Stochastic Integration Based Estimator: Robust Design and Stone Soup Implementation

Jindřich Duník, Jakub Matoušek, Ondřej Straka
Dept. of Cyber., Univ. of West Bohemia
Univerzitní 8, 306 14 Pilsen, Czech Rep.
Email: {dunikj, matoujak, straka30}@kky.zcu.cz

Erik Blasch
MOVEJ Analytics
Fairborn, OH USA
erik.blasch@gmail.com

John Hiles, Ruixin Niu
Dept. of ECE
Virginia Commonwealth University
Email: {hilesj, rniu}@vcu.edu

***Abstract*—This paper deals with state estimation of nonlinear stochastic dynamic models. In particular, the stochastic integration rule, which provides asymptotically unbiased estimates of the moments of nonlinearly transformed Gaussian random variables, is reviewed together with the recently introduced stochastic integration filter (SIF). Using SIF, the respective multi-step prediction and smoothing algorithms are developed in *full* and efficient *square-root* form. The stochastic-integration-rule-based algorithms are implemented in Python (within the Stone Soup framework) and in MATLAB® and are numerically evaluated and compared with the well-known unscented and extended Kalman filters using the Stone Soup defined tracking scenario.**

Keywords: Stochastic integration rule; Nonlinear systems; State estimation; Filtering; Prediction, Smoothing; Stone Soup.

I. Introduction

State estimation of discrete-time stochastic dynamic systems from noisy or incomplete measurements has been a subject of considerable research interest for the last decades. State estimation plays an important role in various fields such as navigation, speech and image processing, fault detection, optimal control, and tracking [28].

Following the Bayesian approach, a general solution to the state estimation problem is given by the functional recursive relations (FRRs) for computing the probability density functions (PDFs) of the state conditioned on the measurements. The conditional PDFs provide a full description of the immeasurable state. The relations are, however, analytically tractable for a limited set of models where linearity is usually a common factor for exact Bayesian estimators, e.g., by the Kalman filter (KF) or the Rauch-Tung-Streibel smoother (RTSS). In other cases, an approximate solution to the FRRs has to be employed. These approximate estimation methods can be divided with respect to the validity of the estimates into global and local ones [21], [26].

Global estimators, such as the particle or the point-mass approach, provide estimates in the form of conditional PDFs without any assumption on the conditional distribution family. These estimators are capable of estimating the state of a strongly nonlinear or non-Gaussian system but usually at the cost of higher computational demands.

As opposed to global estimators, *local*, or alternatively *Gaussian*, estimators (GE) provide computationally efficient Gaussian-assumed estimates in the form of the conditional mean and covariance matrix only. Due to these simplifications, the GEs are especially suitable for mildly nonlinear models.

The GE can be further divided into the derivative and derivative-free estimators. Derivative estimators, developed in 70s, the nonlinear model is linearised using the Taylor expansion, which leads to the extended or second-order filter [2]. On the other hand, derivative-free estimators (DFEs) approximate the Gaussian-assumed conditional PDF by a set of weighted points which are propagated through the non-linear model. DFEs, being developed from 2000, are represented by the unscented, divided difference filter, or the cubature Kalman filter/smoother/predictor to name a few [9], [15], [17], [26].

This paper focuses on the *stochastic integration filter* (SIF) [9], [10], [16], [22], [29], which compared to other Gaussian filters, provides an asymptotically exact integral evaluation needed for the conditional mean and covariance matrix calculation. As a consequence, the SIF was shown to provide superior estimation performance, still with acceptable computational complexity. Although, the SIF was thoroughly analysed and various extensions have been proposed (such as Student's t-distribution filter for heavy-tailed densities or H_∞ filter for robust estimation), the areas of multi-step prediction, smoothing, and robust estimation have been left aside in the literature.

The goal of the paper is threefold: *first*, to complete the family of stochastic integration estimators by the design of multi-step predictor and smoother, *second*, to design efficient and robust versions of the estimators in the full and square-root forms, and *third*, to implement the latest versions of the SIF and related predictor and smoother in the publicly available Python-based Stone Soup framework[1]. In addition, the SIF and related predictor and smoother are implemented also in MATLAB as a counterpart to the Python implementation.

The rest of the paper is organised as follows. In Section II, the state-space model and Bayesian estimation is introduced with the stress on a nonlinear Gaussian filter, smoother, and predictor design. Section III introduces the stochastic integration rule for calculation of moments

[1] https://ostewg.isif.org/stone-soup/

of nonlinearly transformed Gaussian random variable and proposes the rule-based estimators. Then, square-root and other efficient formulations of the estimators are developed in Section IV and V. Subsequently, the Stone Soup framework and estimators' implementation are introduced in Section VI. Numerical illustration is provided in Section VII and the concluding remarks are drawn in Section VIII.

II. State-space Model and Bayesian Estimation

In this paper we consider a discrete-time nonlinear stochastic dynamic system, described by the state-space model

$$\mathbf{x}_{k+1} = \mathbf{f}_k(\mathbf{x}_k) + \mathbf{w}_k, \tag{1}$$

$$\mathbf{z}_k = \mathbf{h}_k(\mathbf{x}_k) + \mathbf{v}_k, \tag{2}$$

where the vectors $\mathbf{x}_k \in \mathbb{R}^{n_x}$ and $\mathbf{z}_k \in \mathbb{R}^{n_z}$ represent the state of the system and the measurement at time instant k, respectively, and $k = 0, 1, 2, \ldots, T$. The state and measurement functions $\mathbf{f}_k : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ and $\mathbf{h}_k : \mathbb{R}^{n_x} \to \mathbb{R}^{n_z}$ are supposed to be known. The state noise $\mathbf{w}_k \in \mathbb{R}^{n_x}$, the measurement noise $\mathbf{v}_k \in \mathbb{R}^{n_z}$, and the initial state $\mathbf{x}_0 \in \mathbb{R}^{n_x}$ are supposed to be independent of each other.

The noises and the initial state are assumed to be normally distributed, i.e.,

$$p(\mathbf{w}_k) = \mathcal{N}\{\mathbf{w}_k; \mathbf{0}_{n_x\times 1}, \mathbf{Q}_k\}, \tag{3}$$

$$p(\mathbf{v}_k) = \mathcal{N}\{\mathbf{v}_k; \mathbf{0}_{n_z\times 1}, \mathbf{R}_k\}, \tag{4}$$

$$p(\mathbf{x}_0) = \mathcal{N}\{\mathbf{x}_0; \bar{\mathbf{x}}_0, \mathbf{P}_0\}, \tag{5}$$

where $\mathbf{0}_{n_x\times 1} \in \mathbb{R}^{n_x\times 1}$ is a zero vector and the notation $\mathcal{N}\{\mathbf{x}; \bar{\mathbf{x}}, \mathbf{P}\}$ stands for the Gaussian PDF of a random variable $\mathbf{x}$ with mean $\bar{\mathbf{x}}$ and covariance matrix $\mathbf{P}$. The first two moments of the random variables in (3)–(5) are supposed to be known.

A. Bayesian State Estimation

The FRRs are given by [2], [26]

$$p(\mathbf{x}_{k+m}|\mathbf{z}^k) = \int p(\mathbf{x}_{k+m}|\mathbf{x}_{k+\ell}) p(\mathbf{x}_{k+\ell}|\mathbf{z}^k) d\mathbf{x}_{k+\ell}, \tag{6}$$

$$p(\mathbf{x}_k|\mathbf{z}^k) = \frac{p(\mathbf{x}_k|\mathbf{z}^{k-1}) p(\mathbf{z}_k|\mathbf{x}_k)}{p(\mathbf{z}_k|\mathbf{z}^{k-1})}, \tag{7}$$

$$p(\mathbf{x}_{k-m}|\mathbf{z}^k) = p(\mathbf{x}_{k-m}|\mathbf{z}^{k-m}) \int \frac{p(\mathbf{x}_{k-m+1}|\mathbf{z}^k)}{p(\mathbf{x}_{k-\ell}|\mathbf{z}^{k-m})} \times p(\mathbf{x}_{k-\ell}|\mathbf{x}_{k-m}) d\mathbf{x}_{k-\ell}. \tag{8}$$

where $m > 0, \ell = m - 1$, and the symbol $\mathbf{z}^k$ represents the set of all measurements up to the time instant k, i.e., $\mathbf{z}^k = [\mathbf{z}_0, \mathbf{z}_1, \ldots \mathbf{z}_k]$. The PDF $p(\mathbf{x}_{k+m}|\mathbf{z}^{k-1})$ is the m-th step *predictive* PDF computed by the Chapman-Kolmogorov equation (6), $p(\mathbf{x}_k|\mathbf{z}^k)$ is the *filtering* PDF computed by the Bayes' rule (7), and $p(\mathbf{x}_{k-m}|\mathbf{z}^k)$ is the smoothed PDF calculated by (8). The PDFs $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ and $p(\mathbf{z}_k|\mathbf{x}_k)$ are the state transition PDF obtained from (1) and the measurement PDF obtained from (2), respectively. The PDF $p(\mathbf{z}_k|\mathbf{z}^{k-1}) = \int p(\mathbf{x}_k|\mathbf{z}^{k-1}) p(\mathbf{z}_k|\mathbf{x}_k) d\mathbf{x}_k$ is the one-step predictive PDF of the measurement. The filtering and one-step predictive recursion (6), (7) can be started from the initial PDF $p(\mathbf{x}_0|\mathbf{z}^{-1}) = p(\mathbf{x}_0)$ and the recursion goes forward in time. The smoothing recursion (8) starts with filtering PDF at the last time instant and the recursion goes backward in time.

Considering the nonlinear description of the system (1)–(2), the FRRs are not exactly solvable. To allow the solution, the GEs assume the predictive conditional joint PDF, i.e.,

$$p(\mathbf{x}_{k+1}, \mathbf{z}_{k+1}|\mathbf{z}^k) \triangleq \mathcal{N}\{\begin{bmatrix}\mathbf{x}_{k+1}\\ \mathbf{z}_{k+1}\end{bmatrix}; \begin{bmatrix}\hat{\mathbf{x}}_{k+1|k}\\ \hat{\mathbf{z}}_{k+1|k}\end{bmatrix}, \begin{bmatrix}\mathbf{P}^{xx}_{k+1|k} & \mathbf{P}^{xz}_{k+1|k}\\ \mathbf{P}^{zx}_{k+1|k} & \mathbf{P}^{zz}_{k+1|k}\end{bmatrix}\}, \tag{9}$$

to be *Gaussian*[2], which allows analytical (but inherently *approximate*) solution to the FRRs leading to the following recursive GE algorithms for prediction, filtering, and smoothing[3], which form the general GE framework.

B. Gaussian Predictor Design

Assuming Gaussian conditional PDFs, the general solution to the multi-step prediction (6) in a GE framework is

$$p(\mathbf{x}_{k+m}|\mathbf{z}^k) \triangleq \mathcal{N}\{\mathbf{x}_{k+m}; \hat{\mathbf{x}}_{k+m|k}, \mathbf{P}^{xx}_{k+m|k}\} \tag{10}$$

with the moments

$$\hat{\mathbf{x}}_{k+m|k} = \mathsf{E}[\mathbf{x}_{k+m}|\mathbf{z}^k] = \int \mathbf{f}_{k+\ell}(\mathbf{x}_{k+\ell}) \mathcal{N}\{\mathbf{x}_{k+\ell}; \hat{\mathbf{x}}_{k+\ell|k}, \mathbf{P}^{xx}_{k+\ell|k}\} d\mathbf{x}_{k+\ell}, \tag{11}$$

$$\mathbf{P}^{xx}_{k+m|k} = \mathsf{E}[(\mathbf{x}_{k+m} - \hat{\mathbf{x}}_{k+m|k})(\cdot)^T|\mathbf{z}^k] = \int (\mathbf{f}_{k+\ell}(\mathbf{x}_{k+\ell}) - \hat{\mathbf{x}}_{k+m|k})(\cdot)^T \times \mathcal{N}\{\mathbf{x}_{k+\ell}; \hat{\mathbf{x}}_{k+\ell|k}, \mathbf{P}^{xx}_{k+\ell|k}\} d\mathbf{x}_{k+\ell} + \mathbf{Q}_{k+\ell}, \tag{12}$$

where the notation $(\mathbf{a})(\cdot)^T$ means $(\mathbf{a})(\mathbf{a})^T$. It is clear that multi-step prediction can be deemed as recursive application of the one-step prediction.

C. Gaussian Filter Design

Calculation of the filtering PDF requires recursive evaluation of the Bayes' rule (7) and one-step predictive Chapman-Kolmogorov equation (6). Assuming (9), their solution in a GE framework leads to Algorithm I:

Algorithm I: Generic Local Filter

Step 1: Set the time instant $k = 0$ and define an initial condition $p(\mathbf{x}_0|\mathbf{z}^0) = \mathcal{N}\{\mathbf{x}_0; \hat{\mathbf{x}}_{0|-1}, \mathbf{P}^{xx}_{0|-1}\}$.

Step 2: The moments of the filtering estimate $p(\mathbf{x}_k|\mathbf{z}^k) \triangleq \mathcal{N}\{\mathbf{x}_k; \hat{\mathbf{x}}_{k|k}, \mathbf{P}^{xx}_{k|k}\}$ are

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1}) \ , \tag{13}$$

$$\mathbf{P}^{xx}_{k|k} = \mathbf{P}^{xx}_{k|k-1} - \mathbf{K}_k \mathbf{P}^{zz}_{k|k-1} \mathbf{K}_k^T \ , \tag{14}$$

[2]This assumption is not always realistic especially for strongly nonlinear systems. However, the Gaussian PDF is fully defined by the first two moments, which allows efficient closed-form solution to the FRRs.

[3]All GEs are linear algorithms with respect to the actual measurement and have the same structure as the KF and the RTSS.

where $\mathbf{K}_k = \mathbf{P}^{xz}_{k|k-1}(\mathbf{P}^{zz}_{k|k-1})^{-1}$ is the filter gain,

$$\hat{\mathbf{z}}_{k|k-1} = \int \mathbf{h}_k(\mathbf{x}_k)\mathcal{N}\{\mathbf{x}_k; \hat{\mathbf{x}}_{k|k-1}, \mathbf{P}^{xx}_{k|k-1}\}d\mathbf{x}_k, \quad (15)$$

$$\mathbf{P}^{zz}_{k|k-1} = \int (\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{z}}_{k|k-1})(\cdot)^T \times \mathcal{N}\{\mathbf{x}_k; \hat{\mathbf{x}}_{k|k-1}, \mathbf{P}^{xx}_{k|k-1}\}d\mathbf{x}_k + \mathbf{R}_k, \quad (16)$$

$$\mathbf{P}^{xz}_{k|k-1} = \int (\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1})(\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{z}}_{k|k-1})^T \times \mathcal{N}\{\mathbf{x}_k; \hat{\mathbf{x}}_{k|k-1}, \mathbf{P}^{xx}_{k|k-1}\}d\mathbf{x}_k. \quad (17)$$

Step 3: The predictive moments of the Gaussian-assumed PDF $p(\mathbf{x}_{k+1}|\mathbf{z}^k) \triangleq \mathcal{N}\{\mathbf{x}_k; \hat{\mathbf{x}}_{k+1|k}, \mathbf{P}^{xx}_{k+1|k}\}$ are calculated according to (11), (12) with $m = 1$.
The algorithm then continues to **Step 2**.

D. Gaussian Smoother Design

The smoothing operation, i.e., solution to (8), can be generally divided into three categories:

- *Fixed-point smoothing*, when time instant of the state to be estimated $k - m$ is fixed,
- *Fixed-lag smoothing*, when lag m is fixed, and
- *Fixed-interval smoothing*, when time instant of the last available measurement k is fixed.

Although each of these categories has its own dedicated smoothing algorithms, we will focus on the Rauch-Tung-Striebel-type smoother in this paper. This smoother can be easily adapted for all the categories [26] and has been used for the vast majority of types of Gaussian filters, e.g., extended or unscented RTSS [19], [25], [26].

The RTSS in the GE framework is given by the following relations for calculating the moments of the state

$$p(\mathbf{x}_{k-m}|\mathbf{z}^k) \triangleq \mathcal{N}\{\mathbf{x}_{k-m}; \hat{\mathbf{x}}_{k-m|k}, \mathbf{P}^{xx}_{k-m|k}\} \quad (18)$$

as

$$\hat{\mathbf{x}}_{\mathtt{m}|k} = \hat{\mathbf{x}}_{\mathtt{m}|\mathtt{m}} + \mathbf{L}_{\mathtt{m}}(\hat{\mathbf{x}}_{\mathtt{m}+1|k} - \hat{\mathbf{x}}_{\mathtt{m}+1|\mathtt{m}}), \quad (19)$$

$$\mathbf{P}^{xx}_{\mathtt{m}|k} = \mathbf{P}^{xx}_{\mathtt{m}|\mathtt{m}} - \mathbf{L}_{\mathtt{m}}(\mathbf{P}^{xx}_{\mathtt{m}+1|\mathtt{m}} - \mathbf{P}^{xx}_{\mathtt{m}+1|k})\mathbf{L}^T_{\mathtt{m}}, \quad (20)$$

$$\mathbf{L}_{\mathtt{m}} = \mathbf{P}^{xx}_{\mathtt{m},\mathtt{m}+1|\mathtt{m}}(\mathbf{P}^{xx}_{\mathtt{m}+1|\mathtt{m}})^{-1}, \quad (21)$$

where the lag $\mathtt{m} = k - m$ and

$$\mathbf{P}^{xx}_{\mathtt{m},\mathtt{m}+1|\mathtt{m}} = \mathsf{E}[(\mathbf{x}_{\mathtt{m}} - \hat{\mathbf{x}}_{\mathtt{m}|\mathtt{m}})(\mathbf{x}_{\mathtt{m}+1} - \hat{\mathbf{x}}_{\mathtt{m}+1|\mathtt{m}})^T|\mathbf{z}^{\mathtt{m}}]. \quad (22)$$

The RTSS is a backward recursion that processes the filtering and predictive estimates calculated by the filter and predictor in their forward run.

III. Stochastic Integration Rule and Estimators

Relations (11)–(22) represent a common framework for all the GE. The particular estimators differ in the calculation of the expected values

- (11), (12) for prediction,
- (15)–(17) for filtering, and
- (22) for smoothing.

Calculation of the required moments lies in an evaluation of the *Gaussian-weighted* integral

$$\mathcal{I} = \mathsf{E}[\mathbf{g}(\mathbf{x}_q)|\mathbf{z}^l] = \int \mathbf{g}(\mathbf{x}_q)\mathcal{N}\{\mathbf{x}_q; \hat{\mathbf{x}}_{q|l}, \mathbf{P}^{xx}_{q|l}\}d\mathbf{x}_q, \quad (23)$$

where the conditional mean $\hat{\mathbf{x}}_{q|l}$ and the covariance matrix $\mathbf{P}^{xx}_{q|l}$ are known from estimator preceding steps and the nonlinear function $\mathbf{g}(\cdot)$ stems from the model and calculated moment definition. The moment calculation can be, thus, seen as a calculation of the moments of a nonlinearly transformed Gaussian random variable with known description [17], [26]. The particular form of $\mathbf{g}(\cdot)$, dimension of $\mathcal{I}$ denoted as $n_{\mathcal{I}}$, and the time indices q, l depend on the selected estimator and are detailed later.

A. Stochastic Integration Rule

As any other integration rule (IR), the stochastic integration rule (SIR) [8], [10], [12] numerically evaluates (23) using

$$\hat{\mathcal{I}} = \sum_{i=1}^{S} \omega^{(i)}\mathbf{g}(\boldsymbol{\xi}^{(i)}_{k|l}), \quad (24)$$

where $\boldsymbol{\xi}^{(i)}_{q|l}$ and $\omega^{(i)}$ are spherical-radial points and corresponding weights of the rule, respectively. The SIR combines a deterministic IR and Monte-Carlo (MC) IR into a single algorithm, where some points are selected randomly, and the remaining are computed using deterministic rules. The SIR takes the best properties of both, namely [10]

- Integration error going to zero with increasing S, i.e.
$$\varepsilon = (\mathcal{I} - \hat{\mathcal{I}}) \to \mathbf{0} \text{ as } S \to \infty, \quad (25)$$
and $\varepsilon = 0$ if $\mathbf{g}(\cdot)$ is degree-d polynomial, where d is degree of used spherical and radial rule,
- Faster convergence rate than MC-based IRs,
- Implicit assessment of integral (24) evaluation error.

Full derivation of the SIR of degrees one, three, and five can be found in [10]. In the algorithm below we briefly introduce 3rd-order SIR, which offers a reasonable compromise between accuracy and computational complexity.

Algorithm II: Degree 3 Stochastic Integration Rule

Step 1: Select a maximum number of iterations $N_{\max}$ or an error tolerance $\varepsilon_{\min}$.
Step 2: Set the current iteration number $N = 0$, initial value of the integral $\hat{\mathcal{I}}_N = \mathbf{0}_{n_{\mathcal{I}}\times 1}$, and the initial square error of the integral $\Sigma_N = \mathbf{0}_{n_{\mathcal{I}}\times n_{\mathcal{I}}}$, and define the central IR point $\boldsymbol{\xi}^{(0)}_{q|l} = \hat{\mathbf{x}}_{k|l}$, which is constant over all SIR iterations.
Step 3: Repeat (until $N = N_{max}$ or $\mathrm{tr}(\Sigma_N) < \varepsilon_{\min}$, but at least once) the following loop:
a) Set $N = N + 1$.
b) Generate a uniformly random orthogonal matrix $\mathbf{C}^{(N)}$ of dimension $n_x \times n_x$ [4] and generate a random number

ρ from the *Chi* distribution with $(n_x + 2)$ degrees of freedom, i.e., $\rho^{(N)} \sim \mathrm{Chi}(n_x + 2)$.

c) Compute a set of IR points and weights according to

$$\boldsymbol{\xi}_{q|l}^{(N,i)} = -\rho^{(N)} \mathbf{S}_{q|l}^{xx} \mathbf{C}^{(N)} \mathbf{e}_i + \hat{\mathbf{x}}_{q|l}, \tag{26}$$

$$\boldsymbol{\xi}_{q|l}^{(N,n_x+i)} = \rho^{(N)} \mathbf{S}_{q|l}^{xx} \mathbf{C}^{(N)} \mathbf{e}_i + \hat{\mathbf{x}}_{q|l}, \tag{27}$$

$$\omega_{q|l}^{(N,0)} = 1 - \tfrac{n_x}{\rho^2},\ \omega_{q|l}^{(N,i)} = \omega_{q|l}^{(N,n_x+i)} = \tfrac{1}{2\rho^2}, \tag{28}$$

where $i = 1, 2, \ldots, n_x$, $\mathbf{e}_i$ is the i-th column of the identity matrix $\mathbf{I}_{n_x}$, and $\mathbf{S}_{q|l}^{xx}$ is the square-root of matrix $\mathbf{P}_{q|l}^{xx}$ so that $\mathbf{P}_{q|l}^{xx} = \mathbf{S}_{q|l}^{xx} \left(\mathbf{S}_{q|l}^{xx}\right)^T$. Matrix $\mathbf{C}^{(N)}$ rotates the IR points and $\rho^{(N)}$ scales them.

d) Compute approximation of the integral at current iteration $\mathfrak{I}^{(N)}$ (SR denotes a spherical-radial SIR), the update of the integral value $\hat{\mathcal{I}}_N$, and the corresponding error Σ_N

$$\mathfrak{I}^{(N)} = \sum_{i=0}^{2n_x} \mathbf{g}(\boldsymbol{\xi}_{q|l}^{(N,i)}) \omega_{q|l}^{(N,i)}, \tag{29}$$

$$\hat{\mathcal{I}}_N = \hat{\mathcal{I}}_{N-1} + (\mathfrak{I}^{(N)} - \hat{\mathcal{I}}_{N-1})/N, \tag{30}$$

$$\Sigma_N = (N-2)\Sigma_{N-1}/N + (\mathfrak{I}^{(N)} - \hat{\mathcal{I}}_{N-1})(\cdot)^T/N^2. \tag{31}$$

Step 4: Once a stopping condition is fulfilled, the calculated approximate value of the integral $\mathcal{I}$ (23) is $\hat{\mathcal{I}} = \hat{\mathcal{I}}_N$ and total number of the IR points is $S = N(2n_x + 1)$.

Note that the *cubature* IR [3], [18], widely used in the filter design, is a special case of the introduced SIR.

B. SIR-based Prediction, Filtering, and Smoothing

Design of the SIR-based predictor, filter, and predictor lies in calculation of the state and measurement moments in the GE framework using the SIR given in Algorithm II.

1) Prediction: The predictive state mean $\hat{\mathbf{x}}_{k+m|k}$ (11) is calculated using the SIR (24) detailed in Algorithm II, where $\hat{\mathbf{x}}_{q|l} = \hat{\mathbf{x}}_{k+\ell|k}$, $\mathbf{P}_{q|l}^{xx} = \mathbf{P}_{k+\ell|k}^{xx}$, and $\mathbf{g}(\mathbf{x}_q) = \mathbf{f}_{k+\ell}(\mathbf{x}_{k+\ell})$ with $\ell = m - 1$. The predictive covariance matrix is calculated analogously with the only change in the definition of the nonlinear function, which is $\mathbf{g}(\mathbf{x}_q) = \left(\mathbf{f}_{k+\ell}(\mathbf{x}_{k+\ell}) - \hat{\mathbf{x}}_{k+m|k}\right)\left(\mathbf{f}_{k+\ell}(\mathbf{x}_{k+\ell}) - \hat{\mathbf{x}}_{k+m|k}\right)^T$ in this case.

2) Filtering: The predictive measurement mean $\hat{\mathbf{z}}_{k|k-1}$ (16), covariance matrix $\mathbf{P}_{k|k-1}^{zz}$ (16), and "cross-covariance" matrix $\mathbf{P}_{k|k-1}^{xz}$ (17) are calculated using the SIR (24), where $\hat{\mathbf{x}}_{q|l} = \hat{\mathbf{x}}_{k|k-1}$, $\mathbf{P}_{q|l}^{xx} = \mathbf{P}_{k|k-1}^{xx}$, and the SIR-related nonlinear function is defined $\mathbf{g}(\mathbf{x}_q) = \mathbf{h}_k(\mathbf{x}_k)$, $\mathbf{g}(\mathbf{x}_q) = \left(\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{z}}_{k|k-1}\right)\left(\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{z}}_{k|k-1}\right)^T$, and $\mathbf{g}(\mathbf{x}_q) = \left(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1}\right)\left(\mathbf{h}_k(\mathbf{x}_k) - \hat{\mathbf{z}}_{k|k-1}\right)^T$.

3) Smoothing: The cross-covariance matrix $\mathbf{P}_{\mathtt{m}+1,\mathtt{m}|\mathtt{m}}^{xx}$ (22) for smoother gain is calculated using the SIR (24), where $\hat{\mathbf{x}}_{q|l} = \hat{\mathbf{x}}_{\mathtt{m}|\mathtt{m}}$, $\mathbf{P}_{q|l}^{xx} = \mathbf{P}_{\mathtt{m}|\mathtt{m}}^{xx}$, and the SIR-related nonlinear function is $\mathbf{g}(\mathbf{x}_q) = \left(\mathbf{x}_{\mathtt{m}} - \hat{\mathbf{x}}_{\mathtt{m}|\mathtt{m}}\right)\left(\mathbf{f}_{\mathtt{m}}(\mathbf{x}_{\mathtt{m}}) - \hat{\mathbf{x}}_{\mathtt{m}+1|\mathtt{m}}\right)^T$, respectively.

IV. Square-Root Estimators

Numerical stability has been a focal point of Gaussian estimators development for the past several decades [2], [20], [26]. The emphasis has been placed on calculations that ensure symmetric and positive-definite conditional covariance matrix. Generally, two types of numerically robust algorithms have been developed

- Filtering and smoothing covariance matrices (14), (20) calculation, where subtraction of two covariance matrices is omitted using the Joseph form [20],
- Square-root implementation, where instead of the conditional covariance matrix, its factor is directly calculated.

The square-root implementation might lead to computational complexity reduction, as the covariance matrix factor $\mathbf{S}_{q|l}^{xx}$ required by the SIR in IR points calculation (26) is directly calculated and thus available without the need of any decomposition.

In this section, we, therefore, propose the square-root version of the SIR-based estimators using *direct calculation* of the state estimate covariance matrix factor for the prediction, filtering, and smoothing. Derivation of the square-root SIR-based estimators adopts the concept of the square-root unscented Kalman estimators developed in [24], [25].

A. SIR Reformulation

The SIR points and weights $\{\boldsymbol{\xi}_{q|l}^{(n,i)}, \omega_{q|l}^{(n,i)}\}_{i=0}^{2n_x}$ with $n = 1, \ldots, N$, generated in N iterations, can be concatenated into the following set of points and corresponding weights

$$\boldsymbol{\Xi}_{q|l} = \{\boldsymbol{\xi}_{q|l}^{(0)}, \boldsymbol{\xi}_{q|l}^{(1,1)}, \ldots, \boldsymbol{\xi}_{q|l}^{(1,2n_x)}, \ldots, \boldsymbol{\xi}_{q|l}^{(N,1)}, \ldots, \boldsymbol{\xi}_{q|l}^{(N,2n_x)}\}, \tag{32}$$

$$\boldsymbol{\Omega}_{q|l} = \{\tilde{\omega}_{q|l}^{(0)}, \tilde{\omega}_{q|l}^{(1,1)}, \ldots, \tilde{\omega}_{q|l}^{(1,2n_x)}, \ldots, \tilde{\omega}_{q|l}^{(N,1)}, \ldots, \tilde{\omega}_{q|l}^{(N,2n_x)}\}, \tag{33}$$

where $\tilde{\omega}_{q|l}^{(n,i)} = {}^{\omega_{q|l}^{(n,i)}}/_N$ and $\tilde{\omega}_{q|l}^{(0)} = \frac{1}{N}\sum_{n=1}^{N} \omega_{q|l}^{(n,0)}$. Number of all SIR points is $n_S = 2n_x N + 1$, The proof is outlined in Appendix A.

For the sake of notional simplicity, we further denote i-th elements of sets $\boldsymbol{\Xi}_{k+\ell|k}, \boldsymbol{\Omega}_{k+\ell|k}$ (32), (33) as $\boldsymbol{\Xi}_{k+\ell|k}^{(i)}, \boldsymbol{\Omega}_{k+\ell|k}^{(i)}$, respectively, where $i = 0, 1, \ldots, n_S$.

B. Predictive Covariance Matrix Factor

SIR-based predictive covariance $\mathbf{P}_{k+m|k}^{xx}$, stemming from (12) and Section III.B.1, reads

$$\mathbf{P}_{k+m|k}^{xx} = \sum_{i=1}^{n_S} \boldsymbol{\Omega}_{k+\ell|k}^{(i)} (\mathbf{f}_{k+\ell}(\boldsymbol{\Xi}_{k+\ell|k}^{(i)}) - \hat{\mathbf{x}}_{k+m|k})(\cdot)^T + \mathbf{Q}_{k+\ell}. \tag{34}$$

A factor of the covariance matrix $\mathbf{P}_{k+m|k}^{xx}$ (34) is simply

$$\sqrt{\mathbf{P}_{k+m|k}^{xx}} = [\tilde{\mathbf{S}}_{k+m|k}^{xx},\ \mathbf{S}_{k+\ell}^{Q}], \tag{35}$$

where

$$\tilde{\mathbf{S}}_{k+m|k}^{xx} = \Big[\sqrt{\boldsymbol{\Omega}_{k+\ell|k}^{(1)}} \left(\mathbf{f}_{k+\ell}(\boldsymbol{\Xi}_{k+\ell|k}^{(i)}) - \hat{\mathbf{x}}_{k+m|k}\right), \ldots, \sqrt{\boldsymbol{\Omega}_{k+\ell|k}^{(n_S)}} \left(\mathbf{f}_{k+\ell}(\boldsymbol{\Xi}_{k+\ell|k}^{(i)}) - \hat{\mathbf{x}}_{k+m|k}\right)\Big] \in \mathbb{R}^{n_x \times n_S} \tag{36}$$

and $\mathbf{S}^Q_{k+\ell}$ is a factor of the state noise covariance matrix s.t. $\mathbf{Q}_{k+\ell} = \mathbf{S}^Q_{k+\ell}(\mathbf{S}^Q_{k+\ell})^T$. Unfortunately, this factor is a *rectangular* matrix unsuitable for further calculations. To make the matrix *square*, we can apply the Householder triangularisation [17], [24] denoted as $\mathrm{ht}(\cdot)$, i.e.,

$$\mathbf{S}^{xx}_{k+m|k} = \mathrm{ht}(\tilde{\mathbf{S}}^{xx}_{k+m|k}) \in \mathbb{R}^{n_x \times n_x}, \tag{37}$$

with $\mathbf{P}^{xx}_{k+m|k} = \tilde{\mathbf{S}}^{xx}_{k+m|k}(\tilde{\mathbf{S}}^{xx}_{k+m|k})^T = \mathbf{S}^{xx}_{k+m|k}(\mathbf{S}^{xx}_{k+m|k})^T$. Optionally, other decomposition, such as the QR decomposition, can be used as well for square matrix design.

C. Filtering Covariance Matrix Factor

Factorisation of the filtering covariance matrix $\mathbf{P}^{xx}_{k|k}$ is more involved due to subtraction of two terms on right-hand side of (14). Using a few matrix manipulations given in Appendix B, we can show that the filtering covariance matrix (14) can be written as [24]

$$\mathbf{P}^{xx}_{k|k} = \left[\tilde{\mathbf{S}}^{xx}_{k|k-1} - \mathbf{K}_k\tilde{\mathbf{S}}^{zz}_{k|k-1},\ \mathbf{K}_k\mathbf{S}^R_k\right][\cdot]^T, \tag{38}$$

where $\mathbf{S}^R_k$ is a factor of the measurement noise covariance matrix s.t. $\mathbf{R}_k = \mathbf{S}^R_k(\mathbf{S}^R_k)^T$ and IR points stacking matrices are defined as

$$\tilde{\mathbf{S}}^{xx}_{k|k-1} = \left[\sqrt{\boldsymbol{\Omega}^{(1)}_{k|k-1}}\left(\boldsymbol{\Xi}^{(1)}_{k|k-1} - \hat{\mathbf{x}}_{k|k-1}\right), \ldots, \sqrt{\boldsymbol{\Omega}^{(n_S)}_{k|k-1}}\left(\boldsymbol{\Xi}^{(n_S)}_{k|k-1} - \hat{\mathbf{x}}_{k|k-1}\right)\right]. \tag{39}$$

$$\tilde{\mathbf{S}}^{zz}_{k|k-1} = \left[\sqrt{\boldsymbol{\Omega}^{(1)}_{k|k-1}}\left(\mathbf{h}_k(\boldsymbol{\Xi}^{(1)}_{k|k-1}) - \hat{\mathbf{z}}_{k|k-1}\right), \ldots, \sqrt{\boldsymbol{\Omega}^{(n_S)}_{k|k-1}}\left(\mathbf{h}_k(\boldsymbol{\Xi}^{(n_S)}_{k|k-1}) - \hat{\mathbf{z}}_{k|k-1}\right)\right]. \tag{40}$$

Then, the matrix square factor can directly be determined as

$$\mathbf{S}^{xx}_{k|k} = \mathrm{ht}\left(\left[\tilde{\mathbf{S}}^{xx}_{k|k-1} - \mathbf{K}_k\tilde{\mathbf{S}}^{zz}_{k|k-1},\ \mathbf{K}_k\mathbf{S}^R_k\right]\right). \tag{41}$$

In the square-root version, the Kalman gain reads

$$\mathbf{K}_k = \mathbf{P}^{xz}_{k|k-1}([\tilde{\mathbf{S}}^{zz}_{k|k-1},\ \mathbf{S}^R_k][\cdot]^T)^{-1}, \tag{42}$$

where $\mathbf{P}^{xz}_{k|k-1} = \tilde{\mathbf{S}}^{xx}_{k|k-1}(\tilde{\mathbf{S}}^{zz}_{k|k-1})^T$.

D. Smoothing Covariance Matrix Factor

Similarly to the filtering matrix, smoothing covariance matrix $\mathbf{P}^{xx}_{\mathtt{m}|k}$ (20) can be written in the form

$$\mathbf{P}^{xx}_{\mathtt{m}|k} = \left[\tilde{\mathbf{S}}^{xx}_{\mathtt{m}|\mathtt{m}} - \mathbf{L}_\mathtt{m}\tilde{\mathbf{S}}^{xx}_{\mathtt{m}+1|\mathtt{m}},\ \mathbf{L}_\mathtt{m}\mathbf{S}^Q_\mathtt{m},\ \mathbf{L}_\mathtt{m}\mathbf{S}^{xx}_{\mathtt{m}+1|k}\right][\cdot]^T, \tag{43}$$

where IR points stacking matrices are defined as

$$\tilde{\mathbf{S}}^{xx}_{\mathtt{m}|\mathtt{m}} = \left[\sqrt{\boldsymbol{\Omega}^{(1)}_{\mathtt{m}|\mathtt{m}}}\left(\boldsymbol{\Xi}^{(1)}_{\mathtt{m}|\mathtt{m}} - \hat{\mathbf{x}}_{\mathtt{m}|\mathtt{m}}\right), \ldots, \sqrt{\boldsymbol{\Omega}^{(n_S)}_{\mathtt{m}|\mathtt{m}}}\left(\boldsymbol{\Xi}^{(n_S)}_{\mathtt{m}|\mathtt{m}} - \hat{\mathbf{x}}_{\mathtt{m}|\mathtt{m}}\right)\right]. \tag{44}$$

$$\tilde{\mathbf{S}}^{xx}_{\mathtt{m}+1|\mathtt{m}} = \left[\sqrt{\boldsymbol{\Omega}^{(1)}_{\mathtt{m}+1|\mathtt{m}}}\left(\mathbf{f}_\mathtt{m}(\boldsymbol{\Xi}^{(1)}_{\mathtt{m}+1|\mathtt{m}}) - \hat{\mathbf{x}}_{\mathtt{m}+1|\mathtt{m}}\right), \ldots, \sqrt{\boldsymbol{\Omega}^{(n_S)}_{\mathtt{m}+1|\mathtt{m}}}\left(\mathbf{f}_\mathtt{m}(\boldsymbol{\Xi}^{(n_S)}_{\mathtt{m}+1|k\mathtt{m}}) - \hat{\mathbf{x}}_{\mathtt{m}+1|\mathtt{m}}\right)\right]. \tag{45}$$

Then, the the square factor of the covariance matrix (43) simply reads

$$\mathbf{S}^{xx}_{\mathtt{m}|k} = \mathrm{ht}\left(\left[\tilde{\mathbf{S}}^{xx}_{\mathtt{m}|\mathtt{m}} - \mathbf{L}_\mathtt{m}\tilde{\mathbf{S}}^{xx}_{\mathtt{m}+1|\mathtt{m}},\ \mathbf{L}_\mathtt{m}\mathbf{S}^Q_\mathtt{m},\ \mathbf{L}_\mathtt{m}\mathbf{S}^{xx}_{\mathtt{m}+1|k}\right]\right). \tag{46}$$

Further details on derivation of (46) are given in Appendix B.

V. SIR-based Estimator Enhancement

The above introduced SIR and SIR-based estimators can be seen as *baseline* algorithms, which can be improved to be less computationally demanding and more computationally robust. These improvements are discussed in this section.

A. Efficient Calculation

In the basic SIR-based algorithms, each moment is calculated using dedicated SIR (Algorithm II). This sequential calculation brings a computational overhead especially in the *filtering* step, where three different measurement-related moments (15)–(16) are calculated. The computational overhead is caused by the necessity to generate the random orthogonal matrix $\mathbf{C}$ and the IR points $\{\boldsymbol{\xi}^{(i)}\}_{i=0}^{2n_x}$ at each iteration. To *reduce* the computational complexity, all the moments can be calculated using evaluation of the single SIR, i.e., using the same IR points.

Following this idea, we can calculate the measurement prediction *raw* moments the SIR, i.e.,

$$\hat{\mathbf{z}}_{k|k-1} = \int \mathbf{h}_k(\mathbf{x}_k)\mathcal{N}\{\mathbf{x}_k; \hat{\mathbf{x}}_{k|k-1}, \mathbf{P}^{xx}_{k|k-1}\}d\mathbf{x}_k, \tag{47}$$

$$\mathbf{M}^{zz}_{k|k-1} = \int \mathbf{h}_k(\mathbf{x}_k)(\cdot)^T\mathcal{N}\{\mathbf{x}_k; \hat{\mathbf{x}}_{k|k-1}, \mathbf{P}^{xx}_{k|k-1}\}d\mathbf{x}_k, \tag{48}$$

$$\mathbf{M}^{xz}_{k|k-1} = \int \mathbf{x}_k(\mathbf{h}_k(\mathbf{x}_k))^T\mathcal{N}\{\mathbf{x}_k; \hat{\mathbf{x}}_{k|k-1}, \mathbf{P}^{xx}_{k|k-1}\}d\mathbf{x}_k, \tag{49}$$

with one run of the SIR, i.e., using the same weighted IR points, and then, the required *central* moments are

$$\mathbf{P}^{zz}_{k|k-1} = \mathbf{M}^{zz}_{k|k-1} - \hat{\mathbf{z}}_{k|k-1}\hat{\mathbf{z}}^T_{k|k-1} + \mathbf{R}_k, \tag{50}$$

$$\mathbf{P}^{xz}_{k|k-1} = \mathbf{M}^{xz}_{k|k-1} - \hat{\mathbf{x}}_{k|k-1}\hat{\mathbf{z}}^T_{k|k-1}. \tag{51}$$

Similarly, the state prediction (11), (12) can be implemented in this efficient way, where the cross-covariance matrix $\mathbf{P}^{xx}_{\mathtt{m},\mathtt{m}+1|\mathtt{m}}$ (22) can be efficiently calculated as well. This covariance matrix is redundant for the prediction step, but it can be readily used in the smoothing (if smoothing is intended).

If the *square-root* implementation is concerned, analogous modification of the moment calculation can be easily done. Moreover, the filtering and predictive IR points needed for construction of the matrices $\tilde{\mathbf{S}}^{xx}_{\mathtt{m}|\mathtt{m}}, \tilde{\mathbf{S}}^{xx}_{\mathtt{m}+1|\mathtt{m}}$ (44), (45) can be stored during the forward run of the SIF to reduce the smoother computational complexity, but at the expense of the higher memory requirements (depending on the smoothing horizon $\mathtt{m}$).

B. Integration Error

SIR implicitly provides information about accuracy, i.e., about the error, of the integral solution. This information is accumulated in Σ_N (31). The error assessment can be used either for

- Stopping condition of the SIR, which ensures that the calculated moments meet the required accuracy defined by the threshold $\varepsilon_{\min}$ (indicated in Step 3 of Algorithm II),
- Increase of the state estimate covariance matrix, which then takes into account the SIR output error.

The latter option increases the estimate covariance matrix with the mean error assessment. To illustrate this concept in the measurement prediction (47)–(51), we can calculate the measurement prediction $\hat{\mathbf{z}}_{k|k-1}$ (47) and the respective error assessment $\Sigma_N^{\hat{\mathbf{z}}_{k|k-1}}$ using the SIR (Algorithm 2) then the predictive covariance matrix can be calculated as

$$\mathbf{P}^{zz}_{k|k-1} = \mathbf{M}^{zz}_{k|k-1} - \hat{\mathbf{z}}_{k|k-1}\hat{\mathbf{z}}^T_{k|k-1} + \Sigma_N^{\hat{\mathbf{z}}_{k|k-1}} + \mathbf{R}_k \quad (52)$$

instead of (50). Utilisation of error assessment of the covariance matrices calculation, i.e., of $\Sigma_N^{\mathbf{P}^{zz}_{k|k-1}}$ and $\Sigma_N^{\mathbf{P}^{xz}_{k|k-1}}$ is currently under investigation.

C. Integration Rule Order

We have introduced the SIR of the 3-rd order. However, the SIR can be designed for any odd degree (typically, first and fifth), which offers the trade-off between the computational complexity and integration error [10], [12].

VI. Stone Soup and Implementation

The Stone Soup project project is an open-source tracking and estimation framework currently available as a python library; to support developers, users, and systems engineers. The framework saw its first public alpha release in 2017 [23] with the first beta release in 2019. The name of the framework is inspired by the tale of the "Stone Soup"; from European folklore in which many ingredients (estimation methods) are contributed by villagers (researchers) to devise a flavourful soup(useful software). Inherent in the common repository is the ability to compare and reuse contributions to extend the quality of the estimation capabilities. While some have sought initial ideas [6], [14], Barr et al. [5] highlight the many capabilities such as track generation, filtering, classification, data association, and sensor management routines. Examples are provided for video tracking, automatic dependent surveillance – broadcast (ADS-B) tracking, orbital estimation, drone tracking, sensor management, and tracking evaluation. Recently, a series of papers began using the techniques, specifically for radar-based tracking. Carniglia et al. [7] used the Stone Soup Generalized Optimal Sub-Pattern Assignment (GOSPA) and Single Integrated Air Picture (SIAP) metrics to compare the Joint-Probabilistic Data Association (JPDA) to track multiple airborne targets being in clutter from two ground-based radars to assess the track quality sensitivity of biased sensor measurements. Similarly, tracking of the

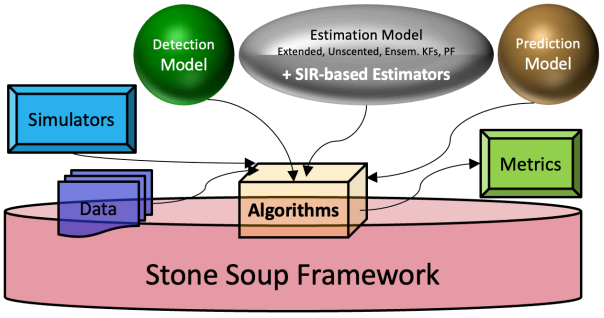

Figure 1: Elements of the ISIF Stone Soup Framework with SIR-based estimators [6].

adaptive Kernel Kalman Filter (AKKF) was compared to the particle filter (PF) using Stone Soup infrastructure [27].

Given the recent interest in artificial intelligence techniques, Stone Soup was used to compare neural net (NN) trackers for drone surveillance using the recurrent NN (RNN), convolutional NN (CNN), and machine learning perceptron (MLP) networks [13] as well as drone tracking with reinforcement learning. The current techniques of Stone Soup are summarised in [1] with models (measurement, transition), filters (extended or unscented KF, PF), state update methods (information, Gromov Flow and PHD), and simulation capabilities (metrics, scenario generators, and track stitching). Among the many capabilities of Stone Soup, it provides a useful infrastructure for novel tracking and estimation method comparisons.

In the spirit of collaboration, our group is providing the SIF and an associated smoother and predictor for inclusion in the framework. Integration in Stone Soup also allows for more easy comparisons between the SIR-based estimators and other estimation methods. Of particular interest here, is the relationship between the unscented Kalman filter (UKF) and the SIF, as the two algorithms are the most similar algorithms in the framework. In fact, the UKF can be seen as a special case of the SIF, when the IR (or sigma) points are suitably generated [10]. The overall scheme of the Stone Soup framework with the SIR-based estimators is illustrated in Figure 1.

A. Stochastic Integration Filter, Smoother, and Predictor Implementation: Current Status

Currently, we have implemented the stochastic integration rule of the 1st, 3rd, and 5th order for Gaussian weighted integral solution and applied it in the SIF predictor, SIF updater, and SIF smoother, introduced and derived in this paper. These algorithms are submitted to be included in the Stone Soup repository. The square-root versions are in development and will be submitted shortly after the publication of this paper.

VII. Numerical Illustration

Performance of the SIF is evaluated using the tracking scenario used for illustration of the extended KF (EKF) and

		EKF	UKF	**SIF**	SIF Improvement
RMSE	x_1	0.7774	0.8344	**0.7398**	5%
	x_2	0.3867	0.3987	**0.3881**	
	x_3	0.6884	0.7343	**0.6781**	
	x_4	0.3758	0.3808	**0.3732**	
ANEES		10.9559	5.3510	**4.0810**	40%

Table I: EKF, UKF, and SIF performance summary.

the UKF [2], [11], [15], [20] in Stone Soup[4]. In particular, the following state-space model is considered

$$\mathbf{x}_{k+1} = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_k + \mathbf{w}_k, \tag{53}$$

$$\mathbf{z}_k = \begin{bmatrix} \arctan \frac{x_{3,k}-r_y}{x_{1,k}-r_x} \\ \sqrt{(x_{1,k}-r_x)^2+(x_{3,k}-r_y)^2} \end{bmatrix} + \mathbf{v}_k, \tag{54}$$

where the state is represented by an sought object position and velocity in x and y directions and the measurement is the bearing angle[5] and range of the object from a radar located at $\mathbf{r} = [r_x, r_y]^T = [50, 0]^T$. Sampling period is $T = 1$ and $k = 0, 1, \ldots, 20$. The state and measurement noises covariance matrices (3), (4) are

$$\mathbf{Q}_k = \begin{bmatrix} q_1\frac{T^3}{3} & q_1\frac{T^2}{2} & 0 & 0 \\ q_1\frac{T^2}{2} & q_1 T & 0 & 0 \\ 0 & 0 & q_2\frac{T^3}{3} & q_2\frac{T^2}{2} \\ 0 & 0 & q_2\frac{T^2}{2} & q_2 T \end{bmatrix}, \mathbf{R}_k = \begin{bmatrix} 0.2\pi/180 & 0 \\ 0 & 1 \end{bmatrix} \tag{55}$$

with $q_1 = q_2 = 0.05$. The initial state (5) description is

$$\bar{\mathbf{x}}_0 = \begin{bmatrix} 50 \\ 1 \\ 1 \\ 1 \end{bmatrix}, \mathbf{P}_0 = \begin{bmatrix} 1.5 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 \\ 0 & 0 & 1.5 & 0 \\ 0 & 0 & 0 & 0.5 \end{bmatrix}. \tag{56}$$

The estimation performance is assessed using

- The root-mean-square error (RMSE) of the i-th state

$$\mathrm{RMSE}_i = \sqrt{\frac{1}{T}\sum_{k=0}^{T}(x_{i,k} - \hat{x}_{i,k|k})^2}, \tag{57}$$

- The averaged normalised estimation error squared (ANEES)

$$\mathrm{ANEES} = \frac{1}{T}\sum_{k=0}^{T}(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})^T (\mathbf{P}^{xx}_{k|k})^{-1}(\cdot), \tag{58}$$

calculated over 10^4 Monte-Carlo (MC) simulations, where $\hat{x}_{i,k|k}$ is the filtering estimate of the i-th element of the state $x_{i,k}$. The UKF was designed with the default scaling parameters of the Stone Soup implementation, i.e., with $\alpha = 0.5, \beta = 2, \kappa = 3 - n_x = -1$ and the SIF with maximum number of iterations $N_{\max} = 10$.

Excellent performance of the SIF was demonstrated in various scenarios and under different settings [9], [10], [16], [22], [29]. This is also confirmed using the Stone Soup defined tracking scenario (53)–(56) run using the Python and MATLAB[6]® implementations and evaluated using the ANEES and RMSE metrics. The results are shown in Table I, where the absolute values and an *average* improvement of the SIF w.r.t. EKF and UKF can be found. In terms of the RMSE all the considered filters provide very similar performance[7], however, as indicated by the ANEES, the UKF and *especially* the SIF provide more consistent estimates (i.e., more realistic estimate covariance matrix).

Regarding the computational complexity, the SIF is more demanding than both considered GEs as it processes high(er) number of IR points. However, the computational complexity significantly depends not only on the implementation, but also by the selected programming language, environment, and target platform. For this reason and w.r.t. to publicly available implementations in Python and MATLAB®, we would leave evaluation on the user.

VIII. Concluding Remarks

This paper dealt with state estimation of the nonlinear models using the Gaussian estimators. The emphasis was laid on the evaluation of the Gaussian weighted integrals needed for the state estimate moment calculation using the stochastic integration rule. Compared to other solutions to the integrals, the SIR provides asymptotically unbiased moment estimates. The SIR-based filter, denoted as the *stochastic integration filter*, was complemented with the multi-step *predictor* and *smoother* in the *full* and *square-root* forms. The SIR-based estimators have been implemented in Python for the Stone Soup infrastructure and in MATLAB® for convenience. Both implementations provide consistent results. The SIR-based estimators offer improved estimation performance over the extended and unscented KFs in a Stone Soup defined tracking scenario.

Acknowledgement: J. Duník, J. Matoušek, and O. Straka's work was co-funded by the European Union under the project ROBOPROX - Robotics and advanced industrial production (reg. no. CZ.02.01.01/00/22_008/0004590). J. Hiles and R. Niu's work was supported in part under FA9550-22-1-0038. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government.

Appendix A

The sketch of the proof of (32), (33) stems from the SIR (Algorithm II) and a recursive calculation of the integral $\hat{\mathcal{I}}_N$ (30), which can be written for increasing N from 1 to 3 (assuming $\mathbf{g}(\cdot)$ for simplicity to be scalar function of vector argument) as

$$\hat{\mathcal{I}}_1 = \mathfrak{I}^{(1)} = \sum_{i=0}^{2n_x} \mathbf{g}(\boldsymbol{\xi}^{(1,i)}_{q|l})\omega^{(1,i)}_{q|l}, \tag{59}$$

$$\hat{\mathcal{I}}_2 = (1-\tfrac{1}{2})\hat{\mathcal{I}}_1 + \mathfrak{I}^{(2)}/2 = \mathfrak{I}^{(1)}/2 + \mathfrak{I}^{(2)}/2, \tag{60}$$

$$\begin{aligned}\hat{\mathcal{I}}_3 &= (1-\tfrac{1}{3})\hat{\mathcal{I}}_2 + \mathfrak{I}^{(3)}/3 = \tfrac{2}{3}(\mathfrak{I}^{(1)}/2 + \mathfrak{I}^{(2)}/2) + \mathfrak{I}^{(3)}/3 \\ &= \mathfrak{I}^{(1)}/3 + \mathfrak{I}^{(2)}/3 + \mathfrak{I}^{(3)}/3,\end{aligned} \tag{61}$$

[4] https://stonesoup.readthedocs.io/en/v1.1/auto_tutorials/index.html

[5] Bearing measurement model is subject to wrapping.

[6] MATLAB® implementation of the unscented and SIR-based estimators is available at https://github.com/IDM-UWB/FUSION2024_SIF.

[7] The EKF estimate diverged in a small number of the MC simulations.

where the i-term can be written

$$\frac{\mathfrak{I}^{(n)}}{N} = \frac{1}{N}\sum_{i=0}^{2n_x} \mathbf{g}(\boldsymbol{\xi}_{q|l}^{(n,i)})\omega_{q|l}^{(n,i)} = \sum_{i=0}^{2n_x} \mathbf{g}(\boldsymbol{\xi}_{q|l}^{(n,i)})\tilde{\omega}_{q|l}^{(n,i)}, \quad (62)$$

which directly leads to (32), (33).

APPENDIX B: MATRIX FACTORISATION

A. Filtering Covariance Matrix

To show identity of the filtering covariance matrix forms (14) and (38), let us start with multiplication and expression of the right-hand side of (38)

$$\begin{aligned}\mathbf{P}_{k|k}^{xx} &= \tilde{\mathbf{S}}_{k|k-1}^{xx}(\tilde{\mathbf{S}}_{k|k-1}^{xx})^T - \tilde{\mathbf{S}}_{k|k-1}^{xx}(\mathbf{K}_k\tilde{\mathbf{S}}_{k|k-1}^{zz})^T \quad (63)\\ &- \mathbf{K}_k\tilde{\mathbf{S}}_{k|k-1}^{zz}(\tilde{\mathbf{S}}_{k|k-1}^{xx})^T + \mathbf{K}_k\tilde{\mathbf{S}}_{k|k-1}^{zz}(\cdot)^T + \mathbf{K}_k\mathbf{S}_k^R(\cdot)^T\end{aligned}$$

and utilisation of the following identities [24]

$$\mathbf{K}_k\mathbf{P}_{k|k-1}^{zz}\mathbf{K}_k^T = \mathbf{P}_{k|k-1}^{xz}\mathbf{K}_k^T = \mathbf{K}_k(\mathbf{P}_{k|k-1}^{xz})^T \quad (64)$$

$$= \tilde{\mathbf{S}}_{k|k-1}^{xx}(\tilde{\mathbf{S}}_{k|k-1}^{zz})^T\mathbf{K}_k^T = \mathbf{K}_k\tilde{\mathbf{S}}_{k|k-1}^{zz}(\tilde{\mathbf{S}}_{k|k-1}^{xx})^T, \quad (65)$$

$$= \mathbf{K}_k\left(\tilde{\mathbf{S}}_{k|k-1}^{zz}(\tilde{\mathbf{S}}_{k|k-1}^{zz})^T + \mathbf{S}_k^R(\mathbf{S}_k^R)^T\right)\mathbf{K}_k^T. \quad (66)$$

Substitution of (64)–(66) into (63) and w.r.t. Algorithm II and (32), (33), the covariance matrix (38) reads

$$\begin{aligned}\mathbf{P}_{k|k}^{xx} &= \tilde{\mathbf{S}}_{k|k-1}^{xx}(\tilde{\mathbf{S}}_{k|k-1}^{xx})^T - \mathbf{K}_k\mathbf{P}_{k|k-1}^{zz}\mathbf{K}_k^T\\ &- \mathbf{K}_k\mathbf{P}_{k|k-1}^{zz}\mathbf{K}_k^T + \mathbf{K}_k\mathbf{P}_{k|k-1}^{zz}\mathbf{K}_k^T, \quad (67)\end{aligned}$$

which is identical with (14).

B. Smoothing Covariance Matrix

Multiplication of (43) right-hand side leads

$$\begin{aligned}\mathbf{P}_{\mathrm{m}|k}^{xx} &= \tilde{\mathbf{S}}_{\mathrm{m}|\mathrm{m}}^{xx}(\tilde{\mathbf{S}}_{\mathrm{m}|\mathrm{m}}^{xx})^T - \tilde{\mathbf{S}}_{\mathrm{m}|\mathrm{m}}^{xx}(\mathbf{L}_\mathrm{m}\tilde{\mathbf{S}}_{\mathrm{m}+1|\mathrm{m}}^{xx})^T + \mathbf{L}_\mathrm{m}\mathbf{S}_\mathrm{m}^Q(\cdot)^T\\ &- \mathbf{L}_\mathrm{m}\tilde{\mathbf{S}}_{\mathrm{m}+1|\mathrm{m}}^{xx}(\tilde{\mathbf{S}}_{\mathrm{m}|\mathrm{m}}^{xx})^T + \mathbf{L}_\mathrm{m}\tilde{\mathbf{S}}_{\mathrm{m}+1|\mathrm{m}}^{xx}(\cdot)^T + \mathbf{L}_\mathrm{m}\mathbf{S}_{\mathrm{m}+1|k}^{xx}(\cdot)^T. \quad (68)\end{aligned}$$

Substituting the following identities [25]

$$\begin{aligned}&\mathbf{L}_\mathrm{m}\mathbf{P}_{\mathrm{m}+1|\mathrm{m}}^{xx}\mathbf{L}_\mathrm{m}^T = \mathbf{P}_{\mathrm{m},\mathrm{m}+1|\mathrm{m}}^{xx}\mathbf{L}_\mathrm{m}^T = \mathbf{L}_\mathrm{m}(\mathbf{P}_{\mathrm{m},\mathrm{m}+1|\mathrm{m}}^{xx})^T\\ &= \tilde{\mathbf{S}}_{\mathrm{m}|\mathrm{m}}^{xx}(\tilde{\mathbf{S}}_{\mathrm{m}+1|\mathrm{m}}^{xx})^T\mathbf{L}_\mathrm{m}^T = \mathbf{L}_\mathrm{m}\tilde{\mathbf{S}}_{\mathrm{m}+1|\mathrm{m}}^{xx}(\tilde{\mathbf{S}}_{\mathrm{m}|\mathrm{m}}^{xx})^T, \quad (69)\\ &= \mathbf{L}_\mathrm{m}\left(\tilde{\mathbf{S}}_{\mathrm{m}+1|\mathrm{m}}^{xx}(\tilde{\mathbf{S}}_{\mathrm{m}+1|\mathrm{m}}^{xx})^T + \mathbf{S}_\mathrm{m}^Q(\mathbf{S}_\mathrm{m}^Q)^T\right)\mathbf{L}_\mathrm{m}^T, \quad (70)\end{aligned}$$

into (68). W.r.t. Algorithm II, (19)–(22), (32), (33), we get

$$\begin{aligned}\mathbf{P}_{\mathrm{m}|k}^{xx} &= \mathbf{P}_{\mathrm{m}|\mathrm{m}}^{xx} + \mathbf{L}_\mathrm{m}\mathbf{P}_{\mathrm{m}+1|k}^{xx}\mathbf{L}_\mathrm{m}^T - \mathbf{L}_\mathrm{m}\mathbf{P}_{\mathrm{m}+1|\mathrm{m}}^{xx}\mathbf{L}_\mathrm{m}^T\\ &+ \mathbf{L}_\mathrm{m}\mathbf{P}_{\mathrm{m}+1|\mathrm{m}}^{xx}\mathbf{L}_\mathrm{m}^T - \mathbf{L}_\mathrm{m}\mathbf{P}_{\mathrm{m}+1|\mathrm{m}}^{xx}\mathbf{L}_\mathrm{m}^T, \quad (71)\end{aligned}$$

which is identical with (20).

REFERENCES

[1] Alhadhrami, E., Seghrouchni, A.E.F., Barbaresco, F., Zitar, R.A.: Drones tracking adaptation using reinforcement learning: Proximal policy optimization. In: 2023 24th International Radar Symposium (IRS). Volume 2023 24th International Radar Symposium (IRS)., German Institute of Navigation (DGON) (2023) 1–10

[2] Anderson, B.D.O., Moore, J.B.: Optimal Filtering. Prentice Hall, New Jersey (1979)

[3] Arasaratnam, I., Haykin, S.: Cubature Kalman filters. IEEE Trans. on Automatic Control **54**(6) (2009) 1254–1269

[4] Arndt, C.: Information measures. Springer (2004)

[5] Barr, J., Harrald, O., Hiscocks, S., Perree, N., Pritchett, H., Vidal, S., Wright, J., Carniglia, P., Hunter, E., Kirkland, D., Raval, D., Zheng, S., Young, A., Balaji, B., Maskell, S., Hernandez, M., Vladimirov, L.: Stone Soup open source framework for tracking and state estimation: enhancements and applications. Volume 12122., SPIE (2022) 1212205–1212205–17

[6] Blasch, E., Niu, R., O'Rourke, S.: Target tracking analysis for Stone Soup. In: Int'l Conf. on Information Fusion. (2020)

[7] Carniglia, P., Balaji, B., Damini, A.: Investigation of sensor bias and signal quality on target tracking with multiple radars. In: Int'l Inst. and Measurement Tech. Conf. (2022)

[8] Duník, J., Straka, O., Šimandl, M.: Nonlinearity and non-Gaussianity measures for stochastic dynamic systems. In: Proceedings of the 16th International Conference on Information Fusion, Istanbul (2013)

[9] Duník, J., Straka, O., Šimandl, M.: Stochastic integration filter. IEEE Transactions on Automatic Control **58**(6) (2013) 1561–1566

[10] Duník, J., Straka, O., Šimandl, M., Blasch, E.: Random-point-based filters: Analysis and comparison in target tracking. IEEE Transactions on Aerospace and Electronic Systems **51**(2) (2015) 303–308

[11] Duník, J., Šimandl, M., Straka, O.: Unscented Kalman filter: Aspects and adaptive setting of scaling parameter. IEEE Transactions on Automatic Control **57**(9) (2012) 2411–2416

[12] Genz, A., Monahan, J.: Stochastic integration rules for infinite regions. SIAM Journal on Scientific Computing **19**(2) (1998) 426–439

[13] Goodall, F., Ahmad, B.I.: Adaptation of multi-target tracker using neural networks in drone surveillance radar. In: 2023 IEEE Radar Conference (RadarConf23). (2023) 1–6

[14] Hiles, J., O'Rourke, S.M., Niu, R., Blasch, E.P.: Implementation of ensemble kalman filters in stone-soup. In: Int'l Conf. on Information Fusion. (2021)

[15] Julier, S.J., Uhlmann, J.K.: Unscented filtering and nonlinear estimation. IEEE Proceedings **92**(3) (2004) 401–421

[16] Khalid, S.S., Rehman, N.U., Abrar, S.: Robust stochastic integration filtering for nonlinear systems under multivariate t-distributed uncertainties. Signal Processing **140** (2017) 53–59

[17] Nørgaard, M., Poulsen, N.K., Ravn, O.: New developments in state estimation for nonlinear systems. Automatica **36**(11) (2000) 1627–1638

[18] Santos-León, J.C., Orive, R., Acosta, D., Acosta, L.: The cubature Kalman filter revisited. Automatica **127** (2021) 109541

[19] Särkkä, S.: Bayesian Filtering and Smoothing. Cambridge University Press (2013)

[20] Simon, D.: Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches. Wiley-Interscience (2006)

[21] Sorenson, H.W.: On the development of practical nonlinear filters. Information Sciences **7** (1974) 230–270

[22] Straka, O., Duník, J.: Stochastic integration student's-t filter. In: 2017 20th International Conference on Information Fusion (Fusion), Xi'an, China (2017)

[23] Thomas, P.A., Barr, J., Balaji, B., White, K.: An open source framework for tracking and state estimation ('stone soup'). In: SPIE Proceedings. Volume 10200., SPIE (2017) 1020008–1020008–10

[24] Šimandl, M., Duník, J.: Sigma point gaussian sum filter design using square root unscented filters, volume 1. In: Proceedings of the 16th IFAC World Congress, Prague, Czech Republic, Oxford: Elsevier (2005)

[25] Šimandl, M., Duník, J.: Design of derivative-free smoothers and predictors. IFAC Proceedings Volumes **39**(1) (2006) 1240–1245 14th IFAC Symposium on Identification and System Parameter Estimation (SYSID).

[26] Šimandl, M., Duník, J.: Derivative-free estimation methods: New results and performance analysis. Automatica **45**(7) (2009) 1749–1757

[27] Wright, J.S., Hopgood, J.R., Davies, M.E., Proudler, I.K., Sun, M.: Implementation of adaptive kernel kalman filter in Stone Soup. In: 2023 Sensor Signal Processing for Defence Conference (SSPD). (2023) 1–5

[28] Yang, C., Blasch, E.: Fusion of tracks with road constraints. J. Adv. Information Fusion **3** (2008) 14–32

[29] Zhou, D., Guo, L.: Stochastic integration H∞ filter for rapid transfer alignment of ins. Sensors **17**(11) (2017)