Cubature Kalman Filter based SLAM

Kumar Pakki. Bharani Chandra, Da-Wei Gu and Ian Postlethwaite

Abstract— **Simultaneous Localization and Mapping (SLAM) is the process of simultaneously building a map and locating a vehicle in it, and can be used for autonomous navigation. SLAM deals with the estimation of vehicle states and landmarks. Most SLAM algorithms are based on extended Kalman filters (EKFs). However, the use of EKF for SLAM is not the best choice, as it works only for 'mild' nonlinear environments owing to the assumption of first order Taylor series approximations of vehicle and observation models. In this paper, we propose the use of a cubature Kalman filter (CKF) for the estimation of augmented state vector in SLAM. CKF SLAM has the ability to deal with nonlinear models, without using any linearization assumptions. A comparison of CKF SLAM and EKF SLAM is given through numerical simulations.**

I. Introduction

Autonomous vehicles are required to determine their own states, since most of their actions (such as path planning, autonomous navigation etc.) depends on them. Localization is the process of estimating a vehicle's position and orientation based on landmarks or beacons. Localization based on an *a priori* map requires the knowledge of the environment defined by the location of different landmarks and hence the environment needs to be explored in advance. As a consequence, the autonomous vehicle is limited to operating within the known environment. If it is necessary to extend the environment of operation, new areas have to be surveyed before autonomous localization can take place. This situation can however be improved by SLAM [3].

The SLAM problem asks: Is it possible for a vehicle to be placed at an unknown location in an unknown environment and to build incrementally a consistent map of the environment while simultaneously determining its location within this map? SLAM has been an active research area for several years and its solution is seen as the "holy grail" by the robotics community [2]. Initial research into this area began with a landmark paper by Smith, Self and Cheeseman which introduced the concept of a stochastic map in which a mobile robot acquires knowledge about its location and organizes its environment by making sensor observations at different places and in different times [1].

The most common representation used in SLAM is a state-space model with additive Gaussian noise leading to the well known EKF [2]. Nonlinear functions are used to represent the process dynamics as well as the observation model, and states are estimated using a recursive process (time update and measurement update). During the state prediction and update stage, linearization is needed to compute an estimate of the new vehicle position in the map as well as the correlated position of the landmarks in the covariance matrix. However, the EKF approach for SLAM is only suitable for 'mild' nonlinearities where first-order approximations of the nonlinear functions are suitable. In order to address the problems caused by linearization, use of an unscented Kalman filter (UKF) appears to be an appealing option. The UKF has in general been shown to perform better than the EKF in nonlinear estimation problems. In the case of SLAM, where hundreds of landmarks are typically included in the state vector, this increases the computational burden and hence can preclude real-time operation. Use of UKF for a SLAM application is given in [5].

To overcome the limitations of currently known filters, a Cubature Kalman Filter (CKF) has been proposed in the recent literature [6]. CKF is a Gaussian approximation of Bayesian filter, but provides a more accurate filtering than existing Gaussian filters and it has the ability to deal with a wider range of nonlinear problems.

According to the authors' knowledge, SLAM based on CKF is not yet investigated in the robotics' community. In this paper, we propose the usage of CKF for nonlinear state estimation of SLAM. The augmented state vector of vehicle states and the location of landmarks are estimated using CKF.

The remainder of this paper is structured as follows. Section II includes the preliminaries of filtering, and Section III briefly describes the SLAM problem and its solutions. Section IV is devoted to numerical simulations of SLAM. Section V concludes with possible refinements of SLAM.

II. Filtering Preliminaries

This section presents a brief introduction to the EKF and CKF. For a detailed formulation and derivation of these methods see for example [7] for EKF and [6] for CKF.

A. The Extended Kalman Filter

The EKF is a set of mathematical equations that provides a computationally efficient (recursive) way to estimate the state of a nonlinear system. In this work, the EKF is described in discrete-time.

1) Process and Observation Models: The discrete-time process and observation models can be written as

$$\mathbf{x}_k = \mathbf{f}[\mathbf{x}_{k-1}, \mathbf{u}_{k-1}] + \mathbf{w}_{k-1} \quad (1)$$

$$\mathbf{z}_k = \mathbf{h}[\mathbf{x}_k, \mathbf{u}_k] + \mathbf{v}_k \quad (2)$$

This work is supported by UKIERI.

Kumar Pakki. Bharani Chandra is a PhD student with Control and Instrumentation Group, University of Leicester, UK, LE1 7RH, bcp3@leicester.ac.uk.

Da-Wei Gu is the Head of the Control and Instrumentation Group, University of Leicester, UK, LE1 7RH, dag@leicester.ac.uk.

Ian Postlethwaite is Deputy Vice-Chancellor at Northumbria University, UK, NE1 8ST, ian.postlethwaite@northumbria.ac.uk.

Preprint submitted to 49th IEEE Conference on Decision and Control.
Received March 30, 2010.

where k is a current time index; $\mathbf{x}_k$ is a state vector, $\mathbf{u}_k$ is a control input, $\mathbf{w}_{k-1}$ and $\mathbf{v}_k$ are the process noise and observation noise respectively. The noises are assumed to be Gaussian-distributed random variables with zero mean and covariances of $\mathbf{Q}_{k-1}$ and $\mathbf{R}_k$.

$$\begin{aligned}\mathbf{w}_{k-1} &= \mathcal{N}(\mathbf{0},\mathbf{Q}_{k-1})\\ \mathbf{v}_k &= \mathcal{N}(\mathbf{0},\mathbf{R}_k)\end{aligned}$$

2) Estimation Process: The EKF estimates the state vector using the process and observation models together with the assumptions on the process and the observation noise.

The following set of equations summarize the overall EKF algorithm. The prediction stage can be expressed as

$$\mathbf{x}_{k|k-1} = \mathbf{f}[\widehat{\mathbf{x}}_{k-1|k-1},\mathbf{u}_{k-1}] \quad (3)$$

$$\mathbf{P}_{k|k-1} = \nabla\mathbf{f}_x\mathbf{P}_{k-1|k-1}\nabla\mathbf{f}_x^T + \nabla\mathbf{f}_w\mathbf{Q}_{k-1|k-1}\nabla\mathbf{f}_w^T \quad (4)$$

The update stage can be expressed as,

$$\widehat{\mathbf{x}}_{k|k} = \widehat{\mathbf{x}}_{k|k-1} + \mathbf{W}_k\nu_k \quad (5)$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{W}_k\mathbf{S}_{k|k-1}\mathbf{W}_k^T \quad (6)$$

where

$$\begin{aligned}\nu_k &= \mathbf{z}_k - \mathbf{h}[\widehat{\mathbf{x}}_{k|k-1}]\\ \mathbf{S}_{k|k-1} &= \nabla\mathbf{h}_x\mathbf{P}_{k|k-1}\nabla\mathbf{h}_x^T + \mathbf{R}_k\\ \mathbf{W}_k &= \mathbf{P}_{k|k-1}\nabla\mathbf{h}_x^T\mathbf{S}_{k|k-1}^{-1}\end{aligned}$$

The Jacobians are evaluated at the best available estimate $\widehat{\mathbf{x}}_{k-1|k-1}$ for $\nabla\mathbf{f}_x$ and $\widehat{\mathbf{x}}_{k|k-1}$ for $\nabla\mathbf{h}_x$.

B. The Cubature Kalman Filter (CKF) [6]

CKF is a recently developed Bayesian filter which can be used for state estimation of nonlinear systems [6]. Unlike the extended Kalman filter, this filter does not require the evaluation of Jacobians during the estimation process. This section describes the basics of Bayesian filtering and CKF.

In Bayesian filtering, the posterior density of the state of a system is used to evaluate the time and measurement updates. The filter evaluates the statistical expectation of these conditional densities through multi-dimensional integrals, providing an optimal analytical solution to nonlinear filtering problems. However, the solution of these integrals involves a great computational cost. Therefore, to obtain a practical solution for non-linear filters, approximations are made to Bayesian filtering. The use of EKF is not the best choice for many practical applications as it works well only in a 'mild' nonlinear environment, and hence can degrade the performance. The CKF is the closest known approximation to the Bayesian filter that could be designed in a nonlinear setting under the Gaussian assumption.

1) The CKF Theory: Consider the nonlinear system with additive noise defined by process and observation models in (1) and (2). The key assumption of the CKF is that the predictive density $p(\mathbf{x}_k|D_{k-1})$ and the filter likelihood density $p(\mathbf{z}_k|D_k)$ are both Gaussian, which eventually leads to a Gaussian posterior density $p(\mathbf{x}_k|D_k)$. Under this assumption, the CKF solution reduces to how to compute their means and covariances more accurately.

2) Prediction: In the prediction step, the CKF computes the mean $\widehat{\mathbf{x}}_{k|k-1}$ and the associated covariance $P_{k|k-1}$ of the Gaussian predictive density numerically using cubature rules. The predicted mean can be written as

$$\widehat{\mathbf{x}}_{k|k-1} = \mathrm{E}[\mathbf{x}_k|\mathbf{D}_{k-1}] \quad (7)$$

$$\widehat{\mathbf{x}}_{k|k-1} = \mathrm{E}[\mathbf{f}(\mathbf{x}_{k-1},\mathbf{u}_{k-1}) + \mathbf{w}_{k-1}|D_{k-1}] \quad (8)$$

where D_{k-1} represents the history of inputs and observations up to time $k-1$.

Since the $\mathbf{w}_k$ is assumed to be zero-mean and uncorrelated with the measurement sequence, we get

$$\widehat{\mathbf{x}}_{k|k-1} = \mathrm{E}[\mathbf{f}(\mathbf{x}_{k-1},\mathbf{u}_{k-1})|D_{k-1}] \quad (9)$$

$$= \int_{R^n}\mathbf{f}(\mathbf{x}_{k-1},\mathbf{u}_{k-1})p(\mathbf{x}_{k-1}|D_{k-1})d\mathbf{x}_{k-1} \quad (10)$$

$$= \int_{R^n}\mathbf{f}(\mathbf{x}_{k-1},\mathbf{u}_{k-1})\mathcal{N}(\mathbf{x}_{k-1};\widehat{\mathbf{x}}_{k-1|k-1},P_{k-1|k-1})d\mathbf{x}_{k-1} \quad (11)$$

Similarly, the associated error covariance an be represented as

$$P_{k|k-1} = \mathrm{E}[(\mathbf{x}_k - \widehat{\mathbf{x}}_{k|k-1}),(\mathbf{x}_k - \widehat{\mathbf{x}}_{k|k-1})^T|z] \quad (12)$$

$$= \int_{R^n}\mathbf{f}(\mathbf{x}_{k-1},\mathbf{u}_{k-1})\mathbf{f}^T(\mathbf{x}_{k-1},\mathbf{u}_{k-1})\mathcal{N}(\mathbf{x}_{k-1};\widehat{\mathbf{x}}_{k-1|k-1},P_{k-1|k-1})d\mathbf{x}_{k-1} - \widehat{\mathbf{x}}_{k|k-1}\widehat{\mathbf{x}}_{k|k-1}^T + Q \quad (13)$$

3) Update: The predicted measurement density can be represented by

$$p(\mathbf{z}_k|D_{k-1}) = \mathcal{N}(\mathbf{z}_k;\widehat{\mathbf{z}}_{k|k-1},P_{zz,k|k-1}) \quad (14)$$

where the predicted measurement and associated covariance are given by

$$\widehat{\mathbf{z}}_{k|k-1} = \int_{R^n}\mathbf{h}(\mathbf{x}_k,\mathbf{u}_k)\mathcal{N}(\mathbf{x}_k;\widehat{\mathbf{x}}_{k|k-1},P_{k|k-1})d\mathbf{x}_k \quad (15)$$

$$P_{zz,k|k-1} = \int_{R^n}\mathbf{h}(\mathbf{x}_k,\mathbf{u}_k)\mathbf{h}^T(\mathbf{x}_k,\mathbf{u}_k)\mathcal{N}(\mathbf{x}_k;\widehat{\mathbf{x}}_{k|k-1},P_{k|k-1})d\mathbf{x}_k - \widehat{\mathbf{z}}_{k|k-1}\widehat{\mathbf{z}}_{k|k-1}^T + R \quad (16)$$

The Gaussian conditional density of the joint state and the measurement can be represented as

$$p(\mathbf{x}_k^T|D_{k-1}) = \mathcal{N}\left[\begin{pmatrix}\widehat{\mathbf{x}}_{k|k-1}\\ \widehat{\mathbf{z}}_{k|k-1}\end{pmatrix},\begin{pmatrix}P_{k|k-1} & P_{xz,k|k-1}\\ P_{xz,k|k-1}^T & P_{zz,k|k-1}\end{pmatrix}\right] \quad (17)$$

where the cross-covariance is

$$P_{xz,k|k-1} = \int_{R^n}\mathbf{x}_k\mathbf{h}^T(\mathbf{x}_k,\mathbf{u}_k)\mathcal{N}(\mathbf{x}_k;\widehat{\mathbf{x}}_{k|k-1},P_{k|k-1})d\mathbf{x}_k - \widehat{\mathbf{x}}_{k|k-1}\widehat{\mathbf{z}}_{k|k-1}^T \quad (18)$$

Once the new measurement $\mathbf{z}_k$ is received, the CKF computes the posterior density $p(\mathbf{x}_k|D_k)$ yielding

$$p(\mathbf{x}_k|D_k) = (\mathbf{x}_k;\widehat{\mathbf{x}}_{k|k},P_{k|k}), \quad (19)$$

where

$$\widehat{\mathbf{x}}_{k|k} = \widehat{\mathbf{x}}_{k|k-1} + G_k(\mathbf{z}_k - \widehat{\mathbf{z}}_{k|k-1}) \quad (20)$$

$$P_{k|k} = P_{k|k-1} - G_kP_{zz,k|k-1}G_k^T \quad (21)$$

with the Kalman gain given as

$$G_k = P_{xz,k|k-1}P_{zz,k|k-1}^{-1} \quad (22)$$

Preprint submitted to 49th IEEE Conference on Decision and Control.
Received March 30, 2010.

It can be seen that the in the above prediction and correction equations, the Bayesian filter solution reduces to computing the multi-dimensional integrals, whose integrands of the form $nonlinear function \times Gaussian$. The heart of the CKF is to find the multi-dimensional integrals using cubature rules.

4) Cubature Rules: The cubature rule to approximate an n-dimensional Gaussian weighted integral is as follows

$$\int_{R^n} \mathbf{f}(\mathbf{x}) \mathcal{N}(\mathbf{x}; \mu, P) d\mathbf{x} \approx \frac{1}{2n} \sum_{i=1}^{2n} \mathbf{f}(\mu + \sqrt{P}\xi_i) \quad (23)$$

where a square-root factor of the covariance P satisfies the relation $P = \sqrt{P}\sqrt{P}^T$ and the set of 2n cubature points are given by

$$\xi_i = \begin{cases} \sqrt{n}\mathbf{e}_i, & i = 1,2,...n \\ -\sqrt{n}\mathbf{e}_{i_n} & i = n+1, n+2, ...2n \end{cases} \quad (24)$$

where n and $\mathbf{e}_i$ represents the number of states and $i-th$ elementary column vector, respectively.

See [6] for a detailed problem formulation and a solution for CKF.

III. Simultaneous Localization and Mapping

This section presents the mathematical framework employed in the study of the SLAM problem. The fundamental equations for EKF based SLAM are presented in Section III-B. However, one can see [3] for a more detailed explanation. The proposed SLAM using CKF is detailed in Section III-C.

SLAM can be performed by storing the vehicle pose and landmarks in a single state vector, and estimating it by a recursive process of prediction and update. In SLAM, the vehicle starts typically at an unknown location without *a priori* knowledge of landmark locations. The vehicle is mounted with a sensor which is capable of identifying the landmarks. The most common sensor used for SLAM is a laser, which takes the observation of the landmarks and outputs the range and bearing of the landmarks. While continuing in motion, the vehicle builds a complete map of landmarks and uses these to provide estimates of the vehicle location. By using the relative position between the vehicle and landmarks in the environment, both the position of the vehicle and the position of the features or landmarks can be estimated simultaneously.

A. The Vehicle, Landmark and Sensor Models

1) Vehicle Model: The vehicle model used in this paper is the common bicycle model, assuming that the control inputs are given by the wheel velocity, V_{k-1}, and steering angle, γ_{k-1} and L is the distance between the front or rear set of wheels [4], [8]. The vehicle's state vector represents its location and orientation.

$$\begin{bmatrix} \mathbf{x}_{v_k} \end{bmatrix} = \begin{bmatrix} x_{v_k} \\ y_{v_k} \\ \phi_{v_k} \end{bmatrix} = \begin{bmatrix} x_{v_{k-1}} + \Delta T V_{k-1} \cos(\phi_{v_{k-1}} + \gamma_{k-1}) \\ y_{v_{k-1}} + \Delta T V_{k-1} \sin(\phi_{v_{k-1}} + \gamma_{k-1}) \\ \phi_{v_{k-1}} + \Delta T V_{k-1} \frac{\sin(\gamma_{k-1})}{L} \end{bmatrix}$$

In the above equations, the process noise $\mathbf{w}_{k-1}$ is eliminated. One popular way to include the process noise in the process model is to insert the noise terms into the control signal $\mathbf{u}$ such that

$$\mathbf{u}_{k-1} = \mathbf{u}_{n_{k-1}} + \mathbf{w}_{k-1} \quad (25)$$

where $\mathbf{u}_{n_{k-1}}$ is a nominal control signal and $\mathbf{w}_{k-1}$ is a zero mean Gaussian distribution noise vector with covariance matrix, $\mathbf{Q}$.

2) Landmark Model: In the context of SLAM, a landmark is a feature of the environment that can be observed using vehicle's sensor. Different kinds of landmark are used in SLAM like point landmarks, corners, lines, etc. In our case, we assumed the landmarks as point features. For the SLAM algorithm, the feature states are assumed to be stationary. Landmarks can be represented by the following expression

$$\mathbf{x}_{m_k} = \mathbf{x}_{m_{k-1}} \quad (26)$$

The SLAM map is defined by an augmented state vector formed by the concatenation of the vehicle and feature map state.

$$\mathbf{x}_a(k+1) = \begin{bmatrix} \mathbf{x}_{v_k}^T & \mathbf{x}_{m(k+1)}^T \end{bmatrix}^T \quad (27)$$

3) Sensor Model: The state observation process can also be modelled by a nonlinear function in the form

$$\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{v}_k \quad (28)$$

where $\mathbf{v}_k$ is the observation noise assumed to be zero mean, Gaussian distribution random variable with a covariance of $\mathbf{R}$. In this paper, it is assumed that the sensor is equipped with a range-bearing sensor that takes observations of the features of the environment. Laser and sonar sensors are two examples of range-bearing sensors that can be used on a vehicle. Given the current vehicle position $\mathbf{x}_{v_k}$ and the position of an observed feature $\mathbf{x}_{m_k}$, the range and bearing can be modelled as

$$\mathbf{z}_k = \begin{bmatrix} \mathbf{z}_{R_k} \\ \mathbf{z}_{\theta_k} \end{bmatrix} = \begin{bmatrix} \sqrt{(x_{v_k} - x_{i_k})^2 + (y_{v_k} - y_{i_k})^2} \\ \arctan \frac{y_{v_k} - y_{i_k}}{x_{v_k} - x_{i_k}} \end{bmatrix} + \begin{bmatrix} v_{r_k} \\ v_{\theta_k} \end{bmatrix} \quad (29)$$

B. EKF SLAM

1) The Estimation Process: The estimate of the vehicle and its covariance can be given as

$$\widehat{\mathbf{x}}_v = \begin{bmatrix} \widehat{x}_v & \widehat{y}_v & \widehat{\phi}_v \end{bmatrix}^T \quad (30)$$

$$\mathbf{P}_v = \begin{bmatrix} \sigma^2_{x_v x_v} & \sigma^2_{x_v y_v} & \sigma^2_{x_v \phi_v} \\ \sigma^2_{x_v y_v} & \sigma^2_{y_v y_v} & \sigma^2_{y_v \phi_v} \\ \sigma^2_{x_v \phi_v} & \sigma^2_{y_v \phi_v} & \sigma^2_{\phi_v \phi_v} \end{bmatrix} \quad (31)$$

Preprint submitted to 49th IEEE Conference on Decision and Control.
Received March 30, 2010.

Similarly, the estimate and its covariance of the feature locations can be given as

$$\widehat{\mathbf{x}}_m = \begin{bmatrix} \widehat{x}_1 & \widehat{y}_1 & \dots & \widehat{x}_n & \widehat{y}_n \end{bmatrix}^T \tag{32}$$

$$\mathbf{P}_m = \begin{bmatrix} \sigma^2_{x_1x_1} & \sigma^2_{x_1y_1} & \dots & \sigma^2_{x_1x_n} & \sigma^2_{x_1y_n} \\ \sigma^2_{x_1y_1} & \sigma^2_{y_1y_1} & \dots & \sigma^2_{y_1x_n} & \sigma^2_{y_1y_n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \sigma^2_{x_1x_n} & \sigma^2_{y_1x_n} & \dots & \sigma^2_{x_nx_n} & \sigma^2_{x_ny_n} \\ \sigma^2_{x_1y_n} & \sigma^2_{y_1y_n} & \dots & \sigma^2_{x_ny_n} & \sigma^2_{y_ny_n} \end{bmatrix} \tag{33}$$

The augmented vector formed by concatenation of vehicle states and feature states $\widehat{x}_a$ and its corresponding covariance $\mathbf{P}_a$ can be written as

$$\widehat{\mathbf{x}}_a = \begin{bmatrix} \widehat{\mathbf{x}}_v^T & \widehat{\mathbf{x}}_m^T \end{bmatrix}^T \tag{34}$$

$$\mathbf{P}_a = \begin{bmatrix} \mathbf{P}_v & \mathbf{P}_{vm} \\ \mathbf{P}_{vm}^T & \mathbf{P}_m \end{bmatrix} \tag{35}$$

2) Prediction: The prediction stage uses the model of the motion of the vehicle to generate an estimate of vehicle states along with its covariance matrix. The estimates of the SLAM states and its covariance matrix can be expressed as

$$\mathbf{x}_{k|k-1} = \mathbf{f}[\widehat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}, k-1] \tag{36}$$

$$\mathbf{P}_{k|k-1} = \nabla\mathbf{f}_x \mathbf{P}_{k-1|k-1} \nabla\mathbf{f}_x^T + \nabla\mathbf{f}_w \mathbf{Q}_{k-1|k-1} \nabla\mathbf{f}_w^T \tag{37}$$

where the Jacobians $\nabla\mathbf{f}_x$ and $\nabla\mathbf{f}_w$ can be given as

$$\nabla\mathbf{f}_x = \begin{bmatrix} 1 & 0 & -\Delta T \mathbf{V}_{k-1} \sin(\phi_{v_{k-1}} + \gamma_{k-1}) \\ 0 & 1 & \Delta T \mathbf{V}_{k-1} \cos(\phi_{v_{k-1}} + \gamma_{k-1}) \\ 0 & 0 & 1 \end{bmatrix} \tag{38}$$

$$\nabla\mathbf{f}_w = \begin{bmatrix} \Delta T \cos(\phi_{v_{k-1}} + \gamma_{k-1}) & -\Delta T \mathbf{V}_{k-1} \sin(\phi_{v_{k-1}} + \gamma_{k-1}) \\ \Delta T \sin(\phi_{v_{k-1}} + \gamma_{k-1}) & \Delta T \mathbf{V}_{k-1} \cos(\phi_{v_{k-1}} + \gamma_{k-1}) \\ \Delta T \frac{\sin(\gamma_{k-1})}{L} & \mathbf{V} \Delta T \frac{cos(\gamma_{k-1})}{L} \end{bmatrix} \tag{39}$$

3) Update: In SLAM, the vehicle observes the relative range and bearing between itself and features in the environment. Given the current position estimate and the estimated position of the observed feature, the predicted observation of range and bearing, between the vehicle and the feature can be computed using

$$\widehat{\mathbf{z}}_k = \mathbf{h}(\widehat{\mathbf{x}}_a) = \begin{bmatrix} \sqrt{(\widehat{x}_{v_k} - \widehat{x}_{i_k})^2 + (\widehat{y}_{v_k} - \widehat{y}_{i_k})^2} \\ \arctan \frac{\widehat{y}_{v_k} - \widehat{y}_{i_k}}{\widehat{x}_{v_k} - \widehat{x}_{i_k}} - \widehat{\phi}_{v_k} \end{bmatrix} \tag{40}$$

The update stage of SLAM is similar to that of the Kalman filter, which involves the computation of innovation covariance and Kalman gain. Once the observation is associated with a particular feature in the map, the state estimate can be updated using the optimal gain matrix $\mathbf{W}_k$. This gain matrix provides the sum of the prediction and observation and is computed using the innovation covariance $\mathbf{S}_k$ and the predicted covariance. The state vector and covariance matrix update can be evaluated using (5) and (6).

4) State Augmentation: As the environment is explored, new features are observed and must be added to the stored map. The state vector and covariance matrix are augmented with the polar values of the new observation $\mathbf{z}$, and its covariance $\mathbf{R}$.

$$\widehat{\mathbf{x}}_{aug} = \begin{bmatrix} \widehat{\mathbf{x}}_a & \mathbf{z} \end{bmatrix}^T \tag{41}$$

$$\mathbf{P}_{aug} = \begin{bmatrix} \mathbf{P}_v & \mathbf{P}_{vm} & \mathbf{0} \\ \mathbf{P}_{vm}^T & \mathbf{P}_m & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{R} \end{bmatrix} \tag{42}$$

The function $\mathbf{g}_i(\mathbf{x}_v, \mathbf{z})$ is used to convert the polar observation $\mathbf{z}$ to a global cartesian feature location. This transformation is a function of the new observation and the current vehicle pose.

$$\mathbf{g}_i(\mathbf{x}_v, \mathbf{z}) = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} x_v + z_r \cos(z_\theta + \phi_v) \\ y_v + z_r \sin(z_\theta + \phi_v) \end{bmatrix} \tag{43}$$

where, 'i' denotes the number of features. The augment state can be initialized to the correct values by performing a linearized transformation by the function $\mathbf{f}_i$ as follows

$$\mathbf{f}_i(\widehat{\mathbf{x}}_{aug}) = \begin{bmatrix} \widehat{\mathbf{x}}_a & \mathbf{g}_i(\widehat{\mathbf{x}}_v, \mathbf{z}) \end{bmatrix}^T \tag{44}$$

$$\mathbf{P}_a = \nabla\mathbf{f}_{x_{aug}} \mathbf{P}_{aug} \nabla\mathbf{f}_{x_{aug}}^T \tag{45}$$

where the Jacobian $\nabla\mathbf{f}_{x_{aug}}$ is given by

$$\nabla\mathbf{f}_{x_{aug}} = \left.\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_{aug}}\right|_{\widehat{x}_{aug}} = \begin{bmatrix} \mathbf{I}_v & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_m & \mathbf{0} \\ \nabla\mathbf{g}_{x_v} & \mathbf{0} & \nabla\mathbf{g}_\mathbf{z} \end{bmatrix} \tag{46}$$

and the Jacobians $\nabla\mathbf{g}_{x_v}$ and $\nabla\mathbf{g}_\mathbf{z}$ are as follows

$$\nabla\mathbf{g}_{x_v} = \left.\frac{\partial \mathbf{g}_i}{\partial \mathbf{x}_v}\right|_{(\widehat{x}_v, \mathbf{z})} = \begin{bmatrix} 1 & 0 & -z_r \sin(z_\theta + \phi_v) \\ 0 & 1 & z_r \cos(z_\theta + \phi_v) \end{bmatrix} \tag{47}$$

$$\nabla\mathbf{g}_\mathbf{z} = \left.\frac{\partial \mathbf{g}_i}{\partial \mathbf{z}}\right|_{(\widehat{x}_v, \mathbf{z})} = \begin{bmatrix} \cos(z_\theta + \phi_v) & -z_r \sin(z_\theta + \phi_v) \\ \sin(z_\theta + \phi_v) & z_r \cos(z_\theta + \phi_v) \end{bmatrix} \tag{48}$$

C. CKF SLAM

This section describes the use of CKF for estimating the state vector of SLAM. As compared to EKF SLAM, this approach need not require the evaluation of Jacobians during the prediction and update stage, which makes this approach more promising. In this paper, we have used the CKF for state estimation. However, one can use square-root CKF for SLAM application. The CKF SLAM is detailed in Algorithm 1. During the prediction stage, the state vector is augmented with the control inputs and the error covariance matrix with process noise covariance matrix, $\mathbf{Q}$. The evaluation of `cub_cal` function is described in algorithm 2; the `veh_model` represents the state equations of vehicle model. In the update stage, the cubature point array and square root of the error covariance matrix ($\mathbf{S}$) are evaluated using Cholesky factorization (steps 1-2). Once the cubature points are evaluated, they are propagated in the observation model

Preprint submitted to 49th IEEE Conference on Decision and Control.
Received March 30, 2010.

(steps 3-4). Then the predicted measurement and covariance matrices are evaluated using steps 5-7. The Kalman gain is evaluated using the evaluated covariance matrices (step 8). The updated state vector and corresponding covariance matrix can be evaluated using step 9.

Algorithm 1 CKF SLAM

Prediction

1: Augment the state vector and covariance matrix

$$\begin{aligned}\widehat{\mathbf{x}}_{k-1|k-1} &= \begin{bmatrix} \widehat{\mathbf{x}}_v & \mathbf{v}_{k-1} & \gamma_{k-1} \end{bmatrix}^T \\ \mathbf{P}_{k-1|k-1} &= \begin{bmatrix} \mathbf{P}_{k-1|k-1} & 0 \\ 0 & Q \end{bmatrix}\end{aligned}$$

2: Predict the state vector and covariance matrix

$$\begin{bmatrix} \widehat{\mathbf{x}}_{k|k-1} & \mathbf{P}_{k|k-1} \end{bmatrix} = \texttt{cub\_cal}(\texttt{veh\_model}, \widehat{\mathbf{x}}_{k-1|k-1}, \mathbf{P}_{k-1|k-1})$$

Measurement Update

1: Evaluate the cubature point array; cpa (i =1,2,...,2n)

$$\mathbf{cpa}_i = \sqrt{(n)} \begin{bmatrix} I_n & -I_n \end{bmatrix}$$

2: Evaluate the square root of P

$$S_{k|k-1} = \mathbf{chol}(P_{k|k-1})^T$$

3: Evaluate the cubature points $(i = 1, 2, ..., 2n)$

$$\mathbf{cp}_{i,k|k-1} = \widehat{\mathbf{x}}_{k|k-1} + S_{k|k-1}\mathbf{cpa}_i$$

4: Evaluate the propagated cubature points $(i = 1, 2, ..., 2n)$ in observation model

$$\boldsymbol{Z}^*_{i,k|k-1} = \mathbf{h}(\mathbf{cp}_{i,k|k-1}, \mathbf{u}_k)$$

5: Estimate the predicted measurement

$$\widehat{\mathbf{z}}_{i,k|k-1} = \frac{1}{2n}\sum_{i=1}^{2n} \boldsymbol{Z}_{i,k|k-1}$$

6: Estimate the innovative covariance matrix

$$P_{zz,k|k-1} = \frac{1}{2n}\sum_{i=1}^{2n} \boldsymbol{Z}_{i,k|k-1}\boldsymbol{Z}^T_{i,k|k-1} - \widehat{\mathbf{z}}_{k|k-1}\widehat{\mathbf{z}}^T_{k|k-1} + R_k$$

7: Estimate the cross covariance matrix

$$P_{xz,k|k-1} = \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{X}_{i,k|k-1}\boldsymbol{Z}^T_{i,k|k-1} - \widehat{\mathbf{x}}_{k|k-1}\widehat{\mathbf{z}}^T_{k|k-1}$$

8: Estimate the Kalman Gain

$$W_k = P_{xz,k|k-1}P^{-1}_{zz,k|k-1}$$

9: Estimate the updated state and corresponding covariance matrix

$$\begin{aligned}\widehat{\mathbf{x}}_{k|k} &= \widehat{\mathbf{x}}_{k|k-1} + W_k(\mathbf{z}_k - \widehat{\mathbf{z}}^T_{k|k-1}) \\ P_{k|k} &= P_{k|k-1} - W_k P_{zz,k|k-1} W_k^T.\end{aligned}$$

During the process of state augmentation, first the state vector is augmented with observations and the corresponding error covariance matrix with measurement noise covariance matrix. The augmented state vector of vehicle states and observed landmarks, and the corresponding covariance matrix can be evaluated using `cub_cal` function, which is detailed in Algorithm 2.

Algorithm 2 State Augmentation

Augmentation

1: Augment the state vector(with observation) and covariance matrix(with measurement noise covariance matrix)

$$\begin{aligned}\mathbf{x}_k &= \begin{bmatrix} x_{v_k} & z_k \end{bmatrix}^T \\ \mathbf{P}_k &= \begin{bmatrix} \mathbf{P}_k & 0 \\ 0 & R \end{bmatrix}\end{aligned}$$

2: Evaluate the augmented state vector and corresponding covariance matrix

$$\begin{aligned}\texttt{aug\_model} &= \begin{bmatrix} \mathbf{x}_v \\ x_v + z_r\cos(z_\theta + \phi_v) \\ y_v + z_r\sin(z_\theta + \phi_v) \end{bmatrix} \\ \begin{bmatrix} \mathbf{x}_a & \mathbf{P}_a \end{bmatrix} &= \texttt{cub\_cal}(\texttt{aug\_model}, \mathbf{x}, \mathbf{P})\end{aligned}$$

function `cub_cal`

1: Evaluate the cubature point array; cpa (i =1,2,...,2n)

$$\mathbf{cpa}_i = \sqrt{(n)} \begin{bmatrix} I_n & -I_n \end{bmatrix}$$

2: Evaluate the square root of P

$$S = \mathbf{chol}(P)^T$$

3: Evaluate the cubature points $(i = 1, 2, ..., 2n)$

$$\mathbf{cp}_i = \widehat{\mathbf{x}} + S\mathbf{cpa}_i$$

4: Evaluate the propagated cubature points $(i = 1, 2, ..., m)$

$$\boldsymbol{X}_i = \mathbf{f}(\mathbf{cp}_i, \mathbf{u})$$

5: Estimate the state vector and corresponding covariance matrix

$$\begin{aligned}\widehat{\mathbf{x}} &= \frac{1}{2n}\sum_{i=1}^{2n} \boldsymbol{X}_i \\ P &= \frac{1}{2n}\sum_{i=1}^{2n} \boldsymbol{X}_i\boldsymbol{X}_i^T - \widehat{\mathbf{x}}\widehat{\mathbf{x}}^T\end{aligned}$$

IV. Simulation Results

This section includes simulation results of SLAM using EKF and CKF. The basic EKF SLAM package is available in [9], and is modified for the CKF SLAM. The process and observation models used for the simulations are given in Section III-A. In the following numerical experiments, the velocity of the vehicle is $3m/s$, the maximum steering angle is $-30^\circ < \gamma < 30^\circ$ and the maximum rate of change in steer angle is 20 deg/sec. The controls are updated every 0.025 seconds and observations occur every 0.2 seconds. The range-bearing sensor has a forward-facing 180° field-of-view and maximum range of 30 metres. The trajectory of the vehicle is known and sixty two landmarks were randomly spread. The measure of the filter consistency is examined by the average error norm ($\bar{J}$) over N Monte Carlo simulations. The error norm of the vehicle positions is given as

$$J_k = \sqrt{(x_{v_k} - \widehat{x}_{v_k})^2 + (y_{v_k} - \widehat{y}_{v_k})^2} \tag{49}$$

and the average error norm over N Monte Carlo simulations is computed as

$$\bar{J} = \frac{1}{N}\sum_{i=1}^{N} J_{ik} \tag{50}$$

Preprint submitted to 49th IEEE Conference on Decision and Control.
Received March 30, 2010.

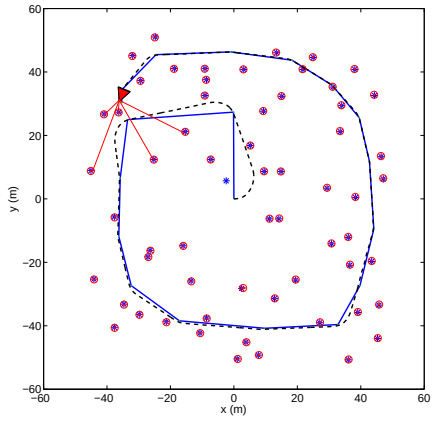

Fig. 1. EKF SLAM with low Gaussian noise, '*' and 'o' represents the actual and estimated landmarks, solid line represents the waypoints (trajectory) and dashed line represents the estimated trajectory

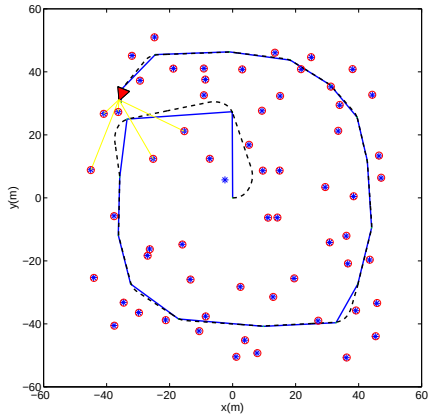

Fig. 2. CKF SLAM with low Gaussian noise, '*' and 'o' represents the actual and estimated landmarks, solid line represents the way-points (trajectory) and dashed line represents the estimated trajectory

Two sets of simulations were performed to compare the EKF SLAM and CKF SLAM. First with low level Gaussian noise and then with high level Gaussian noise. Each set of simulation includes 10 Monte Carlo runs.

A. SLAM with Low Gaussian Noise

The process noise in this case is $\sigma_v = 0.1m/s, \sigma_\gamma = 0.1^\circ$ and the observation noise is $\sigma_r = 0.1m, \sigma_\theta = 0.1^\circ$. The SLAM results with EKF and CKF are shown in Figs 1 to 2 and the norm error plot is shown in Fig 3. It can be seen that the overall error norm plot of CKF SLAM is less that of EKF SLAM. The performance measure (average norm error) for EKF SLAM is 0.0362 and that of CKF SLAM is 0.0291. Hence, the CKF SLAM performance is better that EKF SLAM.

B. SLAM with High Gaussian Noise

The process noise in this case is $\sigma_v = 1m/s, \sigma_\gamma = 5^\circ$ and the observation noise is $\sigma_r = 1m, \sigma_\theta = 5^\circ$. The norm error plot with high Gaussian noise is shown in Fig 4. The average norm error of EKF SLAM is 0.9508 and CKF SLAM is 0.8647. Hence, CKF SLAM outperforms EKF SLAM in this case.

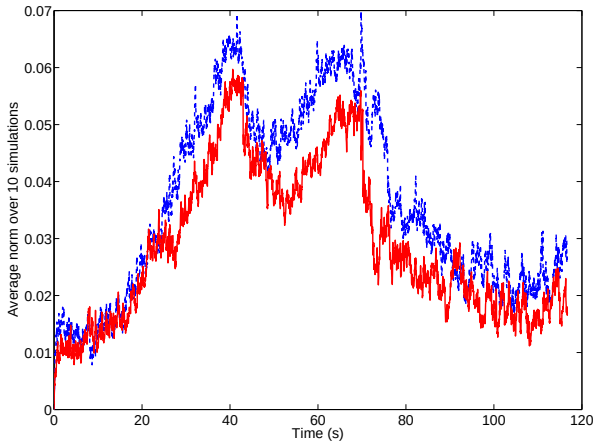

Fig. 3. Average norm of the vehicle positions $[x_{v_k}, y_{v_k}]^T$ over 10 simulations with $\sigma_v = 0.1m/s, \sigma_\gamma = 0.1^\circ$ and $\sigma_r = 0.1m, \sigma_\theta = 0.1^\circ$. Solid line represents CKF SLAM and the dashed line represents EKF SLAM.

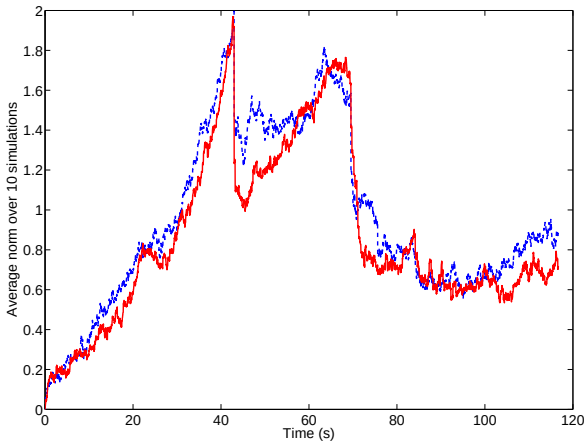

Fig. 4. Average norm of the vehicle positions $[x_{v_k}, y_{v_k}]^T$ over 10 simulations with $\sigma_v = 1m/s, \sigma_\gamma = 5^\circ$ and $\sigma_r = 1m, \sigma_\theta = 5^\circ$. Solid line represents CKF SLAM and the dashed line represents EKF SLAM.

It can also be noted that, in general the computational cost of both EKF and CKF grows cubically in terms of number of state variables [6]. However, the scaling factor of EKF is slightly less than than CKF and hence CKF SLAM is slightly more computationally expensive than the EKF SLAM.

V. Conclusions and Future Work

In this paper we have proposed the use of a cubature Kalman filter for SLAM. The proposed CKF SLAM algorithm does not requires the evaluation of Jacobians during the prediction and update stage. The efficacy of the algorithm is verified by simulations. Two types of Gaussian noise are

Preprint submitted to 49th IEEE Conference on Decision and Control.
Received March 30, 2010.

used in the simulations and it is shown that CKF SLAM outperforms EKF SLAM in both cases.

In this work, the kinematic model of the vehicle is used for the SLAM. When applying this CKF SLAM to fully nonlinear 6-DOF model of UAV, the proposed approach may outperform all other existing SLAM algorithms. The trajectory of the vehicle is assumed to be known in this work. However, for complete autonomous operation, the online trajectory has to be simultaneously generated during exploration.

REFERENCES

[1] R. Smith and P. Cheeseman, "Estimating Uncertain Spatial Relationships in Robotics," in Autonomous Robot Vehicles, I.J. Cox and G.T. Wilfon, Eds. New York: Springer-Verlag, 1990, pp. 167193.

[2] M.W.M.G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, " A Solution to the Simultaneous Localization and Map Building (SLAM) Problem," IEEE Tramsactions on Robotics and Automation, vol. 17, no.3, 2001, pp. 229-241.

[3] T. Bailey, "Mobile Robot Localisation and Mapping in Extensive Outdoor Environments ", PhD thesis, University of Sydney, 2002.

[4] S. Julier, "Process Models for the Navigation of High Speed Land Vehicles", PhD thesis, University of Oxford, 1996.

[5] R. Martinez-Cantin and J.A. Castellanos, "Unscented SLAM for Large-scale Outdoor Environments," in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005, pp. 3427-3432.

[6] I. Arasaratnam and S. Haykin, "Cubature Kalman Filters," IEEE Transactions on Automatic Control, vol. 54, no. 6, 2009, pp. 1254-1269.

[7] P.S. Maybeck, "Stochastic Models, Estimation and Control," Academic Press, vol. 1, 1979.

[8] T. Bailey, J. Nieto, J. Guivant, M. Stevens and E. Nobot, "Consistency of EKF-SLAM Algorithm", in Proceedings of the Intelligent Robots and Systems, 2006, pp. 3562-3568.

[9] [Online]. Available: http://www-personal.acfr.usyd.edu.au/tbailey /software/slam_simulations.htm. Jan. 2010.

Preprint submitted to 49th IEEE Conference on Decision and Control.
Received March 30, 2010.