Mapping with Kalman Filter on Riemannian Manifolds

by Roshan Kumar

Introduction

- Importance of investigating vibrating systems in structural dynamics
- Necessity for sophisticated modeling, especially with Geometrically Consistent Tuned Mass Dampers (GTMDs)
- Research focus on precise parameter and state estimation

Key Objectives

- State and parameter estimation of systems on Riemannian Manifolds
- Optimization and parameter estimation of Pendulum Cart System on SO(3) Manifold
- Mathematical modeling of Geometrically Consistent Pendulum

Background

Vibrating Structures

- Traditional modeling often relies on Euclidean spaces which are the inherent complex and estimation still has significant error
- profound significance in earthquake engineering and structural design

Tuned Mass Dampers

Pendulum tuned mass dampers (PTMDs) have emerged as a prominent solution for mitigating structural vibrations induced primarily by dynamic forces such as wind.

- Time domain modal parametric identification of natural frequencies, mode shapes, and modal damping ratios of structures equipped with PTMDs can help us model Geometrically consistent Tuned Massed Damper

Data Generation

Used Geometric Ito-Taylor 1.5 method for simulating stochastic differential equations we provide the dynamic equation for a chaotic pendulum in the Special Orthogonal Group SO(3) manifold,

$$I \bullet \dot{\omega} = -\omega * I \bullet \omega + u \tag{1.1}$$

Where:

- I is the moment of inertia matrix.
- ω represents the angular velocity vector.
- $\dot{\omega}$ is the time derivative of the angular velocity.
- u represents the control input, which incorporates the effects of the tuned mass damper and any external forces or disturbances.

Manifolds

This research focuses on precise parameter and state estimation, employing Kalman filters on Riemannian manifolds, specifically in the context of Lie Algebra. The aim is to address gaps in literature on parameter and state estimation for vibrating structures with GTMDs.

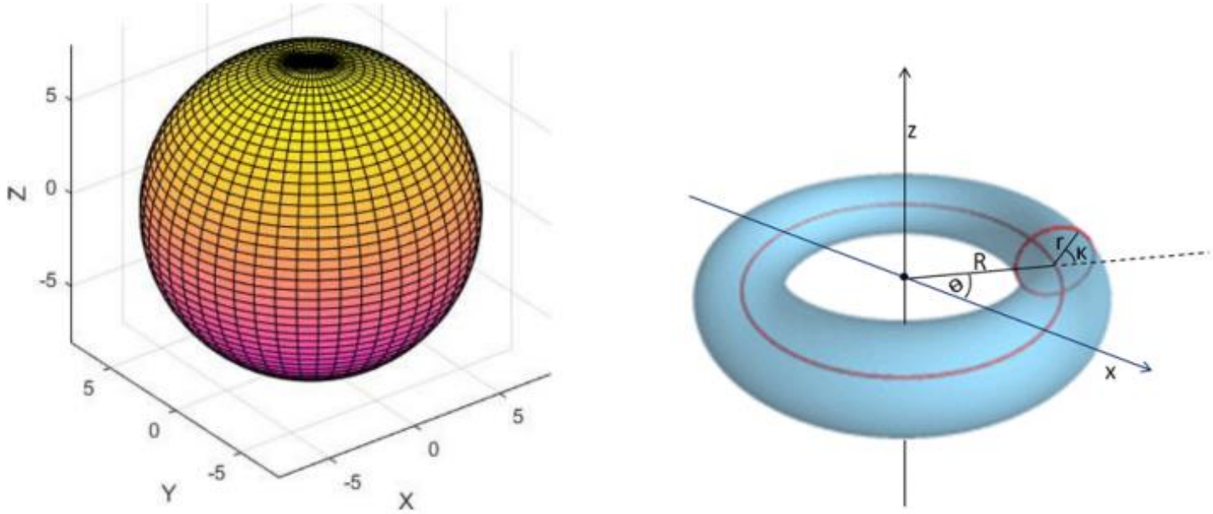


Fig. 2.1 Manifolds (a) Hypersphere (b) Torus

Building Foundation

A **tensor** is an object that is invariant under a change of coordinates and has components that change in a special predictable way under the change of coordinates.

- Forward Transformation (F):

$$T'_{ij} = \frac{\partial x'_i}{\partial x_k}\frac{\partial x'_j}{\partial x_l}T_{kl}$$

- Backward Transformation (B):

$$T_{kl} = \frac{\partial x_i}{\partial x'_k}\frac{\partial x_j}{\partial x'_l}T'_{ij}$$

such that $F \cdot B = I$.

Co-Vectors is A function that takes a vector and produces a scalar is called a co-vector. Spaces of co-vectors are called dual spaces.

Metric Tensor It is an additional structure on a manifold M in the field of differential geometry that allows for the definition of distances and angles, much like the inner product on an Euclidean space does

Geodesics

The straightest possible path we can draw on surfaces between two points is called a geodesic.

$$\frac{d^2u^k}{d\lambda^2} + \Gamma^k_{ij}\frac{du^i}{d\lambda}\cdot\frac{du^j}{d\lambda} = 0$$

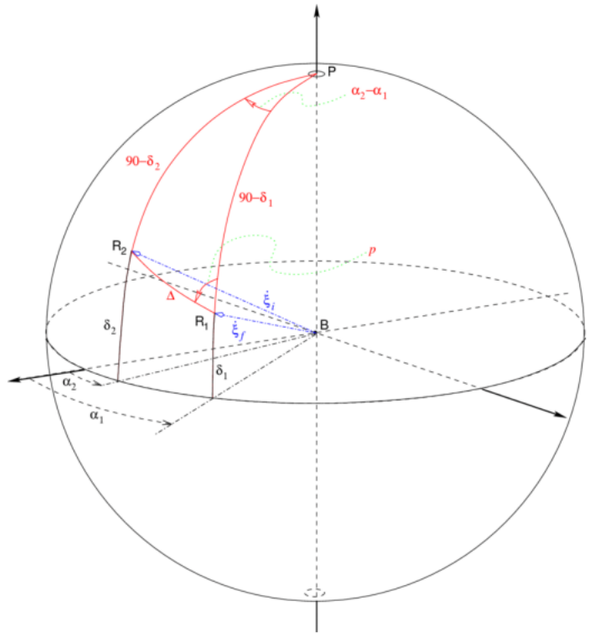


Fig. 2.2 Geodesic between two points on Sphere

Christoffel Symbol

The straightest possible path we can draw on surfaces between two points is called a geodesic.

$$\frac{d^2\vec{R}}{d\lambda^2} = \frac{d^2\vec{R}}{d\lambda^2}_{\text{tangential}} + \frac{d^2\vec{R}}{d\lambda^2}_{\text{normal}} \tag{2.5}$$

Now, Tangential Component = 0, on expanding the above equation we get,

$$\frac{d^2R_j}{d\lambda^2} = \left(\frac{d^2u^k}{d\lambda^2} + \Gamma^k_{ij}\frac{du^i}{d\lambda}\frac{du^j}{d\lambda}\right)\frac{\partial\vec{R}}{\partial u^k} + L_{ij}\frac{du^i}{d\lambda}\frac{du^j}{d\lambda}\hat{n} \tag{2.6}$$

$$\frac{d^2u^k}{d\lambda^2} + \Gamma^k_{ij}\frac{du^i}{d\lambda}\frac{du^j}{d\lambda} = 0 \quad \text{(Geodesic Equation)} \tag{2.7}$$

$$\Gamma^k_{ij} = \frac{\partial^2\vec{R}}{\partial u^j \partial u^j}\cdot\frac{\partial\vec{R}}{\partial u^l}g^{lk} \quad \text{(Christoffel Symbol)} \tag{2.8}$$

Here u^i and u^j are the basis vectors and R is the position vector. And g^{lk} is the component of inverse metric tensor.

Mapping

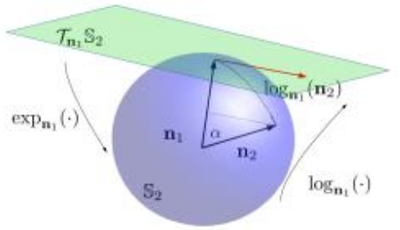


Fig. 2.4 Exponential Mapping and Logarithmic Mapping on surface of S^2 manifold

Lie Group and Lie Algebra

Lie group is a smooth manifold that also carries a group structure whose product and inversion operations are smooth as maps of manifolds

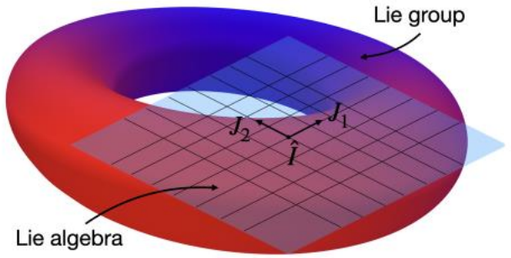


Fig. 2.5 Working of Lie Groups and Lie Algebra

2.3.1 Group Axioms

1. **Closure:** If $a, b \in G$, then $\phi(a, b) \in G$.
2. **Associative:** If $a, b, c \in G$, then $\phi(a, \phi(b, c)) = \phi(\phi(a, b), c)$.
3. **Identity:** If $a \in G$, then there exists $e \in G$ such that $\phi(a, e) = \phi(e, a) = a$.
4. **Inverse:** If $a \in G$, then there exists a unique element $a^{-1} \in G$ such that $\phi(a, a^{-1}) = \phi(a^{-1}, a) = e$.

$$SO(2) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

Lie Algebra:

$$\mathfrak{so}(2) = \left\{ \begin{bmatrix} 0 & -t \\ t & 0 \end{bmatrix}, \quad t \in \mathbb{R} \right.$$

Exponential Map:

$$\exp\left(\begin{bmatrix} 0 & -\theta \\ \theta & 0 \end{bmatrix} \right) = \gamma_\theta(1) = \begin{bmatrix} \cos t\theta & -\sin t\theta \\ \sin t\theta & \cos t\theta \end{bmatrix}$$

SO3 Group

The group SO(3) is made up of rotation matrices or special orthogonal matrices in 3D space that are subject to matrix multiplication. In all groups SO, inversion and composition are accomplished through transposition and product (n). The lie algebra of the group is defined by angular velocities ωx, ωy, ωz.

$$[\omega] = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$$

Exponential Map:

$$R = \exp([\omega]\theta) \in SO(3), \text{ where } R \text{ is the rotation matrix.}$$

$$R = I + [\omega]\sin(\theta) + [\omega]^2(1 - \cos(\theta))$$

Logarithm:

$$\theta[\omega] = \log(R)\theta(R - R^\top)\frac{1}{2\sin(\theta)}$$

Where,

$$\theta = \cos^{-1}\left(\frac{\text{trace}(R) - 1}{2}\right)$$

Kalman Filter

The Kalman filter is a recursive algorithm that estimates the state of a dynamic system from a series of noisy measurements

State Prediction

The prediction of the state is given by the system dynamics:

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_{k-1})$$

Error Covariance Prediction

The error covariance matrix is predicted using the Jacobian of the state transition function:

$$P_{k|k-1} = A_{k-1} P_{k-1|k-1} A_{k-1}^{\top} + Q_{k-1}$$

Measurement Prediction

The predicted measurement is obtained using the measurement model:

$$\hat{z}_{k|k-1} = h(\hat{x}_{k|k-1})$$

Kalman Gain Calculation

The Kalman gain is computed to determine the weight of the measurement in the state correction:

$$K_k = P_{k|k-1} H_k^{\top} (H_k P_{k|k-1} H_k^{\top} + R_k)^{-1}$$

where H_k is the Jacobian of the measurement function, and R_k is the measurement noise covariance.

Kalman Filter

The Kalman filter is a recursive algorithm that estimates the state of a dynamic system from a series of noisy measurements

State Prediction

The prediction of the state is given by the system dynamics:

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_{k-1})$$

Error Covariance Prediction

The error covariance matrix is predicted using the Jacobian of the state transition function:

$$P_{k|k-1} = A_{k-1} P_{k-1|k-1} A_{k-1}^{\top} + Q_{k-1}$$

Measurement Prediction

The predicted measurement is obtained using the measurement model:

$$\hat{z}_{k|k-1} = h(\hat{x}_{k|k-1})$$

Kalman Gain Calculation

The Kalman gain is computed to determine the weight of the measurement in the state correction:

$$K_k = P_{k|k-1} H_k^{\top} (H_k P_{k|k-1} H_k^{\top} + R_k)^{-1}$$

where H_k is the Jacobian of the measurement function, and R_k is the measurement noise covariance.

Simulation

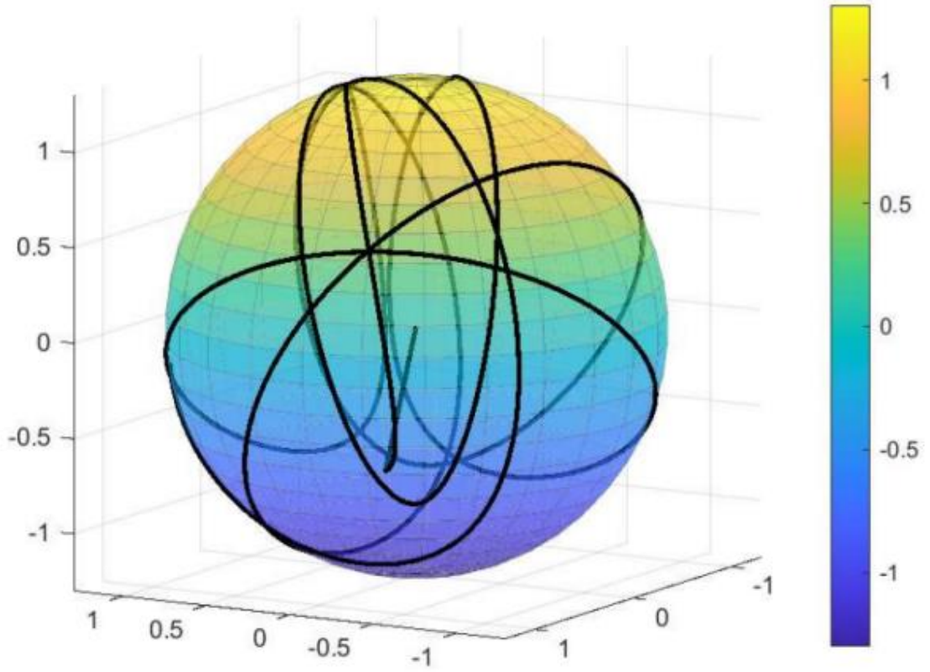
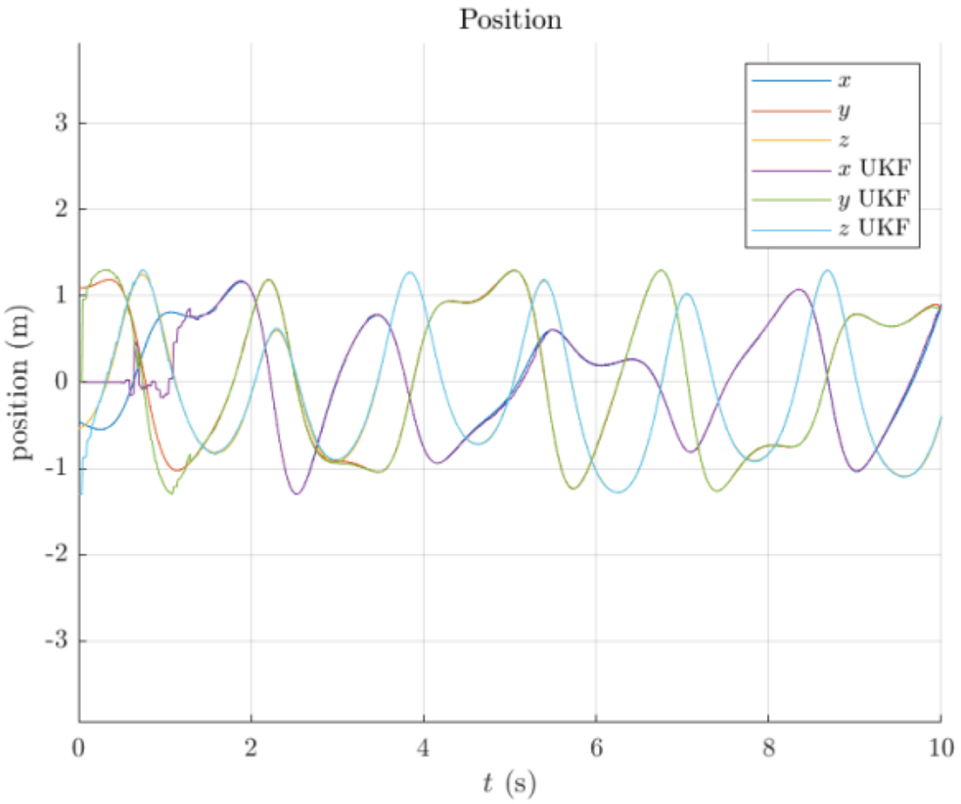


Fig. 3.1 Trajectory on S03 Manifold

Results



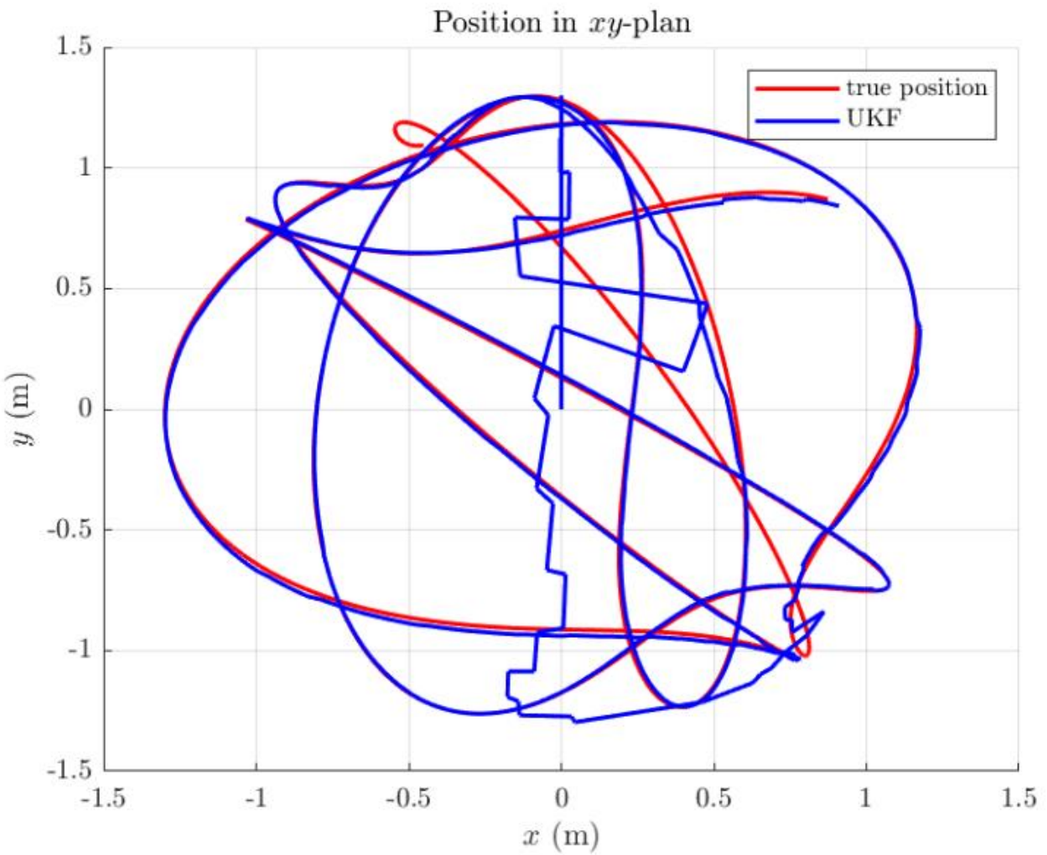
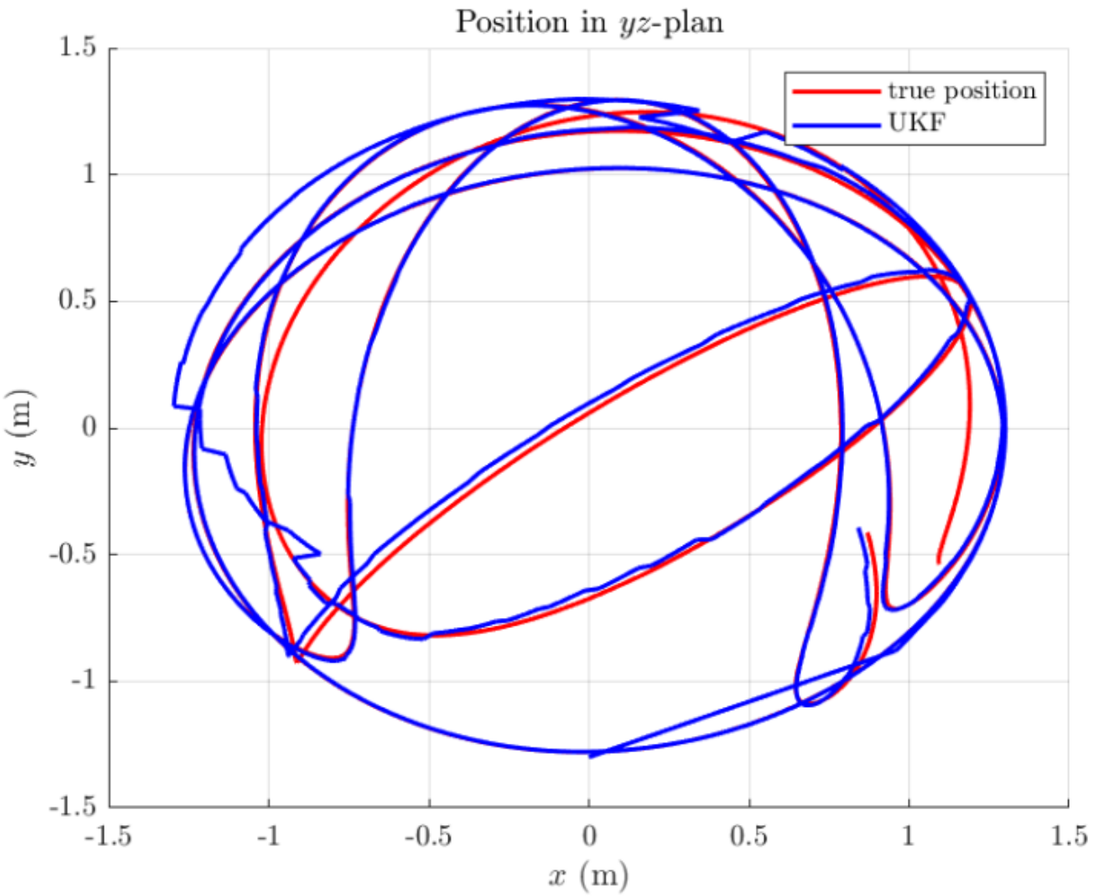
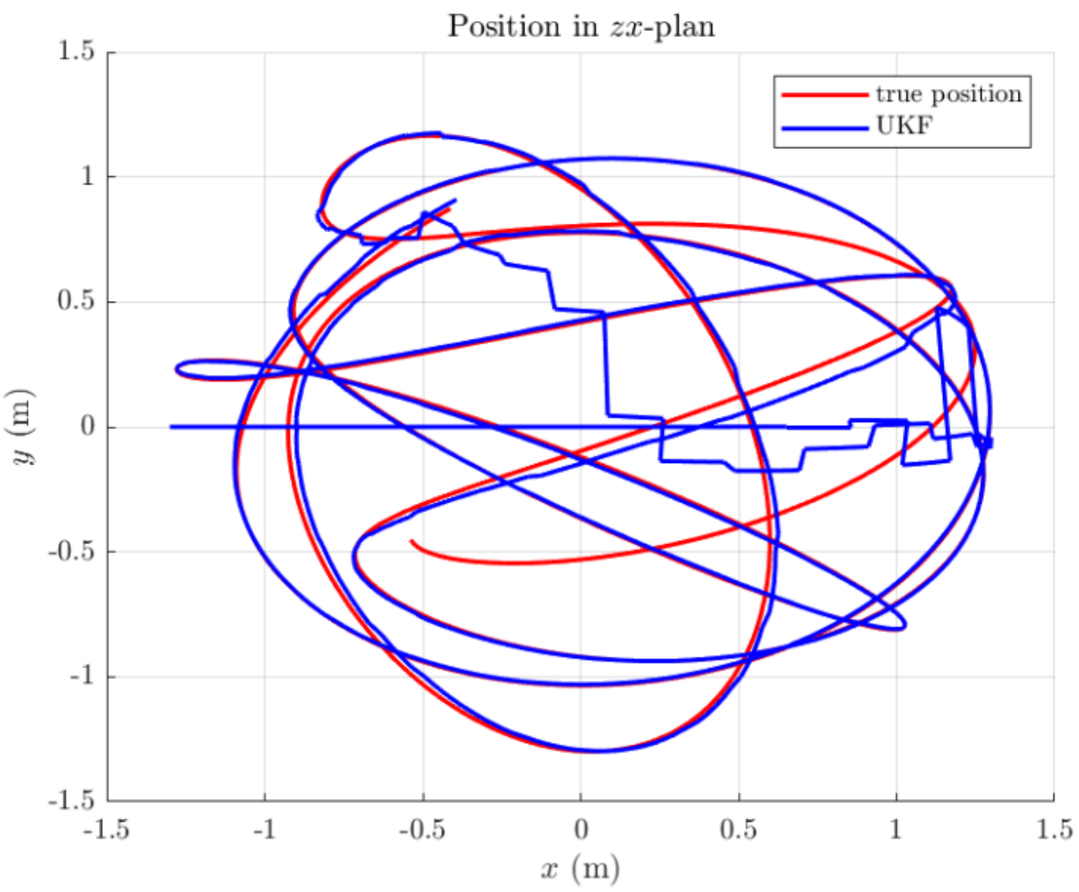


Fig. 3.2 Position(m)

Results


Position in yz-plan
true position
UKF
1.5
1
0.5
0
-0.5
-1
-1.5
y (m)
-1.5
-1
-0.5
0
0.5
1
1.5
x (m)


Position in zx-plan
true position
UKF
1.5
1
0.5
0
-0.5
-1
-1.5
y (m)
-1.5
-1
-0.5
0
0.5
1
1.5
x (m)

Results

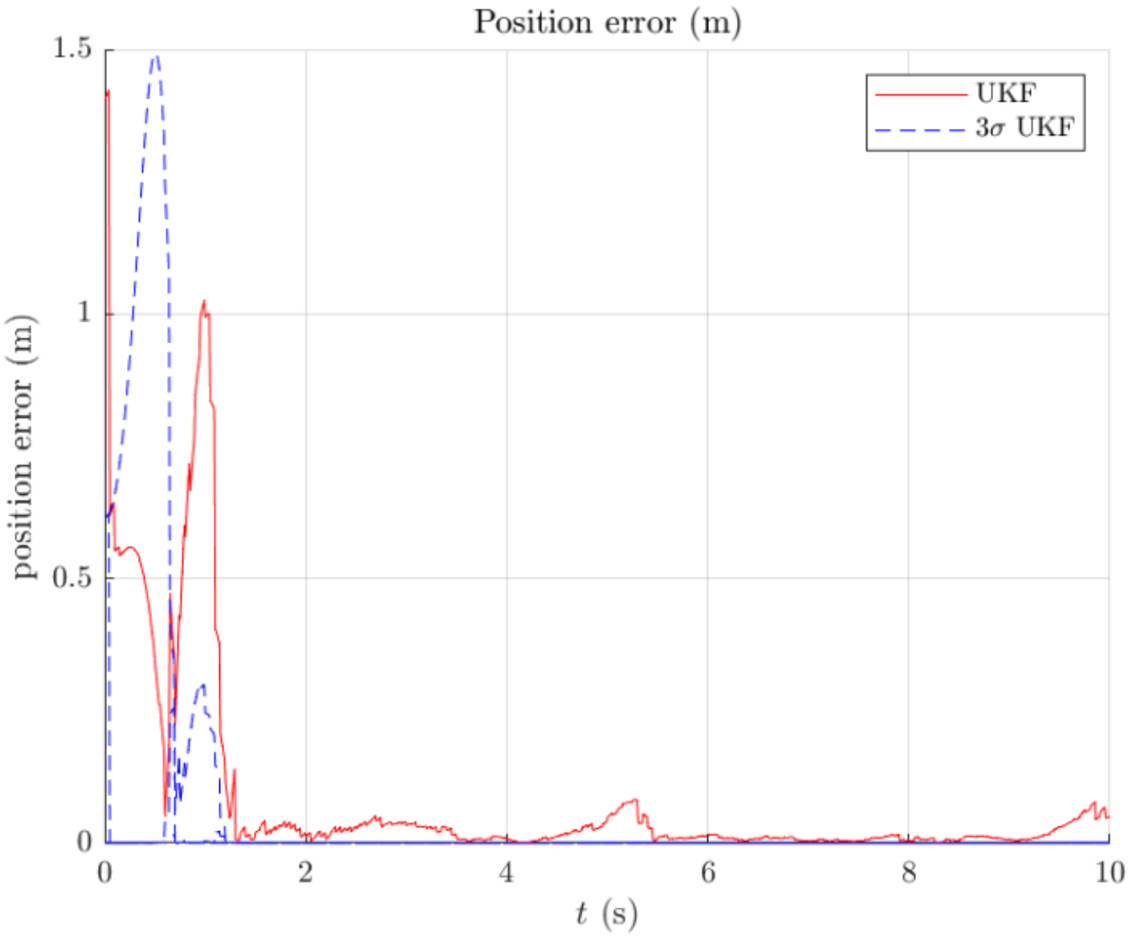


Fig. 3.6 Position Error