A Novel Measurement Processing Approach to the Parallel Expectation Propagation Unscented Kalman Filter

Allan De Freitas[1], Carsten Fritsche[2], Lyudmila Mihaylova[1], and Fredrik Gunnarsson[2,3]
[1]Department of Automatic Control and Systems Engineering, University of Sheffield, United Kingdom
[2]Department of Electrical Engineering, Linköping University, Linköping, Sweden
[3]Ericsson Research, Linköping, Sweden

Abstract—**Advances in sensor systems have resulted in the availability of high resolution sensors, capable of generating massive amounts of data. For complex systems to run online, the primary focus is on computationally efficient filters for the estimation of latent states related to the data. In this paper a novel method for efficient state estimation with the unscented Kalman Filter is proposed. The focus is on applications consisting of a massive amount of data. From a modelling perspective, this amounts to a measurement vector with dimensionality significantly greater than the dimensionality of the state vector. The efficiency of the filter is derived from a parallel filter structure which is enabled by the expectation propagation algorithm. A novel parallel measurement processing expectation propagation unscented Kalman filter is developed. The primary advantage of the novel algorithm is in the ability to achieve computational improvements with negligible loses in filter accuracy. An example of robot localization with a high resolution laser rangefinder sensor is presented. A 47.53% decrease in computational time was exhibited for a scenario with a processing platform consisting of 4 processors, with a negligible loss in accuracy.**

I. Introduction

The Kalman Filter (KF) [1] is the optimal estimator of a latent state vector in a linear dynamic system with independent additive white noise on the dynamics and observation models. However, in reality, the majority of dynamic systems contain some form of nonlinearity, e.g. the integrated navigation system in an unmanned aerial vehicle (UAV) [2]. Simulation-based methods, such as the particle filter (PF) [3], have been proposed for estimation in nonlinear systems. These methods have greater flexibility in their probabilistic representation, however, this comes at the cost of a greater computational expense which increases with the dimensionality of the state vector [4]. Nonlinear adaptations of the KF have also been proposed for estimation in nonlinear systems. The most notable nonlinear KF adaptation is the extended KF (EKF), which is based on the linearization of a nonlinear system. It has been shown that this linearilization can have a negative impact on filter accuracy and stability [5]. Alternatively, deterministic sampling based sigma point KFs have been proposed as a superior alternative. A variety of sigma point KFs have been developed, see [6] for an in-depth theoretical unification and comparison. One of the most popular sigma point KFs, and the starting point of this work, is the unscented KF (UKF).

The UKF is a derivative free approach, introduced in [7], based on the unscented transformation (UT). The UT relies on a deterministic number of weighted sigma points to approximate the mean and covariance of a Gaussian random variable that is propagated through a nonlinear function. In the standard UKF, multiple measurements are modelled by a stacked measurement vector. However, with recent advancements in sensor technology applications where excessively large amounts of measurements are received at each time step are resulting in high dimensional measurement vectors. This can result in significant computational performance degradation, since the computation of the Kalman gain requires a matrix inversion of a large matrix.

Several approaches aimed at improving the efficiency of the UKF have been developed. One basis for improvement is on the reduction of the dimensionality of the sigma points. In [8] a simplex set of sigma points was presented, reducing the number of sigma points from $2n_{\boldsymbol{x}}+1$ to $n_{\boldsymbol{x}}+2$, where $n_{\boldsymbol{x}}$ is the dimensionality of the state vector, with a minimal impact on the performance. In [9] an approach based on the minimal ensemble set of sigma points, consisting of $n_{\boldsymbol{x}}+1$ points, was presented with performance inline with the EKF. Another basis for performance improvement is the algebraic manipulation of the UKF. In [10] the square-root UKF is presented with the introduction of efficient linear algebra techniques, specifically the QR decomposition, Cholesky factor updating, and efficient least squares. In [11] a method of processing the measurements in a serial fashion with a result equivalent to the joint processing of the measurements is presented, based on the Sherman-Morrison-Woodbury identity.

In this paper a novel efficient variant of the UKF is presented. This is based on the utilisation of a computing platform with multiple processors. In this case the high dimensional measurement vector is sub-divided into non-overlapping lower dimensional measurement vectors that are processed in parallel. For the first time the Expectation Propagation (EP) algorithm and UKF are combined to enable a parallel processing UKF structure referred to as the EP-UKF.

The remainder of this paper is organised in the following manner: In Section II the problem is formulated. In Section III the derivation of the proposed estimation method is presented. Section IV describes the experiments performed. Section V illustrates the performance of the EP-UKF in comparison with the standard UKF. Finally, conclusions are summarised in Section VI.

II. Problem Formulation

Consider the evolution of an unobservable state sequence, $\boldsymbol{x}_k \in \mathbb{R}^{n_{\boldsymbol{x}}}$ at discrete time t_k, with $k = 1, ..., T \in \mathbb{N}$, given by:

$$\boldsymbol{x}_k = \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k, \boldsymbol{w}_k), \tag{1}$$

where $\boldsymbol{f}(\cdot)$ represents a function which is generally non-linear, $\boldsymbol{u}_k$ is an input and $\boldsymbol{w}_k$ represents zero-mean white Gaussian noise with covariance $\boldsymbol{Q}_k$. At each discrete time step measurements $\boldsymbol{z}_k \in \mathbb{R}^{n_z}$ are received and related to the state according to

$$\boldsymbol{z}_k = \boldsymbol{g}(\boldsymbol{x}_k) + \boldsymbol{\xi}_k, \tag{2}$$

where $\boldsymbol{g}(\cdot)$ represents a function which is generally non-linear, and $\boldsymbol{\xi}_k$ represents zero-mean white Gaussian noise with covariance $\boldsymbol{R}_k$.

In a Bayesian framework, the aim is to sequentially compute the filtering posterior state probability density function (pdf), $p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$ with $\boldsymbol{z}_{1:k} = \{\boldsymbol{z}_1, ..., \boldsymbol{z}_k\}$. Ideally, this can be achieved through a two step process when the filtering posterior state pdf at the previous time step, $p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1})$, is available. The first step is referred to as the prediction step via the Chapman-Kolmogorov equation, resulting in the predictive posterior state pdf

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) = \int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1}) d\boldsymbol{x}_{k-1}, \tag{3}$$

where $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$ represents the state transition pdf, which is the probabilistic equivalent of (1). The new measurements are utilised to update the predictive posterior state pdf via Bayes' rule

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \frac{p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) p(\boldsymbol{z}_k|\boldsymbol{x}_k)}{\int p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) p(\boldsymbol{z}_k|\boldsymbol{x}_k) d\boldsymbol{x}_k}, \tag{4}$$

where $p(\boldsymbol{z}_k|\boldsymbol{x}_k)$ is referred to as the likelihood function, which is the probabilistic equivalent of (2).

In this formulation it is assumed that the measurement is an uncorrelated high-dimensional vector composed of a large number of low dimensional measurements, i.e. $\boldsymbol{z}_k = [\boldsymbol{z}_{1,k}^\top, \boldsymbol{z}_{2,k}^\top, \ldots, \boldsymbol{z}_{M,k}^\top]^\top$, and thus the covariance reduces to $\boldsymbol{R}_k = \text{diag}(\boldsymbol{R}_{1,k}, \boldsymbol{R}_{2,k}, \ldots, \boldsymbol{R}_{M,k})$. It is also assumed that a computing platform with D processors capable of processing measurements in parallel is available. The measurement vector is decomposed into D measurement vectors, i.e. $\{\boldsymbol{z}_{k,d}\}_{d=1}^{D}$, where $\boldsymbol{z}_k = \bigcup_{d=1}^{D} \boldsymbol{z}_{k,d}$ and $\boldsymbol{z}_{k,i} \bigcap \boldsymbol{z}_{k,j} = \emptyset \ \forall \ i \neq j$, with each processor assigned a measurement vector. The filtering posterior state pdf in (4) is accordingly given by

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \frac{p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) \prod_{d=1}^{D} p(\boldsymbol{z}_{k,d}|\boldsymbol{x}_k)}{\int p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) \prod_{d=1}^{D} p(\boldsymbol{z}_{k,d}|\boldsymbol{x}_k) d\boldsymbol{x}_k}. \tag{5}$$

This results in the definition of a local likelihood for each processor d, $p(\boldsymbol{z}_{k,d}|\boldsymbol{x}_k)$.

III. Expectation Propagation Unscented Kalman Filter Derivation

EP is a deterministic approximate inference scheme, based on the minimisation of the Kullback-Leibler (KL) divergence [12]. Typically the EP approach is used to approximate posterior distributions with a simpler distribution, which is close in the sense of the KL divergence. EP is a flexible scheme which has been shown to naturally extend to the parallel processing of partitioned data [13], [14]. Here, the EP framework is utilised to approximate the likelihood terms for each processor with a member of the exponential family.

The local filtering posterior state pdf at each processor d is given by:

$$p_d(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \frac{p(\boldsymbol{z}_{k,d}|\boldsymbol{x}_k) p_d(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) \prod_{i \neq d} \pi_i(\boldsymbol{x}_k)}{\int p(\boldsymbol{z}_{k,d}|\boldsymbol{x}_k) p_d(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) \prod_{i \neq d} \pi_i(\boldsymbol{x}_k) d\boldsymbol{x}_k}, \tag{6}$$

with

$$\pi_i(\boldsymbol{x}_k) = h(\boldsymbol{x}_k) \ell(\boldsymbol{\eta}) \exp\left\{\boldsymbol{\eta}^T \boldsymbol{u}(\boldsymbol{x}_k)\right\}, \tag{7}$$

where $\boldsymbol{\eta}$ represents the natural parameters (NPs), and $h(\boldsymbol{x}_k)$, $\ell(\boldsymbol{\eta})$ and $\boldsymbol{u}(\boldsymbol{x}_k)$ are functions which vary depending on the member of the exponential family. Clearly, the local filtering posterior state pdf takes information about the measurements from the other processors into account via $\pi_i(\boldsymbol{x}_k)$, thus being an approximation of the global posterior distribution, $p_d(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) \approx p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k})$. The degree to which the approximation is true is dependent on how accurately the likelihood terms are approximated.

In this paper a KF based approach is considered. In this case all the pdfs related to the Bayesian recursion are approximated with Gaussian distributions, i.e.

$$p_d(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1}) \approx \mathcal{N}(\boldsymbol{x}_{k-1}; \hat{\boldsymbol{x}}_{d,k-1|k-1}, \boldsymbol{P}_{d,k-1|k-1}), \tag{8a}$$

$$p_d(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) \approx \mathcal{N}(\boldsymbol{x}_k; \hat{\boldsymbol{x}}_{d,k|k-1}, \boldsymbol{P}_{d,k|k-1}), \tag{8b}$$

$$p_d(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) \approx \mathcal{N}(\boldsymbol{x}_k; \hat{\boldsymbol{x}}_{d,k|k}, \boldsymbol{P}_{d,k|k}), \tag{8c}$$

$$\pi_d(\boldsymbol{x}_k) \approx \mathcal{N}(\boldsymbol{x}_k; \boldsymbol{\mu}_d, \boldsymbol{P}_d). \tag{8d}$$

Due to the Gaussian approximations, the local posterior state pdf in (6) is further reduced into the same form as the general posterior state pdf in (4),

$$p_d(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \frac{p(\boldsymbol{z}_{k,d}|\boldsymbol{x}_k) \frac{1}{\zeta} p_{\tilde{d}}(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1})}{\int p(\boldsymbol{z}_{k,d}|\boldsymbol{x}_k) \frac{1}{\zeta} p_{\tilde{d}}(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) d\boldsymbol{x}_k}, \tag{9}$$

where ζ represents a normalisation constant independent of $\boldsymbol{x}_k$, and $p_{\tilde{d}}(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) = \mathcal{N}(\boldsymbol{x}_k; \hat{\boldsymbol{x}}_{\tilde{d},k|k-1}, \boldsymbol{P}_{\tilde{d},k|k-1})$ with

$$\hat{\boldsymbol{x}}_{\tilde{d},k|k-1} = \left(\boldsymbol{P}_{d,k|k-1}^{-1} + \sum_{i \neq d} \boldsymbol{P}_i^{-1}\right)^{-1} \times \left(\boldsymbol{P}_{d,k|k-1}^{-1} \hat{\boldsymbol{x}}_{d,k|k-1} + \sum_{i \neq d} \boldsymbol{P}_i^{-1} \boldsymbol{\mu}_i\right),$$

$$\boldsymbol{P}_{\tilde{d},k|k-1} = \left(\boldsymbol{P}_{d,k|k-1}^{-1} + \sum_{i \neq d} \boldsymbol{P}_i^{-1}\right)^{-1}. \tag{10}$$

Another form of the local posterior state pdf is

$$p_d(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \frac{p_d(\boldsymbol{x}_k, \boldsymbol{z}_{k,d}|\boldsymbol{z}_{1:k-1})}{\int p_d(\boldsymbol{x}_k, \boldsymbol{z}_{k,d}|\boldsymbol{z}_{1:k-1})d\boldsymbol{x}_k} \tag{11}$$

where the local joint pdf is

$$p_d(\boldsymbol{x}_k, \boldsymbol{z}_{k,d}|\boldsymbol{z}_{1:k-1}) = \mathcal{N}((\boldsymbol{x}_k^\top, \boldsymbol{z}_{k,d}^\top)^\top; \boldsymbol{\mu}_{xz}, \boldsymbol{P}_{xz}), \tag{12}$$

with

$$\begin{aligned}
\boldsymbol{\mu}_{xz} &= \begin{pmatrix} \hat{\boldsymbol{x}}_{\tilde{d},k|k-1} \\ \hat{\boldsymbol{z}}_{d,k|k-1} \end{pmatrix} \\
\boldsymbol{P}_{xz} &= \begin{pmatrix} \boldsymbol{P}_{\tilde{d},k|k-1} & \boldsymbol{P}_{xz,k|k-1} \\ \boldsymbol{P}_{xz,k|k-1}^\top & \boldsymbol{P}_{zz,k|k-1} \end{pmatrix},
\end{aligned} \tag{13}$$

and where

$$\begin{aligned}
\hat{\boldsymbol{z}}_{d,k|k-1} &= \int \boldsymbol{g}(\boldsymbol{x}_k)\mathcal{N}(\boldsymbol{x}_k; \hat{\boldsymbol{x}}_{\tilde{d},k|k-1}, \boldsymbol{P}_{\tilde{d},k|k-1})d\boldsymbol{x}_k \\
\boldsymbol{P}_{xz,k|k-1} &= \int (\boldsymbol{x}_k - \hat{\boldsymbol{x}}_{\tilde{d},k|k-1})(\boldsymbol{g}(\boldsymbol{x}_k) - \hat{\boldsymbol{z}}_{d,k|k-1})^\top \\
&\quad \times \mathcal{N}(\boldsymbol{x}_k; \hat{\boldsymbol{x}}_{\tilde{d},k|k-1}, \boldsymbol{P}_{\tilde{d},k|k-1})d\boldsymbol{x}_k \\
\boldsymbol{P}_{zz,k|k-1} &= \int (\boldsymbol{g}(\boldsymbol{x}_k) - \hat{\boldsymbol{z}}_{d,k|k-1})(\boldsymbol{g}(\boldsymbol{x}_k) - \hat{\boldsymbol{z}}_{d,k|k-1})^\top \\
&\quad \times \mathcal{N}(\boldsymbol{x}_k; \hat{\boldsymbol{x}}_{\tilde{d},k|k-1}, \boldsymbol{P}_{\tilde{d},k|k-1})d\boldsymbol{x}_k.
\end{aligned} \tag{14}$$

It has been shown, e.g. in [15], that the substitution of (12) into (11) leads to the following update of the local posterior state pdf parameters:

$$\begin{aligned}
\hat{\boldsymbol{x}}_{d,k|k} &= \hat{\boldsymbol{x}}_{\tilde{d},k|k-1} + \boldsymbol{K}_k(\boldsymbol{z}_{k,d} - \hat{\boldsymbol{z}}_{d,k|k-1}) \\
\boldsymbol{P}_{d,k|k} &= \boldsymbol{P}_{\tilde{d},k|k-1} - \boldsymbol{K}_k\boldsymbol{P}_{zz,k|k-1}\boldsymbol{K}_k^\top \\
\boldsymbol{K}_k &= \boldsymbol{P}_{xz,k|k-1}\boldsymbol{P}_{zz,k|k-1}^{-1}.
\end{aligned} \tag{15}$$

However, due to the non-linearities associated with (1) and (2), no analytical solution exists for the expressions in (10) and (14). A natural approach to overcoming this challenge is through the utilisation of sigma point transformation. Here this is exemplified with the scaled unscented transformation (SUT). This is a method for determining the statistics of a random variable which undergoes a non-linear transformation. Given a random variable $\boldsymbol{y}$ of dimension n_y that follows a Gaussian distribution with mean $\hat{\boldsymbol{y}}$ and covariance $\boldsymbol{P}_y$. The random variable is subjected to a non-linear transformation resulting in a new random variable

$$\boldsymbol{r} = \boldsymbol{g}(\boldsymbol{y}). \tag{16}$$

The aim of the SUT is to determine the first two moments of the distribution of $\boldsymbol{r}$. Initially a set of $2n_y + 1$ weighted samples, $\boldsymbol{S} = (\boldsymbol{W}, \boldsymbol{Y})$ referred to as sigma points, are deterministically selected. The sigma points capture the statistics of the distribution of $\boldsymbol{y}$. The sigma points are determined according to the following set of equations [16]

$$\begin{aligned}
\boldsymbol{Y}_0 &= \hat{\boldsymbol{y}} \\
\boldsymbol{Y}_i &= \hat{\boldsymbol{y}} + \left(\sqrt{(n_y + \lambda)\boldsymbol{P}_y}\right)_i \quad i = 1, ..., n_y \\
\boldsymbol{Y}_i &= \hat{\boldsymbol{y}} - \left(\sqrt{(n_y + \lambda)\boldsymbol{P}_y}\right)_i \quad i = n_y + 1, ..., 2n_y \\
W_0^{(m)} &= \frac{\lambda}{(n_y + \lambda)} \\
W_0^{(c)} &= \frac{\lambda}{(n_y + \lambda)} + (1 - \alpha^2 + \beta) \\
W_i^{(m)} &= W_i^{(c)} = \frac{1}{2(n_y + \lambda)} \quad i = 1, ..., 2n_y,
\end{aligned} \tag{17}$$

where $\lambda = \alpha^2(n_y + \kappa) - n_y$, $\left(\sqrt{(n_y + \lambda)\boldsymbol{P}_y}\right)_i$ is the ith row or column of the matrix square root of $(n_y + \lambda)\boldsymbol{P}_y$, and α, β and κ are scaling parameters. The indices (m) and (c) represent the weights for the mean and covariance. Each sigma point is then propagated through the non-linear transformation,

$$\boldsymbol{R}_i = \boldsymbol{g}(\boldsymbol{Y}_i) \quad i = 0, ..., 2n_y. \tag{18}$$

The first two moments of the transformed random variable, $\boldsymbol{r}$, can then be approximated by

$$\begin{aligned}
\hat{\boldsymbol{r}} &= \sum_{i=0}^{2n_y} W_i^{(m)}\boldsymbol{R}_i \\
\boldsymbol{P}_r &= \sum_{i=0}^{2n_y} W_i^{(c)}(\boldsymbol{R}_i - \hat{\boldsymbol{r}})(\boldsymbol{R}_i - \hat{\boldsymbol{r}})^\top.
\end{aligned} \tag{19}$$

Once this principal has been applied to obtain the parameters in (15) for the local posterior state pdfs, the likelihood approximation for each processor d is updated according to

$$\begin{aligned}
\boldsymbol{\mu}_d &= \left(\boldsymbol{P}_{d,k|k}^{-1} - \boldsymbol{P}_{\tilde{d},k|k-1}^{-1} - \sum_{i \neq d} \boldsymbol{P}_i^{-1}\right)^{-1} \\
&\quad \times \left(\boldsymbol{P}_{d,k|k}^{-1}\hat{\boldsymbol{x}}_{d,k|k} - \boldsymbol{P}_{\tilde{d},k|k-1}^{-1}\hat{\boldsymbol{x}}_{\tilde{d},k|k-1} - \sum_{i \neq d} \boldsymbol{P}_i^{-1}\boldsymbol{\mu}_i\right), \\
\boldsymbol{P}_d &= \left(\boldsymbol{P}_{d,k|k}^{-1} - \boldsymbol{P}_{\tilde{d},k|k-1}^{-1} - \sum_{i \neq d} \boldsymbol{P}_i^{-1}\right)^{-1}.
\end{aligned} \tag{20}$$

The updated parameters are shared among the processors and the procedure is iterated until reaching convergence. However, convergence is not always guaranteed [17]. Here the number of iterations is treated as a fixed parameter, L. The algorithm for the EP-UKF is presented in Algorithm 1.

IV. Application to Mobile Robot Localization

Mobile robots are required to know their position and orientation, also known as their pose, when performing tasks in a known environment. The online estimation of the pose is referred to as localization. The localization problem can be defined as either local or global. Here the local form of the problem is considered, where the aim is to compensate for odometry errors which occur during robot navigation. This problem has been previously approached with the standard forms of the EKF and UKF [18].

The pose of the robot at discrete time instance k is represented by $\boldsymbol{x}_k = (x_k, y_k, \theta_k)^\top$, where (x_k, y_k) represents the position of the robot in a two dimensional plane, and θ_k is the orientation of the robot. The pose of the robot evolves according to [19]

$$\begin{aligned}\boldsymbol{x}_k &= \boldsymbol{x}_{k-1} + T_s \begin{bmatrix} \cos\theta_k & 0 \\ \sin\theta_k & 0 \\ \frac{-\sin\gamma_k}{l_F\cos\gamma_k + l_R} & \frac{-l_R}{l_F\cos\gamma_k + l_R} \end{bmatrix} \begin{bmatrix} v_k \\ \omega_k \end{bmatrix}, \\ &= \boldsymbol{f}(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k), \end{aligned} \tag{21}$$

where $\boldsymbol{u}_k = (v_k, \omega_k)^\top$ is the speed and steering rate inputs, γ_k is the observed steering angle, parameters l_F and l_R represent the distances from the front and rear wheel axes to the hinge angle respectively, and T_s is the sample time. The motion noise covariance matrix, $\boldsymbol{Q}_k$, is given by

$$\boldsymbol{Q}_k = \begin{pmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_\omega^2 \end{pmatrix}, \tag{22}$$

where σ_v^2 and σ_ω^2 are the variances associated with the speed and steering rate inputs.

Algorithm 1 Expectation Propagation Unscented Kalman Filter for node d

```
Initialisation ($k$ = 0): $\hat{\boldsymbol{x}}_{d,0|0}$, $\boldsymbol{P}_{d,0|0}$
$\hat{\boldsymbol{x}}^a_{d,0|0} = \left(\hat{\boldsymbol{x}}^\top_{d,0|0} \;\; \mathbf{0}\right)^\top$
$\boldsymbol{P}^a_{d,0|0} = \begin{pmatrix} \boldsymbol{P}_{d,0|0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{Q}_0 \end{pmatrix}$
for $k$ = 1,...,$T$ do
  for $l$ = 1,...,$L$ do
    Calculate sigma points, $\boldsymbol{S}^a_{k-1}$, according to (17) based on $\hat{\boldsymbol{x}}^a_{d,k-1|k-1}$ and $\boldsymbol{P}^a_{d,k-1|k-1}$.
    Perform time updates:
    $\boldsymbol{X}^x_{d,k|k-1} = \boldsymbol{f}\left(\boldsymbol{X}^x_{d,k-1}, \boldsymbol{u}_k, \boldsymbol{X}^w_{d,k-1}\right)$
    $\hat{\boldsymbol{x}}_{d,k|k-1} = \sum_{i=0}^{2n_a} W_i^{(m)} \boldsymbol{X}^x_{i,d,k|k-1}$
    $\boldsymbol{P}_{d,k|k-1} = \sum_{i=0}^{2n_a} W_i^{(c)} (\boldsymbol{X}^x_{i,d,k|k-1} - \hat{\boldsymbol{x}}_{d,k|k-1}) \times (\boldsymbol{X}^x_{i,d,k|k-1} - \hat{\boldsymbol{x}}_{d,k|k-1})^\top$
    Calculate $\hat{\boldsymbol{x}}_{\widetilde{d},k|k-1}$ and $\boldsymbol{P}_{\widetilde{d},k|k-1}$ with (10).
    Re-calculate sigma points, $\boldsymbol{S}^a_{k|k-1}$, according to (17) based on $\hat{\boldsymbol{x}}_{\widetilde{d},k|k-1}$ and $\boldsymbol{P}_{\widetilde{d},k|k-1}$.
    $\boldsymbol{Z}_{d,k|k-1} = \boldsymbol{g}\left(\boldsymbol{X}^x_{d,k|k-1}\right)$
    $\hat{\boldsymbol{z}}_{d,k|k-1} = \sum_{i=0}^{2n_a} W_i^{(m)} \boldsymbol{Z}_{i,d,k|k-1}$
    Perform measurement updates:
    $\boldsymbol{P}_{zz,k|k-1} = \sum_{i=0}^{2n_a} W_i^{(c)} (\boldsymbol{Z}_{i,d,k|k-1} - \hat{\boldsymbol{z}}_{d,k|k-1}) \times (\boldsymbol{Z}_{i,d,k|k-1} - \hat{\boldsymbol{z}}_{d,k|k-1})^\top$
    $\boldsymbol{P}_{xz,k|k-1} = \sum_{i=0}^{2n_a} W_i^{(c)} (\boldsymbol{X}^x_{i,d,k|k-1} - \hat{\boldsymbol{x}}_{\widetilde{d},k|k-1}) \times (\boldsymbol{Z}_{i,d,k|k-1} - \hat{\boldsymbol{z}}_{d,k|k-1})^\top + \boldsymbol{R}_{d,k}$
    $\boldsymbol{K}_k = \boldsymbol{P}_{xz,k|k-1} \boldsymbol{P}^{-1}_{zz,k|k-1}$
    $\hat{\boldsymbol{x}}_{d,k|k} = \hat{\boldsymbol{x}}_{\widetilde{d},k|k-1} + \boldsymbol{K}_k (\boldsymbol{z}_{d,k} - \hat{\boldsymbol{z}}_{d,k|k-1})$
    $\boldsymbol{P}_{d,k|k} = \boldsymbol{P}_{\widetilde{d},k|k-1} - \boldsymbol{K}_k \boldsymbol{P}_{zz,k|k-1} \boldsymbol{K}_k^\top$
    Calculate $\boldsymbol{\mu}_d$ and $\boldsymbol{P}_d$ according to (20).
    Share $\boldsymbol{\mu}_d$ and $\boldsymbol{P}_d$ to the set $D \setminus d$ processors.
    Receive $\boldsymbol{\mu}_i$ and $\boldsymbol{P}_i$ $\forall\ i \neq d$ from the set $D \setminus d$ processors.
  end for
end for
*Where $\boldsymbol{X}^a = \left((\boldsymbol{X}^x)^\top (\boldsymbol{X}^w)^\top\right)^\top$.
```

The novelty of the EP-UKF lies within the measurement update. For this application the UKF time update step is interchanged with the EKF time update step without retracting from the novelty of the EP-UKF. This is done purely to increase the computational efficiency and is possible in this application since there is a closed form motion model. The time update is thus given by

$$\begin{aligned} \hat{\boldsymbol{x}}_{k|k-1} &= \boldsymbol{f}(\hat{\boldsymbol{x}}_{k-1|k-1}, \boldsymbol{u}_k), \\ \boldsymbol{P}_{k|k-1} &= \boldsymbol{H}_{\boldsymbol{u},k} \boldsymbol{Q}_k \boldsymbol{H}^\top_{\boldsymbol{u},k} + \boldsymbol{H}_{\boldsymbol{x},k} \boldsymbol{P}_{k-1|k-1} \boldsymbol{H}^\top_{\boldsymbol{x},k}, \end{aligned} \tag{23}$$

where

$$\begin{aligned} \boldsymbol{H}_{\boldsymbol{x},k} &= \begin{pmatrix} 1 & 0 & -T_s v_k \sin\theta_k \\ 0 & 1 & h v_k \cos\theta_k \\ 0 & 0 & 1 \end{pmatrix}, \\ \boldsymbol{H}_{\boldsymbol{u},k} &= \begin{pmatrix} T_s\cos\theta_k & 0 \\ T_s\sin\theta_k & 0 \\ \frac{-T_s\sin\gamma_k}{l_F\cos\gamma_k + l_R} & \frac{-T_s l_R}{l_F\cos\gamma_k + l_R} \end{pmatrix}. \end{aligned} \tag{24}$$

The robot is located in a room with a known map. The measurement sensor used by the robot for localization is a laser rangefinder. The sensor uses a laser beam to determine the distance to an obstruction. The laser rangefinder obtains a set of distance measurements dispersed 360° around the robot with equi-distance angular spacing between measurements, see Figure 1. The ith element of $\boldsymbol{z}_k$ is a range measurement related

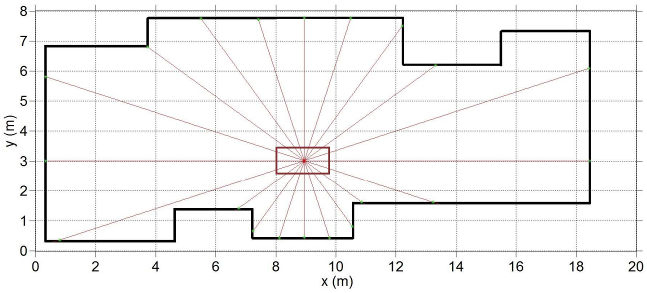

Fig. 1: An illustrative example of the range measurements observed by a robot, for the case of 20 measurements.

to the position of the robot through

$$z_{i,k} = \sqrt{(x_k - x_{w,i})^2 + (y_k - y_{w,i})^2} + \xi_{i,k}, \qquad (25)$$

where $(x_{w,i}, y_{w,i})^\top$ represents the position of the wall coinciding with the laser beam, and $\xi_{i,k} \sim \mathcal{N}(0, R_i)$.

V. Results

Consider the scenario of a robot with a multi-core digital signal processor navigating a known environment with a high angular resolution laser rangefinder. Both the standard UKF, and the EP-UKF are utilised for the inference of the robot pose over several experiments, based on the models in Section IV. The algorithms are implemented in the interpreted language MATLAB. The parallel processing for the EP-UKF is achieved in MATLAB with the `parfor` command. All simulations were performed on a mobile computer with Intel(R) Core(TM) i7-4702HQ CPU @ 2.20GHz with 16GB of RAM.

Two different methods are used to compare the performance of the filters. The first is the root mean square error (RMSE) of the pose. The RMSE for each time step is calculated over a number of independent simulation runs according to

$$RMSE = \sqrt{\frac{1}{N_I}\sum_{i=1}^{N_I}(\hat{X}_i - X_i)^2}, \qquad (26)$$

where X represents a specific component of the state vector $\boldsymbol{x}_k$, with X_i the ground truth, $\hat{X}_i$ represents the algorithm estimate, which corresponds to the mean of the UKF in this application, and N_I represents the number of independent runs. The RMSE of the states corresponding to the position are averaged to obtain a single result. The RMSE illustrates the tracking accuracy of the algorithms. The second is the average MATLAB execution time, which illustrates the computational efficiency of the algorithms.

A. Parameters

The following parameters were utilised unless otherwise specified. The number of independent simulation runs is $N_I = 100$. The number of time simulation steps is $T = 80$. The motion model parameters are $T_s = 1$, $\boldsymbol{u} = (0.2, 0)^\top$, $\sigma_v^2 = 1 \times 10^{-3}$, $\sigma_\omega^2 = 1 \times 10^{-4}$ and $\gamma_{0:39} = 0$ with a step to $\gamma_{40:80} = -0.5$, and $l_F = 2$, $l_R = 2$. The number of processor cores is $D = 4$. The target observation model parameters are $M = 200$ and $R_i = 1 \times 10^{-2}$. The number of EP iterations is $L = 2$.

B. Performance Evaluation

The robot trajectory for the experiments is illustrated in Figure 2. The RMSE of the robot position has been averaged over the position dimensions, and also over the estimates for the individual processors in the EP-UKF, this result is presented in Figure 3. This result illustrates that there is no significant reduction in the accuracy of the estimate obtained by the EP-UKF.

The RMSE fluctuates over the course of the simulation because of the complex environment, but overall is highly accurate since a high number of measurements are collected at each discrete time step. For the given experiment, the average MATLAB execution time per time step for each algorithm is illustrated in Figure 4. An increase in the number of processors reduces the algorithm execution time. There is a decrease in computational time of 47.53% given 4 processors. In this scenario, it may be expected to have a result closer to 75%, however, this is closer to a value of 50% due to the EP iteration $L = 2$. This value of EP iteration also explains why there is no computational gain when considering 2 processors.

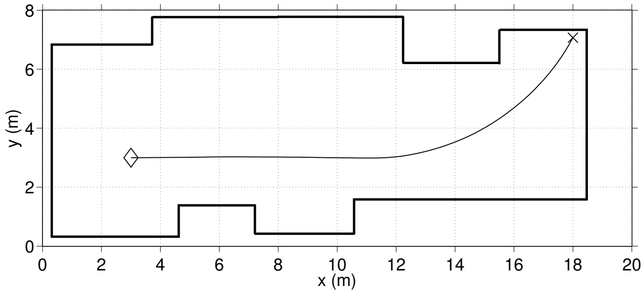

Fig. 2: The robot trajectory for the experiments, where ◇ and × represents the starting and end points, respectively.

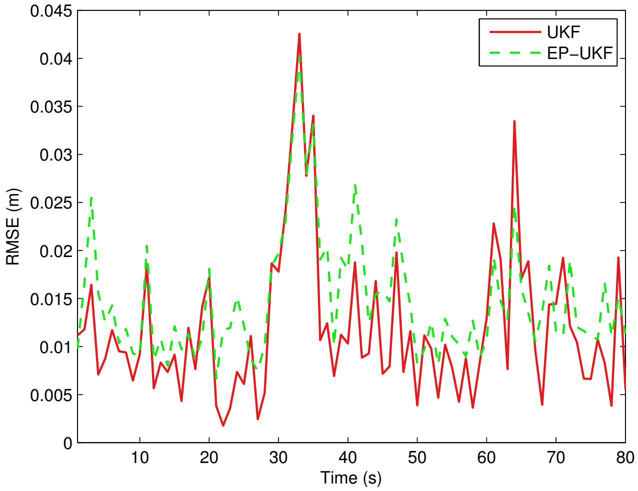

Fig. 3: Average RMSE of the robot position.

VI. Conclusions

In this paper a novel method for efficient state estimation with the EP-UKF for massive amounts of measurements is proposed. This is based on a parallel filter structure enabled by the combination of the EP algorithm and UKF. The primary advantage of the algorithm is in the ability to achieve computational improvements with negligible loses in filter accuracy. In this paper a 47.53% decrease in computational time was exhibited for a case with a processing platform consisting of 4 processors. An additional advantage is the flexibility of the algorithm. The number of processors can vary according to the processing platform available.

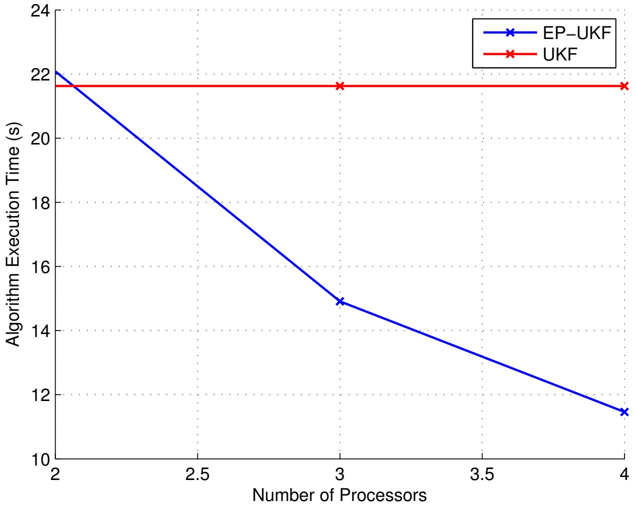

Fig. 4: Average MATLAB algorithm execution time per time step.

Acknowledgments

We acknowledge the support from the EC Seventh Framework Programme [FP7 2013-2017] TRAcking in compleX sensor systems (TRAX) Grant agreement no.: 607400. The authors are also grateful to the UK-China Mobility grant: Multivehicle tracking and classification for intelligent transportation systems (Reference number E150823) from the UK Royal Society fund.

References

[1] R. E. Kalman, “A New Approach to Linear Filtering and Prediction Problems,” *Trans. of ASME-Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[2] E. Wan, “Sigma-Point Filters: An Overview with Applications to Integrated Navigation and Vision Assisted Control,” in *IEEE Nonlinear Statistical Signal Processing Workshop*, Sep. 2006, pp. 201–202.

[3] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, “A Tutorial on Particle Filters for Online Nonlinear/non-Gaussian Bayesian Tracking,” *IEEE Trans. on Signal Processing*, vol. 50, no. 2, pp. 174 –188, Feb. 2002.

[4] F. Dellaert, D. Fox, and S. Thrun, “Monte Carlo Localization for Mobile Robots,” in *Proc. of Intl. Conf. on Robotics and Automation*, May 1999.

[5] E. Wan and R. van der Merwe, “The Unscented Kalman Filter,” in *Kalman Filtering and Neural Networks*, S. Haykin, Ed. Wiley Publishing, 2001.

[6] M. Roth, G. Hendeby, and F. Gustafsson, “Nonlinear Kalman Filters Explained: A Tutorial on Moment Computations and Sigma Point Methods,” *Journal of Advances in Information Fusion*, vol. 11, no. 1, pp. 47–70, June 2016.

[7] S. Julier, J. Uhlmann, and H. Durrant-White, “A New approach for Filtering Nonlinear Systems,” in *Proc. of American Control Conf.*, June 1995, pp. 1628–1632.

[8] S. J. Julier and J. K. Uhlmann, “Reduced Sigma Point Filters for the Propagation of Means and Covariances Through Nonlinear Transformations,” in *Proc. of American Control Conf.*, vol. 2, May 2002, pp. 887–892.

[9] B. M. Quine, “A Derivative-free Implementation of the Extended Kalman Filter,” *Automatica*, vol. 42, no. 11, pp. 1927 – 1934, Nov. 2006.

[10] R. van der Merwe and E. Wan, “The Square-Root Unscented Kalman Filter for State and Parameter Estimation,” in *Proc. of Intl. Conf. on Acoustics, Speech, and Signal Processing*, vol. 6, May 2001, pp. 3461–3464.

[11] C. McManus and T. D. Barfoot, “A Serial Approach to Handling High-Dimensional Measurements in the Sigma-Point Kalman Filter,” in *Proc. of Robotics; Science and Systems*, June 2011.

[12] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 2006.

[13] M. Xu, Y. W. Teh, J. Zhu, and B. Zhang, “Distributed Context-Aware Bayesian Posterior Sampling via Expectation Propagation,” in *Proc. of Advances in Neural Information Processing Systems*, Dec. 2014.

[14] A. Gelman, A. Vehtari, P. Jylnki, C. Robert, N. Chopin, and J. P. Cunningham, “Expectation propagation as a way of life,” *preprint, http://arxiv.org/abs/1412.4869*, 2014.

[15] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*, 1st ed. Wiley-Interscience, June 2001.

[16] S. Julier, “The Scaled Unscented Transformation,” in *Proc. of American Control Conf.*, vol. 6, May 2002, pp. 4555–4559.

[17] M. Seeger, “Expectation Propagation for Exponential Families,” Max Planck Institute, Tuebingen, Tech. Rep., Nov. 2005.

[18] L. D’Alfonso, W. Lucia, P. Muraca, and P. Pugliese, “Mobile Robot Localization via EKF and UKF: A Comparison Based on Real Data,” *Robotics and Autonomous Systems*, vol. 74, Part A, pp. 122 – 127, Dec. 2015.

[19] J. Marshall, T. Barfoot, and J. Larsson, “Autonomous Underground Tramming for Center-Articulated Vehicles,” *Journal of Field Robotics*, vol. 25, no. 6-7, pp. 400–421, April 2008.