite this version:


HAL Id: hal-01524740
https://inria.hal.science/hal-01524740v2

Submitted on 19 May 2017
Shape from sensors: curve networks on surfaces from 3D orientations

Tibor Stanko$^{a,b}$, Stefanie Hahmann$^b$, Georges-Pierre Bonneau$^b$, Nathalie Saguin-Sprynski$^a$

$^a$CEA, LETI, MINATEC Campus, Université Grenoble Alpes
$^b$Université Grenoble Alpes, CNRS (Laboratoire Jean Kuntzmann), INRIA

Abstract

We present a novel framework for acquisition and reconstruction of 3D curves using orientations provided by inertial sensors. While the idea of sensor shape reconstruction is not new, we present the first method for creating well-connected networks of curves on surfaces using only orientation and distance measurements and a set of user-defined constraints. By working directly with orientations, our method robustly resolves problems arising from data inconsistency and sensor noise. Although originally designed for reconstruction of physical shapes, the framework can be used for “sketching” new shapes directly in 3D space. We test the performance of the method using two types of acquisition devices: a standard smartphone, and a custom-made device.

Keywords: curve networks, shape acquisition, surface reconstruction, inertial sensors, 3D sketching

1. Introduction

Curve networks play an important role in CAD and often serve for conveying designs [Coons, 1967; Moreton & Séquin, 1991]. At the same time, well-defined curve networks are used for automatic inference of underlying surfaces by minimizing fairing energy or using methods with prescribed degree of continuity.

Recent research in sketch-based modeling and virtual reality produced tools and devices for intuitive design of 3D curves (Fig. 3). These tools and devices attempt to overcome some of the limitations of the traditional CAD approach – designers can sketch curves directly, either on a flat screen or via a 3D interface. Although intuitive, these methods are targeted towards design of new shapes and are not suitable for reconstruction of existing real-world shapes.

We present a novel framework for acquisition of 3D curves using orientations measured by inertial sensors. While the idea of sensor shape reconstruction is not new, we present the first method for creating well-connected networks of curves on surfaces using only orientation and distance measurements and a small set of user-defined constraints. We address three main challenges that arise when working with sensor data.

- **Unknown positions.** Sensors measure local orientations of the surface – no absolute positions in the world space nor relative positions of two adjacent sensors are known.
- **Inconsistent data.** Intersecting curves often provide conflicting data, for instance two different normals for the same point in the world space.
- **Sensor noise.** Raw data from inertial sensors is noisy and needs to be pre-processed prior to reconstruction.

Our approach differs from standard curve acquisition and reconstruction methods in the fact that we formulate all algorithms in terms of orientations. This way, we can leverage consistency of reconstructed curves with respect to user-defined topological constraints. By working directly with orientations, our framework robustly resolves all above challenges in two steps. We first filter the acquisition noise by combining pre-filtering of the raw orienta-
tions in the quaternion space with a spline-based smoothing in the group of rotations (Sec. 4). We then introduce a curve-based Poisson reconstruction method, which transforms the pre-filtered orientation samples into a smooth and consistent curve network by satisfying in particular the user annotated topological constraints (Sec. 5). To enable a thorough evaluation and comparison with respect to ground truth data (Sec. 6), we use two physical objects fabricated from digital models: the lilium (Fig. 1) and the cone (Fig. 15).

To demonstrate our framework, we use a dynamic acquisition setup, where we suppose the data is measured by a single moving node of sensors. We test and compare two types of devices: a standard smartphone, and a custom-made prototype for measuring orientations and distances (Fig. 10). Nevertheless, the algorithms presented in this paper are not limited to these devices; they are also applicable in a static acquisition setup, such as a grid/mesh of sensors [Saguin-Sprynski et al., 2014; Sprynski, 2007] or instrumented materials [Hermanis et al., 2016].

The primary application of the presented framework is reconstruction of existing shapes. Our setup provides a valuable alternative to traditional shape acquisition methods (3D scanners, depth images). Unlike most 3D scanners, which often use cameras and lasers, inertial sensors are independent from light conditions, material’s optical properties and scale of the scanned object. They render the acquisition possible outdoors or underground, and can also be used for large and/or moving objects; imagine acquisition of underground pipes [Saguin-Sprynski et al., 2016], or an instrumented sail moving in the wind (Fig. 20). Possible applications include smart materials and structural health monitoring.

A major advantage of our approach is its generality. No special device is required for data acquisition; in fact, any sensor-instrumented device could be used for the task, thus making 3D acquisition accessible to everyone with an ordinary smartphone. Even though originally designed for reconstruction, our framework can also be used for design by “sketching” new shapes from scratch (Fig. 2).

Figure 2: The mushroom network created from scratch entirely with a smartphone. Curve lengths were estimated from acquisition time. The surfacing was computed using the method of Stanko et al. [2016] with soft positional constraints.

Figure 3: Recent curve acquisition devices that use inertial sensors. Left to right: Tilt Brush [Google], SmartPen [Milosevic et al., 2016] and 01 [InstruMMents, 2017].

2. Related work

Shape from curves. The problem of generating shapes from collections of curves has been well studied in CAGD (Computer Aided Geometric Design) and all standard techniques can be found in Farin [2002]. In the recent years, the interest in this problem was re-ignited due to its applications in sketch-based modeling [Andrews et al., 2011; Bae et al., 2008; Igarashi et al., 1999; Nealen et al., 2007; Xu et al., 2014] and in virtual reality (VR) systems [Google]. Inspired by years of research in CAGD, the surface from curves paradigm is invaluable for shape design in modern sketching and VR systems. Recently, Arora et al. [2017] have studied issues with accurate design of shapes in the existing VR systems. Their study suggests that users struggle even with simple tasks (drawing a closed circle) when sketching in three dimensions. A system like ours can help in solving such accuracy and consistency issues.

Although equally important, the problem of reconstruction of existing physical shapes from a collection of curves received less attention. 3D scanners usually provide large amounts of data and tend to ignore intrinsic structure of the scanned shape. Alternatively, objects can be defined by their characteristic curves as often done in perception and sketch-based modeling. Cao et al. [2016] detect characteristic curves in noisy point clouds, then use these curves for surface reconstruction. The use of inertial measurement units (IMUs) for shape acquisition might provide a good alternative in situations for which the optical methods do not yield proper results, positioning sensors along object’s characteristic curves. Milosevic et al. [2016] introduced SmartPen, a low-cost system for capturing 3D curves, which combines an IMU with a stereo camera (Fig. 3 middle). By combining a stereo camera with a sensor unit, their system is a mixture of traditional 3D scanners and our shape from sensors setup. However, SmartPen’s sensors only serve for determining device’s orientation needed for estimating relative position of the tip of the pen. Much like traditional point-cloud scanners, the system relies on visual input to get 3D position of the device in world space; this limits size of the scanned object. A recent example of curve acquisition device is the commercially available 01 [InstruMMents, 2017], a dimensioning tool with an IMU and a laser (Fig. 3 right). Allegedly, the device can be used to reconstruct 3D curves; this claim rests to be verified experimentally.
Shape from sensors. The use of sensors for shape acquisition was first explored by Sprynski [2007]. Instead of measuring absolute position of surface points of the scanned object in the world space, the reconstruction algorithms need to be formulated in terms of orientations provided by sensors, and geodesic distances between points of measurement. Curves are represented using natural parametrization and reconstructed via numerical integration. Surfaces are defined via geodesic interpolation [Sprynski et al., 2008] or using parallel ribbons of sensors [Saguin-Sprynski et al., 2014]. Huard et al. [2013] introduced method for computing smooth patches from a given piecewise geodesic boundary curve. Hoshi & Shinoda [2008] reconstruct the target surface using two families of sensors placed in orthogonal directions. Hermanis et al. [2016] construct sensor-instrumented fabric and compare the reconstruction results with Kinect data. Antonya et al. [2016] use an array of sensors for real-time tracking of human spine.

Attitude estimation and filtering. Attitude control has been studied extensively in aeronautics where accurate algorithms are indispensable for correct estimation of vehicle’s orientation with respect to celestial objects [Markley & Mortari, 2000]. Noise in data is usually reduced using a Kalman filter – specific approaches depend on the representation used for orientations [Crasidis et al., 2007], such as unit quaternions, a representation well-known in the graphics community [Shoemake, 1985]. Markley et al. [2007] describe a classical algorithm for computing means in the group SO(3). Average rotation is defined as the minimizer of weighted penalty function, and the corresponding unit quaternion is computed efficiently via eigendecomposition of a 4-by-4 matrix; see Sec. 4.1 for more details.

Apart from statistical approaches such as Kalman filters, geometric methods can be used to denoise orientations. Shoemake [1985] introduced quaternion splines using spherical Bézier curves; Nielson [2004] later extended this algorithm and introduced \( \nu \)-quaternion splines. Although spline methods work well for interpolation of rotations in a sequence of keyframes, the Bézier representation is not suitable for filtering noisy orientations. Energy-minimizing splines for data in Euclidean spaces have been well-studied. Reinsch [1967] introduced smoothing splines that minimize stretching and bending while approximating given data in a Euclidean space. Hofer & Pottmann [2004] describe a method for computing such splines for data on a Riemannian manifold \( \mathcal{M} \). Data are first optimized in the ambient (Euclidean) space, then projected onto \( \mathcal{M} \); these two steps are repeated until convergence. An equivalent result can be found by optimizing data directly on the manifold [Boumal & Absil, 2011]. In this sense, classical splines are obtained for \( \mathcal{M} = \mathbb{R}^d \). We use smoothing splines on the manifold SO(3) to filter raw orientation data from sensors (Sec. 4.3).

Poisson reconstruction. Poisson’s equation \( \Delta \phi = \nabla \cdot \mathbf{v} \) often arises in the context of reconstruction of shapes (curves and surfaces). Kazhdan et al. [2006] resolve a Poisson problem \( \Delta \chi = \nabla \cdot \mathbf{N} \) to find the indicator function \( \phi \), which implicitly defines the surface, from a sample of the surface’s oriented normal field \( \mathbf{N} \). This approach is a state of the art method for surface reconstruction from unorganized points clouds. Our reconstruction algorithm is inspired by the work of Crane et al. [2013] who use the Poisson equation \( \Delta f = \nabla \cdot \mathbf{T} \) to retrieve positions \( f \) of a closed planar curve from its tangent field \( \mathbf{T} \) during iso-metric curvature flow. We extend this approach to closed curve networks on surfaces, and use it to reconstruct positions of the scanned curves (Sec. 5).

Figure 5: In this example, only the top left cell complex is a valid input to our algorithm. The other three networks have non-contractible cycles (top right), violate the definition of a cell complex (bottom left) or they are disconnected (bottom right).
3. Definitions and input data

Given a smooth connected 2-manifold surface \( S \subset \mathbb{E}^3 \), we consider a collection \( \Gamma \) of \( G^1 \) smooth curves embedded on \( S \) (\( \Gamma \) is therefore a topological subspace of \( S \)). Curves in \( \Gamma \) divide the surface into a finite number of cells forming a two-dimensional cell complex \( C = (V, E, \Gamma) \) on \( S \), consisting of nodes \( V \) (0-cells), segments \( E \) (1-cells) and cycles \( \Gamma \) (2-cells) [Berberich et al., 2010; Hatcher, 2002]. Moreover we suppose the cell complex is connected and the cycles are contractible (Fig. 5). Each curve \( \gamma \in \Gamma \) can be thought of as a path in \( C \). We say that two curves intersect at a node \( \eta \in \Gamma \) if their corresponding paths contain \( \eta \).

Denote by \( x : [0, L] \to \mathbb{E}^3 \) the natural parametrization of \( \gamma \in \Gamma \) where \( L \) is the length of \( \gamma \). For a fixed point \( x \) on the curve, the orthonormal Darboux frame \( D = (T, B, N) \) is defined as

\[
T = x', \quad N \perp T_N, \quad B = N \times T.
\]

Here, \( T_N \) denotes the two-dimensional tangent space of \( S \) at the point \( x \in S \) and \( N \) is chosen as the outward surface normal. We represent the Darboux frame \( D \) as an orientation matrix \( A = [T \ B \ N] \in \text{SO}(3) \), which is a member of the special orthogonal group

\[
\text{SO}(3) = \left\{ A \in \mathbb{R}^{3\times3} : A^T A = I \text{ and } \det(A) = 1 \right\}.
\]

We denote by \( A_{|T}, A_{|B}, A_{|N} \) the projections on the first, second and third column of the matrix \( A \), respectively. Throughout this paper, the terms “orientation” and “frame” refer to the same concept – the Darboux frame of a curve point with respect to the underlying surface.

The function \( A : \Gamma \to \text{SO}(3) \), which associates each curve point to its Darboux frame, is not continuous. Indeed, at intersections the tangent vector \( T \) differs for each adjacent curve (Fig. 6). Note however that the projection on the normal component \( A_{|N} \) is continuous and is identical to the Gauss map of \( S \) restricted to the curves in \( \Gamma \).

Our goal is to retrieve the unknown positions \( x \) provided a sample of orientations \( A_i = A(x(d_i)) \) for each curve \( \gamma \) at known distances \( d_i \in [0, L] \). We suppose the topology of the underlying cell complex \( C \) is known. The main steps of our algorithm are shown in Fig. 4.

4. Filtering orientations

In this section, we describe our approach for filtering raw orientation data acquired using inertial sensors. Sensor noise and limited scanning precision are the main problems that prevent using raw orientations in the reconstruction of positions. By consequence, the acquired network of orientations is often inconsistent: at intersections, the normals are different for each passing curve (Fig. 16). To eliminate acquisition problems, we use a two-step filtering process – see Fig. 7. First, raw data are pre-filtered using distance-weighted Gaussian convolution with fixed edge length. This way, we obtain uniform sampling of orientations with respect to curve lengths. Second, the uniform samples are smoothed and made consistent using regression on the manifold \( \text{SO}(3) \).

![Figure 7: Two-step filtering of acquired orientations from Fig. 1, here visualized using the Gauss map \( A_{|N} \) (surface normals extracted from orientations mapped to the unit sphere). Left to right: raw data, pre-filtered data (Sec. 4.2), filtered data (Sec. 4.3).](image)

4.1. Averaging orientations

To compute (weighted) means in the group of rotations, it is convenient to use the quaternion representation [Markley & Mortari, 2000]. The rotation around the (unit) Euler axis \( e \) by the angle \( 2\theta \) is given by the unit quaternion \( q = [\cos \theta; e \sin \theta] \in S^3 \subset \mathbb{H} \), where \( S^3 \) denotes the unit 3-sphere in the quaternion space \( \mathbb{H} \approx \mathbb{R}^4 \). We will denote by \( A(q) \) the conversion between the orientation matrix \( A \) and the unit quaternion \( q \) (see Shoemake [1985] for details).

Given a set of unit quaternions \( q_i \in S^3 \) with weights \( w_i \) for \( i = 1, \ldots, N \), the average quaternion \( \bar{q} \) is defined via the following minimization problem [Markley et al., 2007]:

\[
\bar{q} = \arg \min_{q \in S^3} \sum_{i=1}^{N} w_i \| A(q) - A(q_i) \|_F^2,
\]

where \( \| \cdot \|_F \) is the Frobenius norm. This is equivalent to solving

\[
\bar{q} = \arg \max_{q \in S^3} q^T M q,
\]

where the 4x4 matrix \( M \) is the structure tensor

\[
M = \sum_{i=1}^{N} w_i q_i q_i^T.
\]
The solution to this optimization problem is known to be the (unit) eigenvector \( \mathbf{q} \) of \( \mathbf{M} \) corresponding to the largest eigenvalue. This procedure is known as the q method. The q method is computationally efficient and naturally overcomes the fact that unit quaternions \( S^3 \) are a 2-to-1 covering of the rotation group: antipodal quaternions \( \mathbf{q} \) and \( -\mathbf{q} \) represent the same rotation. This is why spherical linear interpolation [Slerp, Shoemake, 1985] is not suitable for averaging quaternions – changing the sign of a quaternion should not change the average, which is not the case for Slerp.

### 4.2. Pre-filtering via Gaussian convolution

Working with orientations which are sampled uniformly with respect to geodesic distance facilitates both the filtering (Sec. 4.3) and the reconstruction of positions (Sec. 5) – the uniform sampling is convenient for discretization of differential operators and integrals. In this section we describe the computation of such uniform sampling from raw orientation measures in the quaternion space.

Before computing the uniform sampling, we first remove outliers and process duplicate measures. Outliers are removed by looking at the dot products of quaternion \( \mathbf{q} \) with its immediate neighbors \( \mathbf{q}_{i-1} \) and \( \mathbf{q}_{i+1} \). If both dot products are smaller than the specified threshold \( \alpha \), the angles between \( \mathbf{q} \) and its neighbors are too big and \( \mathbf{q} \) is removed. For the examples in this paper, we use the threshold \( \alpha = 0.9 \); this is consistent with the assumption that the acquisition frequency is high enough and each two adjacent data points are close in the space of orientations. For sparsely sampled networks, a smaller threshold or an adaptive scheme might be more suitable. Duplicate measures, i.e. different orientations which correspond to the same curve point, are averaged using the q method described in the previous section (with \( w_i \equiv 1 \)).

To compute uniform sampling from unique sampling (no outliers or duplicates), we fix the edge length \( h_{\text{fixed}} \) and compute uniform distance parameters \( t_k \) for each curve \( \gamma \):

\[
t_k = \frac{kL}{N}, \quad k = 0, \ldots, N, \quad t_k - t_{k-1} = h
\]

where \( L \) is the length of \( \gamma \) and \( N = \text{round}(L/h_{\text{fixed}}) \). For each \( t_k \), we compute the corresponding orientation \( \mathbf{q}_k \) using convolution with distance weighting (Fig. 8):

\[
\mathbf{q}_k = \text{mean}( \{ w_{kj} \mathbf{q}_j \} ).
\]

Here, mean( ) refers to the q method described in Sec. 4.1. The Gaussian weights are computed as

\[
w_{kj} = \exp \left( -\frac{1}{2} \left( \frac{t_k - d_j}{\sigma h_{\text{fixed}}} \right)^2 \right).
\]

The parameter \( \sigma \) controls the radius of convolution; in general we use values between 0.2 and 0.5.

### 4.3. Regression on SO(3) with normal constraints

To some extent, the pre-filtering smooths out acquisition artifacts in the scanned data. By varying the parameter \( \sigma \), it might be possible to sufficiently filter the data while preserving the acquired geometry. However, this approach has a major drawback: since the Gaussian convolution is applied to each curve individually, there is no guarantee that the filtered orientations will be consistent at intersections (see Sec. 3). This type of coherence is however essential, not only for the final reconstruction step of our method (orientations transformed into positional surface data) which assumes consistent input, but also to ensure a correct subsequent processing of the curve network, for instance for surface fitting.

Our goal is therefore to compute a function which is as smooth as possible while fitting the set of frames and satisfying a set of consistency constraints. To have a filtering framework with unified representation, our initial idea was to use quaternion splines [Nielsen, 2004; Park & Ravani, 1997; Shoemake, 1985]. However, expressing the consistency constraints directly in the quaternion space gives nonlinear equations; on the other hand, these constraints are linear in SO(3). Inspired by spline functions, which provide smooth curves that reasonably fit the input data by minimizing some energy functional, we propose in the following to formulate filtering of frames (orientations) as a customized energy minimization problem on SO(3).

Recall that a spline in tension \( x(t) \) in the \( d \)-dimensional Euclidean space \( \mathbb{R}^d \) is the minimizer of the energy

\[
\lambda \int_{t_0}^{t_N} ||\dot{x}||^2 dt + \mu \int_{t_0}^{t_N} ||\ddot{x}||^2 dt + \int_{t_0}^{t_N} f(t) dt
\]

under the interpolation constraints \( x(t_i) = p_i \in \mathbb{R}^d \). The weights \( \lambda \) and \( \mu \) control stretching and bending of the spline, respectively. Incorporating positional constraints directly in the energy

\[
\sum_{i=0}^{N} \|x(t_i) - p_i\|^2 + \lambda \int_{t_0}^{t_N} ||\dot{x}||^2 dt + \mu \int_{t_0}^{t_N} ||\ddot{x}||^2 dt
\]

yields a smoothing spline, useful for non-parametric regression and data filtering in Euclidean spaces.

Smoothing splines have been generalized for data on Riemannian manifolds by many authors [Boumal & Absil, 2011; Brunnett & Crouch, 1994; Hofer & Pottmann, 2004].
To discretize this problem, the continuous curve integral operators in (2) are replaced by the tensor $\text{SO}(3)$, where $\gamma$ is the Riemannian (geodesic) distance on $\text{SO}(3)$, $\langle \cdot , \cdot \rangle$ is the Riemannian metric, $\mathbf{X}$ is the first derivative and $D^2 \mathbf{X} / dt^2$ is the second covariant derivative of $\mathbf{X}$. We will write the above cost function as

$$E(\mathbf{X}) = E_0(\mathbf{X}) + \lambda E_1(\mathbf{X}) + \mu E_2(\mathbf{X}).$$

To discretize this problem, the continuous curve $\mathbf{X}(i) : [0, L] \rightarrow \text{SO}(3)$ is replaced by the tensor

$$\gamma = (\mathbf{X}(t_0), \ldots, \mathbf{X}(t_N)) = (\mathbf{X}_0, \ldots, \mathbf{X}_N) \in \text{SO}(3)^{N+1}$$

where $t_k$ is the uniform parametrization defined in (1).

We use the method for spline fitting on manifolds of Boumal & Absil [2011]. For the sake of completeness, we briefly review the main ideas here. The differential and integral operators in (2) are approximated via geometric finite differences. Geometrically, the difference $\mathbf{B} - \mathbf{A}$ of two points $\mathbf{A}, \mathbf{B} \in \mathbb{R}^n$ in the Euclidean space is the vector starting at $\mathbf{A}$ and pointing towards $\mathbf{B}$ whose length is the distance between $\mathbf{A}$ and $\mathbf{B}$. On manifolds, the same notion is expressed using the Riemannian logarithmic map as $\log (\mathbf{A}^\top \mathbf{B})$. Note that the logarithmic map is the inverse of the exponential map, which generates geodesics. For orientation data $\mathbf{A}_i \in \text{SO}(3)$, the cost function (2) has the following form:

$$\sum_{i=0}^{N} \| \log (\mathbf{X}_i^\top \mathbf{A}_i) \|_F^2 + \frac{1}{h} \sum_{i=0}^{N-1} \| \log (\mathbf{X}_i^\top \mathbf{X}_{i+1}) \|_F^2$$

$$+ \frac{1}{h^2} \sum_{i=1}^{N-1} \| \log (\mathbf{X}_i^\top \mathbf{X}_{i+1}) + \log (\mathbf{X}_i^\top \mathbf{X}_{i-1}) \|_F^2.$$  

We use the following simplified costs [Boumal, 2013]:

$$E_0(\gamma) = \sum_{i=0}^{N} \| \mathbf{X}_i - \mathbf{A}_i \|_F^2, \quad E_1(\gamma) = \frac{1}{h} \sum_{i=0}^{N-1} \| \mathbf{X}_i - \mathbf{X}_{i+1} \|_F^2,$$

$$E_2(\gamma) = \frac{1}{h^2} \sum_{i=1}^{N-1} \| \text{skew} (\mathbf{X}_i^\top (\mathbf{X}_{i+1} + \mathbf{X}_{i-1})) \|_F^2,  \quad (3)$$

where $\gamma$ is the vector of unknown orientations (discrete curve), $h = t_i - t_{i-1}$ is the uniform discretization step, and $\text{skew}(\mathbf{M}) = (\mathbf{M} - \mathbf{M}^\top) / 2$. For closed curves, all sums run from 0 to $N$ due to periodicity.

Let us recall how the simplified energy is obtained. The terms $E_0, E_1$ result from the approximation of the geodesic distance on $\text{SO}(3)$ by the chordal distance in the ambient space $\mathbb{R}^{3 \times 3}$:

$$\text{dist}^2 (\mathbf{A}, \mathbf{B}) = \| \log (\mathbf{A}^\top \mathbf{B}) \|_F^2 \approx \| \mathbf{A} - \mathbf{B} \|_F^2.$$  

To obtain the simplified term $E_2$, consider the map $\mathbf{B} \mapsto \mathbf{A} \log (\mathbf{A}^\top \mathbf{B})$, which associates to $\mathbf{B} \in \text{SO}(3)$ a tangent vector at $\mathbf{A} \in \text{SO}(3)$. A similar vector is obtained by projecting the vector $\mathbf{B} - \mathbf{A}$ from the ambient space of $\mathbb{R}^{3 \times 3}$ to the tangent space $T_\mathbf{A} \text{SO}(3)$. This projection is characterized by the operator $\mathbf{A} \text{skew}(\mathbf{A}^\top \mathbf{B})$. It is therefore convenient to approximate the log operator by the skew operator:

$$\log (\mathbf{A}) \approx \text{skew} (\mathbf{A}).$$  

See the paper of Boumal [2013] for an analysis and the error bounds of the simplified cost.

In our setup, the raw data $\mathbf{A}_i$ is associated to a network $\Gamma$ and the above energy terms are summed over all curves $\gamma \in \Gamma$. Simultaneously with regression on $\text{SO}(3)$, we solve the consistency constraints as follows. Let $\mathbf{X}_i, \mathbf{X}_j$ be the local frames of two intersecting curves at their common node $\eta \in \mathcal{V}$. Then the two frames are called consistent, if the frame $\mathbf{X}_i = [\mathbf{T}_i, \mathbf{B}_i, \mathbf{N}_i]$ is obtained by rotating the frame $\mathbf{X}_j = [\mathbf{T}_j, \mathbf{B}_j, \mathbf{N}_j]$ around the surface normal $\mathbf{N}$ at the node $\mathcal{N}$. Denote by $N$ the set of all such pairs $(\mathbf{X}_i, \mathbf{X}_j)$. To enforce consistency, we add a term penalizing the difference in projection on the normal component

$$E_N (\Gamma) = \sum_{(\mathbf{X}_i, \mathbf{X}_j) \in N} \| \mathbf{X}_i^\top \mathbf{N} - \mathbf{X}_j^\top \mathbf{N} \|_2^2.  \quad (4)$$

The final energy has the form

$$E (\Gamma) = \xi E_N (\Gamma) + \sum_{\gamma \in \Gamma} E_0 (\gamma) + \lambda E_1 (\gamma) + \mu E_2 (\gamma),  \quad (5)$$

where $\xi, \lambda, \mu \geq 0$ are the weights controlling consistency of surface normals, stretching, and bending, respectively (Fig. 16).

The energy $E$ is minimized using the Riemannian trust-region (RTR), a generalization of the classical trust-region in $\mathbb{R}^n$ to differentiable manifolds; see Absil et al. [2007] for the description of the algorithm and a detailed convergence analysis. We use the RTR implemented in the ROPTLIB library [Huang et al., 2016].

5. Poisson network reconstruction

In the previous section, we described a filtering method of raw orientation data measured by sensors, which yields uniform, smooth and consistent sampling. We now address the problem of curve network reconstruction under topological constraints: these include intersections of two or more curves, and closure of a single curve.
Previous approaches treat each curve individually and employ various heuristics to glue the curves together to obtain the correct topology [Hermanis et al., 2016; Sprynski, 2007]. This process is limited to specific (grid) topologies and often requires manual corrections to obtain proper intersections. Our approach, on the other hand, naturally satisfies topological constraints by construction. In this section, we extend the discretization of the Poisson equation for a single closed curve described by Crane et al. [2013] to curve networks with the topology of a cell complex as defined in Sec. 3.

If x is the natural parametrization of the curve γ (one-dimensional Riemannian manifold), the unit tangent field T along γ is equal to the gradient of positions

$$\nabla x = T$$

The vector field T is generally not integrable [Crane et al., 2013; Kazhdan et al., 2006] and the solution needs to be found in the least-squares sense. To that end, the divergence operator is applied to both sides and x is found by solving the Poisson equation

$$\Delta x = \nabla \cdot T$$

(6)

Even though the curve network \(\Gamma\) is not a Riemannian manifold, it can be viewed as a collection of one-dimensional Riemannian manifolds constrained at intersections. This allows us to apply the above Poisson equation to retrieve the network from local orientations. With that in mind, we discretize the gradient \(\nabla\) and the Laplacian \(\Delta\) individually for each curve and retrieve the positions by solving a single global linear system. The topological constraints are enforced by representing each node as a unique point in the system (Fig. 9).

After the filtering, each curve is represented as a discrete collection of unknown points \(x_i = x(t_i) \in \mathbb{R}^3, i = 0, \ldots, N\), with known Darboux frames \(X_i\). We will denote by \(T_i = X_{i+1}\) the (unit) tangent extracted from \(X_i\). With uniform parametrization \(h = t_i - t_{i-1}\), the differential operators are discretized via finite differences by

$$\Delta x_i = 1/h^2 (x_{i-1} - 2x_i + x_{i+1}), \quad \nabla T_i = 1/h (T_i - T_{i-1}).$$

The above discretization is valid for all interior vertices. At endpoints of open curves, we directly impose the following boundary conditions

$$1/h (x_1 - x_0) = T_0, \quad 1/h (x_N - x_{N-1}) = T_N.$$ 

The resulting linear system is sparse with at most three non-zero coefficients per row (Fig. 9). Thanks to the regression step described in Sec. 4.3, the tangent vectors of all curves meeting at network node are coplanar at that node. Therefore the above boundary conditions ensure the reconstruction of a \(G^1\) curve network. Note that the discretization does not take into account the fact that \(h = t_i - t_{i-1}\) is the geodesic distance along the reconstructed curve, which differs from the (Euclidean) norm \(r_i = \|x_{i-1} - x_i\|\). In general, this is not an issue, as the least-squares minimization of the Poisson energy distributes the error evenly.

The euclidean distance \(r_i\) could be estimated from the geodesic distance \(h\), for instance by approximating \(r_i\) by the chordal length of the circular arc defined by the length \(h\) and angle between \(T_{i-1}\) and \(T_i\) (see inset). This estimation comes at a price – the modified parametrization is no longer uniform. Our experiments have shown that the error \(|r_i - h|\) is generally too small to influence the reconstruction.

6. Evaluation

6.1. Data acquisition

To acquire data on physical surfaces, we use dynamic acquisition setup – the orientations are measured using a single IMU, which is moved around on the surface by the user. In our experiments we used two types of devices: a standard smartphone, and a custom-made device. The Morphorider (Fig. 10) is a wireless mouse-like device for measuring local orientation, containing a single 3A3M sensor (a tri-axial accelerometer and a tri-axial magnetometer) with an incremental wheel for tracking distance. It is a prototype specifically designed for acquisition of curves on surfaces.

![Figure 10: Left: the Morphorider and the wireless keypad used for marking nodes during acquisition. Right: the schema shows limitations of the device with respect to radius of curvature at locally convex (red) and concave (blue) surface points.](image)

We use a MATLAB interface to acquire data with the Morphorider (Fig. 11 left). Prior to acquisition, we assign a unique index between 0 and \(n - 1\) to each of the \(n\) network nodes. The acquisition is controlled remotely using...
a wireless numpad (Fig. 10 left). When starting a new curve, we first indicate if the curve is at the boundary or in the interior. Then, every time the device passes through a node, we record its index. See the accompanying video for the acquisition setup.

![Image](image1.png)

Figure 11: The devices used for data acquisition. Left: the Morphorider (a custom-made device) and a screenshot of the MATLAB acquisition interface. Right: a standard smartphone and the application with a simple acquisition interface.

Similarly, orientation data can be measured with a standard smartphone. In addition to accelerometers and magnetometers, smartphones also contain gyroscopes. This enables orientation acquisition that is more robust to magnetic perturbations. We use the state of the art sensor fusion algorithms to robustly estimate orientation from sensor measurements [Madgwick et al., 2011; Pacha, 2013]. However, measuring distances with a smartphone is difficult. The GPS is not an option for mid-sized objects such as the lilium (Fig. 11) whose bounding box has dimensions 1.00m \( \times \) 1.00m \( \times \) 0.25m. To overcome this limitation, we assume the speed of acquisition is constant, then we parametrize each curve using acquisition timestamps. Optionally, manual distance measurements can be used to increase the precision of reconstruction. We measure the length \( L_s \) for each segment \( s \) which is then used to transform the time parameter \( t \in [t_{\text{start}}, t_{\text{end}}] \) into the natural parameter \( d \in [0, L_s] \). In this case, we suppose the speed of acquisition is constant for each segment, but it does not need to be constant through the whole acquisition process. In practice, the constant-speed-per-segment constraint is much easier to handle and the manual distance measurement improves the reconstruction (Fig. 12). To acquire data with a smartphone, we use a lightweight Java application (Fig. 11 right).

6.2. Convergence and error of reconstruction

We demonstrate the performance of our reconstruction framework on synthetic and acquired data.

- We use synthetic data extracted from known ground truth networks for convergence analysis of the Poisson network reconstruction (Fig. 13). The synthetic data are not corrupted by noise and therefore do not need to be filtered.
- Data acquired from known surfaces serve for evaluating the overall performance of the framework on real-world examples (Fig. 14 and 15). Moreover, they enable quantitative comparison of the two acquisition methods.

![Image](image2.png)

Figure 12: Comparison of three acquisition setups for the lilium. Top pictures show surfaced networks rendered with reflection lines; bottom pictures show the error of Poisson reconstruction for each network. In the left network, the orientations were acquired using the Morphorider and parametrized by distances. In middle and right networks, the orientations were acquired with the smartphone and parametrized by acquisition time – middle network uses measured segment lengths, right network uses time-estimated segment lengths. All lengths are relative to the diameter (length of AABB diagonal) of the ground truth surface.

![Image](image3.png)

Figure 13: As \( h \to 0 \), synthetic networks reconstructed by solving the Poisson equation (6) converge to the ground truth network (black). All lengths are relative to the diameter (length of AABB diagonal) of the ground truth surface (orange).

The error metric associated with each point \( x \) in the reconstructed network is defined as the distance between \( x \) and its closest point on the ground truth network or surface. For synthetic data, the closest point is found on the ground truth network. For data acquired from a physical surface with a known digital model (lilium, cone), closest points are instead found on the underlying surface as the ground truth network is not available. Note that the dimensions of the fabricated surfaces are 1.00m \( \times \) 1.00m \( \times \) 0.25m (lilium) and 1.00m \( \times \) 1.00m \( \times \) 2.00m (cone). Prior to error computation the acquired network needs to be registered to the surface, since the two objects do not live in the same coordinate systems. We use the ICP algorithm [Besl & McKay, 1992] for the registration.

The convergence plot for decreasing \( h \) is shown in Fig. 17 and the corresponding numerical data is given in Table 1. All errors are relative to the diameter (the length of AABB diagonal) of the corresponding ground truth surface. For synthetic data (bowl, bumpycube), the
mean error approaches zero for decreasing $h$ (Fig. 13). For acquired networks on the Lilium, the mean error is about 1% using measured distances (both devices) and around 2% using time-estimated distances (smartphone). It is not surprising that when using the measured distances, the smartphone error decreases faster than the Morphorider error. One can further observe that even though the smartphone error (both mean and max) is bigger than the Morphorider error at coarser resolutions, it becomes smaller at finer scales. We explain this behavior by the fact that the distance data are not the same for the two devices. For the Morphorider data, we have exact correspondences between distance and orientation measurements; for the smartphone data, only the total length of each segment is known and the correspondences are estimated from acquisition time. It seems like at finer scales, the error of this estimation is better-distributed, which results in a reconstruction with higher precision.

6.3. Filtering

The main goal of the filtering algorithm described in Sec. 4 is to obtain clean and consistent orientations from raw sensor measures. The weights $\lambda, \mu$ in the filtering energy (5) influence the shape of the reconstructed network indirectly by controlling stretching and bending of splines on the manifold SO(3) used for orientation smoothing. In all examples, we use the weight $\xi = 1e4$ to enforce the consistency of normals.

The influence of $\lambda$ and $\mu$ on the Poisson-reconstructed positions is compared in Fig. 18. Interestingly, even though $\lambda$ and $\mu$ control stretching and bending of orientations $X_i$, they also seem to control stretching and bending of reconstructed positions $x_i$. The weights need to be cho-
7. Limitations and future work

Most limitations of our method are device-related. The current Morphorider is a proof of concept and suffers from construction drawbacks. Its relatively big size inhibits the acquisition; we often found it hard to manipulate, and some regions with high curvature (in absolute value) could not have been scanned (see Fig. 10). Another problem is that the distance-measuring wheel is not aligned with the sensor unit – in practice this means that there is an offset in the orientation measurement. Moreover, the device relies on magnetometers and cannot be used around ferromagnetic objects. A next generation prototype currently in development might help in resolving some of these problems. The biggest limitation of the smartphone setup is the inability to measure distances – these need to be estimated or obtained manually.

The current approach requires the user to have a specific network topology in mind prior to the acquisition. Nodes cannot be added during acquisition – adding a new node means that all previously scanned curves passing through this node need to be re-scanned, which is impractical and slows down the scanning process. Moreover, the system needs user feedback to know when the acquisition device passes through a node. In future, we plan to implement a more intuitive interface by using a heuristic for snapping curves to existing nodes to ease the acquisition.

Curvature plots in Fig. 18 imply that the reconstructed curves are $G^2$, but there are issues with overall $G^2$-continuity at nodes. We believe these issues might be resolved by a slight modification of the normal penalty term in the filtering energy, for instance by taking into account normals of node’s neighboring vertices. We also plan to extend the method to surfaces with sharp features, i.e., curves with two sets of frames that agree up to rotation around curve’s tangent vector.

Acknowledgments. This research was partially funded by the ERC advanced grant no. 291184 EXPRESSIVE.
Figure 18: Reconstruction of \( \lambda = \mu = 1 \) with various filtering weights. Left to right in each row: reconstructed network with porcupine plot (surface normal scaled by curvature); Gauss map; curvature plot.

Figure 19: Surface of a roof box reconstructed with different sets of curves.

<table>
<thead>
<tr>
<th>data</th>
<th>samples</th>
<th>time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>raw</td>
<td>uni</td>
</tr>
<tr>
<td>MUSHROOM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fig. 2</td>
<td>831</td>
<td>508</td>
</tr>
<tr>
<td>LILIJUM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fig. 14 top</td>
<td>96</td>
<td>196</td>
</tr>
<tr>
<td></td>
<td>3243</td>
<td>382</td>
</tr>
<tr>
<td></td>
<td>742</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>1416</td>
<td></td>
</tr>
<tr>
<td>LILIJUM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fig. 14 bottom</td>
<td>99</td>
<td>186</td>
</tr>
<tr>
<td></td>
<td>531</td>
<td>362</td>
</tr>
<tr>
<td></td>
<td>713</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1420</td>
<td></td>
</tr>
<tr>
<td>CONE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fig. 15</td>
<td>949</td>
<td>738</td>
</tr>
<tr>
<td></td>
<td>1553</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3094</td>
<td></td>
</tr>
<tr>
<td>CHAIR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fig. 16</td>
<td>584</td>
<td>247</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LILIJUM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fig. 18</td>
<td>3243</td>
<td>428</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Performance of the algorithm implemented in C++ using ROPTLIB on a standard quad-core CPU computer. The stats include number of raw samples, number of uniform samples, Gaussian convolution time, SO(3) regression time, Poisson network reconstruction time. All timings are in milliseconds.