

Chapter 1

Calculation of rotation coefficient

We are interested in the rotation. Let p, q be points on the manifold and ℓ a path connecting them. Let $R : \mathcal{M} \rightarrow SO(3)$ be a rotation field, i.e. $R(p), R(q)$ are orthogonal frames. Under the initial condition that $R(p) = \text{Id}$, the equation

$$R(q) = \exp(R_{pq}(\omega))R(p)$$

is a differential equation that corresponds to the parallel transport under the connection ω of the frame along ℓ . To recover this rotation R_{pq} , we integrate ω along ℓ .

Discretization We parametrize the path by $\ell(0) = a, \ell(1) = b$. We resort to numerical integration for R_{ab} and cut the path into n small segments, i.e.

$$R_{ab} = R_n R_{n-1} \cdots R_1$$

where $R_i = \exp(-\omega(\dot{\ell}(i\gamma))\gamma) = \exp((\int W^\top dp)_\times)$, and γ is the length of a segment and $\dot{\ell}(s) = \frac{\partial \ell}{\partial s}(s)$. Calculating the exponential map of an antisymmetric matrix (which $\omega \in \mathfrak{so}(3)$ is) can be done with Rodrigues' formula:

$$\exp(u_\times) = \text{Id} + \sin(\theta)\hat{u}_\times + (1 - \cos(\theta))\hat{u}_\times^2$$

where $\theta = \|u\|_2$ is the rotation angle and $\hat{u} = u/\theta$ is the rotation axis. We use the trapezoidal rule to evaluate the a short interval of the integral of W , which is given by

$$R_{ab} = \exp\left(\left(\frac{1}{2}(W_a + W_b)^\top(b - a)\right)_\times\right)$$

where W_a, W_b is solved for by the 9x9 linear system given by A_x and $\nabla \times A$.

Piecewise linear discretization We discretize our metric field with a tetrahedral mesh \mathcal{T} . At each vertex, we attach a metric and linearly interpolate with barycentric coordinates within a tet.

Let $A_i \in \mathbb{R}^{3 \times 3}, i \in \{1, 2, 3, 4\}$ be the square root metrics at the vertices $v_i \in \mathbb{R}^3$ of a tet, such that $A_i^2 = g(v_i)$. We represent a tet given by its four vertices by a 3x4 matrix, i.e.

$$\begin{pmatrix} | & | & | & | \\ v_1 & v_2 & v_3 & v_4 \\ | & | & | & | \end{pmatrix} \in \mathbb{R}^{3 \times 4}.$$

Any point p within the tet can then be represented as

$$p = \alpha v_1 + \beta v_2 + \gamma v_3 + \delta v_4$$

with $\alpha, \beta, \gamma, \delta \geq 0$ and $\alpha + \beta + \gamma + \delta = 1$. This is a linear transformation between two coordinate systems, which we can write in matrix form as

$$\underbrace{\begin{pmatrix} | & | & | & | \\ v_1 & v_2 & v_3 & v_4 \\ | & | & | & | \\ 1 & 1 & 1 & 1 \end{pmatrix}}_{T^{-1}} \underbrace{\begin{pmatrix} \alpha \\ \beta \\ \gamma \\ \delta \end{pmatrix}}_{\lambda} = \underbrace{\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}}_p \iff T^{-1}\lambda = p \iff \lambda = Tp$$

T always exists because v_1, \dots, v_4 are linearly independent, else it would not be a tetrahedron. By denoting $T = \{t_{ij}\}_{i,j \in \{1, \dots, 4\}}$, we can write our barycentric functions as

$$\begin{aligned} \alpha(x, y, z) &= t_{11}x + t_{12}y + t_{13}z + t_{14} \\ \beta(x, y, z) &= t_{21}x + t_{22}y + t_{23}z + t_{24} \\ \gamma(x, y, z) &= t_{31}x + t_{32}y + t_{33}z + t_{34} \\ \delta(x, y, z) &= t_{41}x + t_{42}y + t_{43}z + t_{44} \end{aligned}$$

The convex combination

$$A(x, y, z) = \alpha A_1 + \beta A_2 + \gamma A_3 + \delta A_4$$

is the metric prescribed in the tet. To find $\nabla \times A$, let $A = (A^1, A^2, A^3)$. We will need the derivatives for the curl, so let

$$(A_j^i)_x \triangleq \frac{\partial A_j^i}{\partial x}$$

be the derivative with respect to x of entry i, j . E.g. $(A_j^i)_x$ is given by

$$(A_j^i)_x = \alpha_x (A_1)_j^i + \beta_x (A_2)_j^i + \gamma_x (A_3)_j^i + \delta_x (A_4)_j^i = t_{11}(A_1)_j^i + t_{21}(A_2)_j^i + t_{31}(A_3)_j^i + t_{41}(A_4)_j^i$$

If we write $T = (T^1, T^2, T^3, T^4)$ and collect $(A_k)_j^i$ into a vector

$$\bar{A}_j^i = \begin{pmatrix} (A_1)_j^i \\ (A_2)_j^i \\ (A_3)_j^i \\ (A_4)_j^i \end{pmatrix}$$

this can be shortened to $\bar{A}_j^{i\top} T^1 = (A_j^i)_x$. Analogously, we get

$$(A_j^i)_y = \bar{A}_j^{i\top} T^2 \text{ and } (A_j^i)_z = \bar{A}_j^{i\top} T^3$$

The curl is then given by

$$\nabla \times A^i = \begin{pmatrix} (A_3^i)_y - (A_2^i)_z \\ (A_1^i)_z - (A_3^i)_x \\ (A_2^i)_x - (A_1^i)_y \end{pmatrix} = \begin{pmatrix} \bar{A}_3^{i\top} T^2 - \bar{A}_2^{i\top} T^3 \\ \bar{A}_1^{i\top} T^3 - \bar{A}_3^{i\top} T^1 \\ \bar{A}_2^{i\top} T^1 - \bar{A}_1^{i\top} T^2 \end{pmatrix}$$

and $\nabla \times A = \nabla \times (A^1, A^2, A^3)$. Notice that the curl is constant within a tetrahedron.

Recursive Subdivision Whenever the rotation coefficient R_{pq} is needed between some points p and q , it is unclear ahead of time how many sampling points on the line ℓ are needed such that R_{pq} accurately describes how the frame rotates along ℓ . We apply a recursive subdivision scheme to recursively sample more points on ℓ only where is needed until sampling more points leads to no noticable improvement

anymore (see figure 1.1). We begin by calculating R_{ab} with just the endpoints. This coefficient is then compared to the result if the midpoint was sampled aswell, so i.e. if

$$\frac{||R_{ab} - R_{am} \cdot R_{mb}||_2^2}{\ell^2} < \varepsilon$$

then no measurable improvement happened. We divide by the length of the segment \vec{ab} , because the segments get smaller and we want the tolerance ε to remain the same. This approach has the advantage of only sampling points where there is improvement. See section ?? for results.

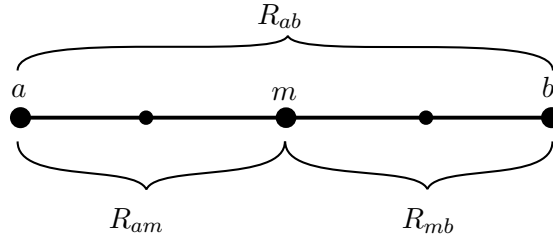


Figure 1.1. Points are recursively sampled as midpoints between the line \vec{ab} . If sampling more points within some line segment leads to noticable improvement, more points are sampled.

The following code does what is described above.

Algorithm 1 Recursive Subdivision

```

1: recursiveDivide ( $a, b$ )
2:    $R_1 = \text{Rotation}(a, b)$ 
3:    $\ell = \text{length}(a - b)$ 
4:    $\text{midpoint} = \frac{a+b}{2}$ 
5:   if  $\frac{||R_1 - \text{Rotation}(a, \text{midpoint}) \cdot \text{Rotation}(\text{midpoint}, b)||_2^2}{\ell^2} < \varepsilon$ 
6:     return  $R_1$ 
7:   else
8:     return  $\text{recursiveDivide}(a, \text{midpoint}) \cdot \text{recursiveDivide}(\text{midpoint}, b)$ 
```
