Simultanuous Localization and Mapping

A Simulation Platform

Thomas Hanefeld Sejerøe, Ole Ravn
Ørsted•DTU, Automation,
E-mail:s991707@student.dtu.dk, or@oersted.dtu.dk
Building 326,

Niels Kjølstad Poulsen
Informatics and Mathematical Modelling
E-mail:nkp@imm.dtu.dk
Building 321,

The Technical University of Denmark,
DK-2800 Kgs. Lyngby, Denmark

March 21, 2005

Abstract

In this paper we present a simulation platform for evaluate methods for simultaneous location and mapping. The platform is based on The KALMTOOL 2 toolbox which is a set of MATLAB tools for state estimation for nonlinear systems. The toolbox contains functions for extended Kalman filtering as well as for two new filters called the DD1 filter and the DD2 filter. It also contains function for Uncented Kalman filters as well as three versions of particle filters. The toolbox requires MATLAB ver. 6, but no additional toolboxes are required.

1 Introduction

The simulation platform, which is described in this paper, is build as an extention to **Kalmtool 2**. This toolbox is a collection of estimation algorithms for solving nonlinear state estimation problems.

During the work it was found that the extended Kalman filter was somewhat inconvenient to use in some of our applications. A small modification of the application sometimes had serious implications on the EKF implementation. Moreover, it was often difficult to implement. Our problem was that the EKF requires a linearization of the system model. Sometimes this is easy to find but sometimes it can be pretty hard. In any case, it makes things inflexible.

If a small change is made in the model, one has to work out a new set of derivatives. This is particularly inconvenient in model calibration where certain model parameters are temporarily included in the state vector and estimated simultaneously with the actual states.

Since it was suggested, the extended Kalman filter (EKF) has undoubtly been the dominating technique for nonlinear state estimation. Nevertheless, the EKF is known to have several drawbacks. These are mainly due to the Taylor linearization of the nonlinear transformations around the current state estimate. The linearization requires that Jacobians of state transition and observation equations are derived, which is often a quite complex task. Moreover, sometimes there are points in which the Jacobians are not defined. In addition to the difficulties with implementation, convergence problems are often encountered due to the fact that the linearized models describe the system poorly.

There have been significant focus on this area recently and previous work include several toolboxes and other platforms. ReBEL (Recursive Bayesian Estimation Library) (van der Merwe 2004) is a Matlabő toolkit of functions and scripts, designed to facilitate sequential Bayesian inference (estimation) in general state space models. The CAS Robot Navigation Toolbox (Arras 2004) is a tool for doing off-line off-board localization and SLAM on mobile robots. The design of the CAS toolbox decouples robot model, sensor models, features and algoritms used giving the user ability to adapt the toolbox by just modifying or adding the pieces in question. The toolbox does not in its present form support the generation of realtime code for use on the robot. The present platform Kalmtool II has its root in Kalmtool but focus here is on comparision and transparency giving the developer more control over the process of adapting changes and keeping housekeeping code minimal.

The paper is organized as follows: first the overall design philosophy behind the platform is described. Next a description of the estimation algorithms are given including the extended Kalman filter, the Uncented Kalman filter and different types of particlefilters. Section 4 gives an extensive example study as well as a demonastration of the platform for comparing algortims for navigation of a mobile robot. Finally conclusions and references are given.

2 The Platform

The overall design philosophy has been to put focus on making a simple, transparent, yet powerfull platform that and makes life easy to use both for application and algorithm developer.

Transperancy overcomes the barrier effect that is often expirenced when using tools that at first sight seem very user friendly but when used on real problems becomes difficult to handle due to the inherent complexity.

The approach taken uses MATLAB as a numerical and graphical basis for developing the platform. The platform is driven from Simulink as this provides

a shorter path to implementation using for instance Realtime Workshop and makes is simple to use real data for comparison.

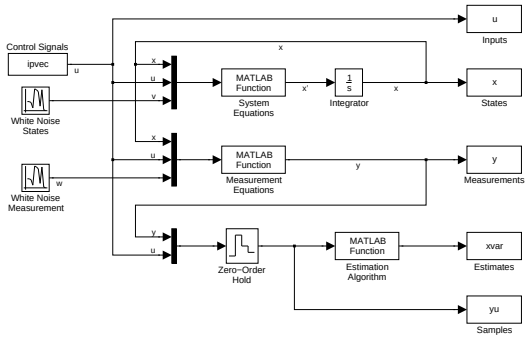

Figure 1. *The Simulink layout of a continous system.*

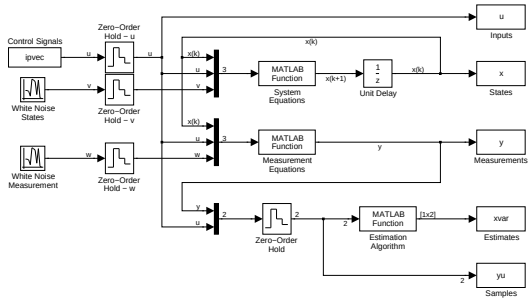

Figure 2. *The Simulink layout of a discrete time systems*

As seen in the above figures the user can easily add new algorithm into the platform by modifying the MATLAB function in the Estimation block and change the system by modifying the system and measurement MATLAB blocks.

3 Location and Mapping

When maneuvering an autonomous guide vehicle (AGV) it is important to know the position and orientation of the vehicle. This is often done by using the odonometry of the vehicle. This is basically just to use the measured traveling distance and measured change of orientation. This is also denoted as dead-reckoning. It is however well known that this method has an inherit nature of accumulating errors. The determination of the position and orientation is therefore supplied with measurement of the robot position and orientation in relation to some guide marks with known (or relative well known) positions. This, dead-reckoning and eventually the use of some guide marks is denoted as location or navigation.

The model of the mobile robot (unicycle type) is given by the set of equations which are slightly nonlinear. The equations yield a position as well as a heading.

The input signals (i.e. control signals) are the velocity, γ, and turnrate, ω.

$$\frac{d}{dt}\begin{pmatrix} x_t \\ y_t \\ \theta_t \end{pmatrix} = \begin{pmatrix} \gamma_t\ cos(\theta_t) \\ \gamma_t\ sin(\theta_t) \\ \omega_t \end{pmatrix} + v_t \tag{1}$$

The process noise is in the (later) example studies simulated as $N(0, 0.01\ I_3)$. The estimation procedure in the location part and the mapping part is based on a sampled version of the above process equation. The sampling can be done analytically (for this simple example) or by means of a numerical ODE solver.

Location in relation autonomuous guided vehicle is based on a fusion of results from several sensors. Normally one of the sensors set is the odometry, i.e. noisy measurements of the speed of the wheels:

$$\omega_r = \frac{2\gamma_t + b\omega_t}{2r_r}$$

$$\omega_l = \frac{2\gamma_t - b\omega_t}{2r_l}$$

Another set of measurements is the relative position between a guide marks and the robot. Assume a guide mark has a position which is known with some precision embedded in

$$\begin{bmatrix} x_g \\ y_g \end{bmatrix} \in \mathbf{N}\left(0, P_g\right)$$

The position of the robot is also known with some precision reflected by

$$\begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} \in \mathbf{N}\left(\begin{bmatrix} \hat{x}_t \\ \hat{y}_t \\ \hat{\theta}_t \end{bmatrix}, P_t\right)$$

The actual measurement is the distance and the direction to the guide mark which can be transformed into a set of Cartesian measurement:

$$y_t = \begin{bmatrix} 1 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_t \\ y_t \\ \theta_t \\ x_g \\ y_g \end{bmatrix} + e_t \tag{2}$$

The mesurement noise is assumend to be $\mathbf{N}\left(0, R_2\right)$ where R_2 reflects the transformation of the uncertainty in the mesurements of the distanse and the direction from the robot to the guide mark.

Both location and mapping is based on the same principle. In connection to mapping a newly observed guide mark is assumed to have a position given by the a'priori distribution

$$\begin{bmatrix} x_g \\ y_g \end{bmatrix} \in \mathbf{N}\left(p, P_0\right) \tag{3}$$

reflecting the lack of knowledge. As a limit it can be assumed to be totally flat.

4 Estimation algorithms

Consider a system in which the evolution of the state sequence $\{x_k \in \mathbb{R}^n,\ k \in \mathbb{N}\}$ is given by

$$x_{k+1} = f_k(x_k, u_k, v_k) \tag{4}$$

where f_k is a possible nonlinear function of the state, x_k, the input (control) signal, u_k and the process noise, v_k. The process noise is assumed to be a sequence $\{v_k \in \mathbb{R}^n\ k \in \mathbb{N}\}$ of i.i.d. stochastic vectors.

The objective is to estimate x_k from measurements

$$y_k = g_k(x_k, e_k) \qquad \in \mathbb{R}^m \tag{5}$$

where also g_k is a possible nonlinear function of the state and the measurement noise, e_k. The measurement noise is assumed to be a sequence, $\{e_k \in \mathbb{R}^m\ k \in \mathbb{N}\}$, of i.i.d. stochastic vectors. More specific we seek an estimate of x_k based on all available measurements (and known inputs) $Y_{0:k} = \{(y_i, u_i),\ i = 0, ..., k\}$.

The solution to this problem is embedded in the conditional degree of belief in the state, x_k given the data, $Y_{0:k}$. The problem is then (recursively) to determine the pdf. $p(x_k|y_{0:k})$. If the initial distribution, $p(x_0)$, is known then the solution can in principle be determined through the recursions:

$$p(x_k|Y_{0:k-1}) = \int_{\Omega_x} p(x_k|x_{k-1})p(x_{k-1}|Y_{0:k-1})dx_{k-1} \tag{6}$$

and

$$p(x_k|Y_{0:k}) = \frac{p(y_k|x_k)}{p(y_k|Y_{0:k-1})}p(x_k|Y_{0:k-1}) \tag{7}$$

These two recursions are related to the dynamic ((6)) and the inference ((7)) step, respectively and can only in special cases be solved analytically. In the linear Gaussian case the pdf. can be parameterized in terms of mean and variance and the recursions results in the well known Kalman filter. In that case (the linear Gaussian case with standard assumptions including $x_0 \in \mathbf{N}(\hat{x}_0, P_0)$) the system is assumed to be given by the recursions:

$$x_{k+1} = Ax_k + Bu_k + v_k \qquad v_k \in \mathbf{N}_{iid}(0, R_1)$$

$$y_k = Cx_k + e_k \qquad e_k \in \mathbf{N}_{iid}(0, R_2)$$

The Kalman filter is given by the prediction or the time updates

$$\hat{x}_{k+1|k} = A\hat{x}_{k|k} + Bu_k \tag{8}$$

$$P_{k+1|k} = AP_{k|k}A^T + R_1 \tag{9}$$

and the inference recursion

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - \hat{y}_{k|k-1}) \tag{10}$$

$$P_{k|k} = P_{k|k-1} - K_kCP_{k|k-1} \tag{11}$$

where:

$$K_k = P_{k|k-1}C^T S_k^{-1}$$

and

$$\hat{y}_{k|k-1} = X\hat{x}_{k|k-1} \qquad S_k = CP_{k|k-1}C^T + R_2$$

In this case, the prediction in (6) results in (8) and can also be found as an application of calculus for linear operations on Gaussian vectors. The inference recursion in (10) emerge from (7) or as an application of the Projection Theorem.

The various filters differs in the way the handle the propagation of the distributions through the two nonlinearities, f and g, and how the inference is carried out. The next three filters are all based on the projection Theorem.

In this case, the prediction in (6) results in (8) and can also be found as an application of calculus for linear operations on Gaussian vectors. The inference recursion in (10) emerge from (7) or as an application of the Projection Theorem on:

$$\left[\begin{array}{c} x_k \\ y_k \end{array} \right] \Bigg| Y_{0:k-1} \in \mathbf{N}\left(\left[\begin{array}{c} m_x \\ m_y \end{array} \right], \left[\begin{array}{cc} P_x & P_{xy} \\ P_{yx} & P_y \end{array} \right] \right)$$

In this case

$$x_k | \, Y_{0:k} \in \mathbf{N}\left(\bar{m}_x, \bar{P}_x\right)$$

$$\bar{m}_x = m_x + P_{xy}P_y^{-1}(y_k - m_y) \qquad \bar{P}_x = P_x - P_{xy}P_y^{-1}P_{yx}$$

The connection to (10)-(11) is simply through

$$m_x = \hat{x}_{k|k-1} \qquad m_y = C\hat{x}_{k|k-1}$$

$$P_x = P_{k|k-1} \qquad P_{xy} = P_{k|k-1}C^T \qquad P_y = S_k$$

4.1 The Extended Kalman filter

The Extended Kalman filter is as its name indicate based on an extension of the application of the Kalman filter to the nonlinear case. The Extended Kalman filter (EKF) is based on a standard Taylor expansion of the nonlinear functions and can be regarded as a local approximation. In general the approximation is best for small deviations from the point of linearization.

The basic idea is related to the problem of determine the distribution of z if

$$z = F(x)$$

and the distribution of x is known to be $\mathbf{N}(\hat{x}, P_x)$. The approximation is simply to use

$$z \in \mathbf{N}\left(F(\hat{x}), AP_xA^T\right)$$

where

$$A = \frac{\partial}{\partial x}F\Big|_{\hat{x}}$$

This approximation applies both to the process equation (and f) and the measurement equation (and g). In fact, the only changes with respect to (8)-(11) is

$$\hat{x}_{i+1|i} = f_i(\hat{x}_{i|i}, u_i, 0) \qquad \hat{y}_{i|i} = g_i(\hat{x}_{i|i}, 0)$$

The variance update, (9) and (11), are unchanged (except for the state dependent system matrices).

4.2 Divided difference filters

The divided difference filter exists in a first order version (DD1) and in a second order version (DD2) and is based on Stirlings interpolation formula (see (Nørgaard, Poulsen & Ravn 2000) and Figure 3 for illustration).

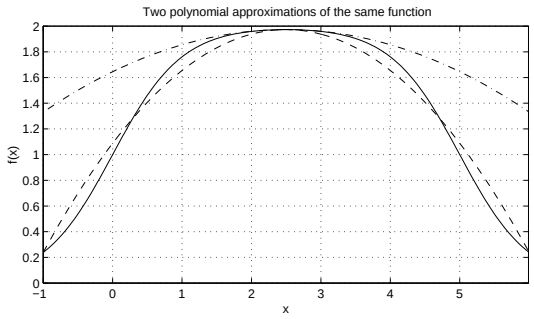

Figure 3. Comparison of a second-order polynomial approximation obtained with the Taylor (dot-dashed) and the Stirling method (dashed)

Let again, x be a stochastic variable and $x \in \mathbf{N}\left(\hat{x}, S_x S_x^T\right)$. The approximation which takes the variation of w into account is

$$F(x) = F(\hat{x}) + \bar{\nabla}_z F(\hat{x})(x - \hat{x}) + \frac{1}{2}\bar{\nabla}_x^2 F(\hat{x})(x - \hat{x})^2 + \varepsilon$$

where

$$\bar{\nabla}_x F(\hat{x}) = \mathbf{M}_{ij}\left\{\frac{1}{2h}\left[F_i(\hat{x} + hS_{xj}) - F_i(\hat{x} - hS_{xj})\right]\right\}$$

$$\bar{\nabla}_x^2 F(\hat{x}) = \mathbf{M}_{ij}\left\{\frac{1}{h^2}\left[F_i(\hat{x} + hS_{xj}) + F_i(\hat{x} - hS_{xj}) - 2F_i(\hat{x})\right]\right\}$$

Here h is a scale parameter and S_{xj} is the j'th column in S_x. In the Gaussian case the choice $h^2 = 3$ is in some sense optimal (see (Nørgaard et al. 2000)).

Introduce the notation

$$F_p^+ = F(\hat{x} + hS_{x,p}) \quad F_p^- = F(\hat{x} - hS_{x,p}) \quad F^0 = F(\hat{x})$$

For the DD2 filter the approximation is then

$$\hat{z} \;=\; \frac{h^2 - n_x}{h^2}F^0 + \frac{1}{2h^2}\sum_{p=1}^{n_x} F_p^+ + F_p^-$$

and

$$\begin{aligned} P_z &= \frac{1}{4h^2}\sum_{i=1}^{n_x}(F_p^+ - F_p^-)(F_p^+ - F_p^-)^T \\ &+ \frac{h^2-1}{4h^2}\sum_{i=1}^{n_x}(F_p^+ + F_p^- - 2F^0)(F_p^+ + F_p^- - 2F^0)^T \end{aligned}$$

For the first order filter (DD1) only the first terms in the approximations are used.

In the divided difference filters (DD1 and DD2) the propagation of mean and variance is determined through the approximations mentioned above. The inference is based on the Projection Theorem.

4.3 The Unscented kalman filter

The Uncented filter is based on the (uncentedi) transformation of a stochastic variable, x, through a nonlinear function, $F(x)$ (see (Julier & Uhlmann 2004)). Assuming again the mean of x is $\hat{x}$ and the variance matrix is $P_x = S_x S_x^T$, then the sigma points are defined as:

$$\begin{aligned} x^{(1)} &= \hat{x} & w_0 &= \frac{\kappa}{n_x+\kappa} \\ x^{(i)} &= \hat{x} + \sqrt{(n_x+\kappa)}S_{xi} & w_i &= \frac{\kappa}{2(n_x+\kappa)} \\ & & & i = 1,\ \ldots\ n_x \\ x^{(j+n_x)} &= \hat{x} - \sqrt{(n_x+\kappa)}S_{x,j} & w_{j+n_x} &= \frac{\kappa}{2(n_x+\kappa)} \\ & & & j = 1,\ \ldots\ , n_x \end{aligned}$$

Here κ is a scaling parameter and w_i is the weight associated with a point and

$$\sum_{i=0}^{2n_x} w_i = 1$$

Each sigma point is propagated through the nonlinear function

$$z^{(i)} = F(x^{(i)}) \qquad i = 0,\ \ldots\ 2n_x$$

and the approximation is then

$$\hat{z} = \sum_{i=0}^{2n_x} w_i z^{(i)}$$

and

$$P_z = \sum_{i=0}^{2n_x} w_i (z^{(i)} - \hat{z})(z^{(i)} - \hat{z})^T$$

The standard UKF is based on the approximation mentioned above and the Projection Theorem. In the scaled version of UKF the weight is chosen in a slightly different manner (see (Julier 2002) or (Wan & van der Merwe 2000) for details).

4.4 Particle filters

Particle filters comes in several versions and implementations (see e.g. (Arulampalam, Maskell, Gordon & Clapp 2002) or (van der Merwe, Doucet, de Freitas & Wan 2000)). In the most basic version (Exp. PF) implemented in the platform the nonlinearities are dealt with by propagating a swarm of particle through the nonlinearities. Again assuming $x \in \mathbf{N}(\hat{x}, P_x)$ a number (N) of particles are generated

$$x^{(i)} \leftarrow \mathbf{N}(\hat{x}, P_x) \qquad i = 1, \ ... \ N$$

and propagated through the nonlinear function

$$z^{(i)} = F(x^{(i)})$$

The approximation is then simply

$$\hat{z} = \sum_{i=1}^{N} z^{(i)} \qquad P_z = \sum_{i=1}^{N} (z^{(i)} - \hat{z})(z^{(i)} - \hat{z})^T$$

In the most basic version (Exp. PF) the inference is based on the Projection Theorem and the nonlinearities are handled with the method mentioned above.

In the generic particle filters (Gen. PF) the inference is not based on the Projection Theorem, but is carried out by applying 7 directly. That results in weights associated with each of the particles. In this version the particle are only initially generated as described above. After the inference step the particles are resampled from a distribution characterized by the weights. In the last version (MH. PF) implemented here on this platform, the resampling is performed by means of the Metropolis-Hastings algorithm.

5 Example study

The versatility of the simulation framework is most evident when implementing a number of examples. For the purpose of this demonstration, a discrete time difference equation system and a continuous time differential equation system are selected. The example studies concludes with a simultanous location and mapping problem.

5.1 Nonlinear state estimation

The first example i a discrete time system and is an academic example of a nonlinear system (though in a simplified form), which has been used previously as a benchmark for testing filter algorithms ((Netto, Gimeno & Mendes 1978)).

In the example the process is a nonlinear equation with a linear and noisy measurements. First, the process equation, x_{k+1} is listed, next the measurement

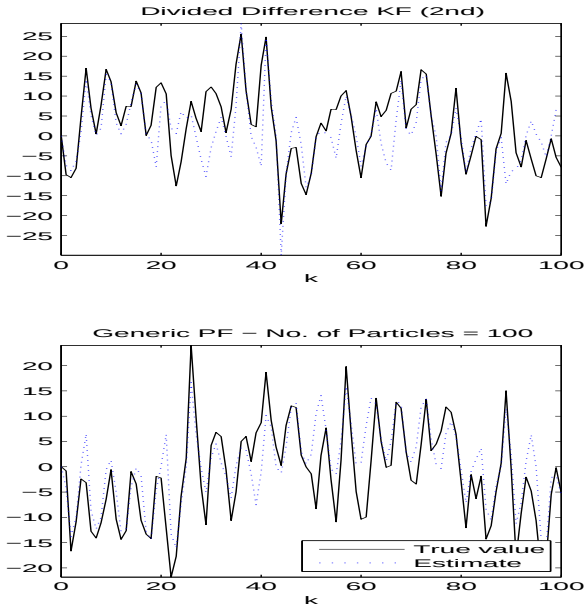

Figure 4. *Two examples of the highly nonlinear and noisy system given in equation 12. The topmost is the 2nd order Divided Difference filter, while the bottommost is a generic Particle Filter.*

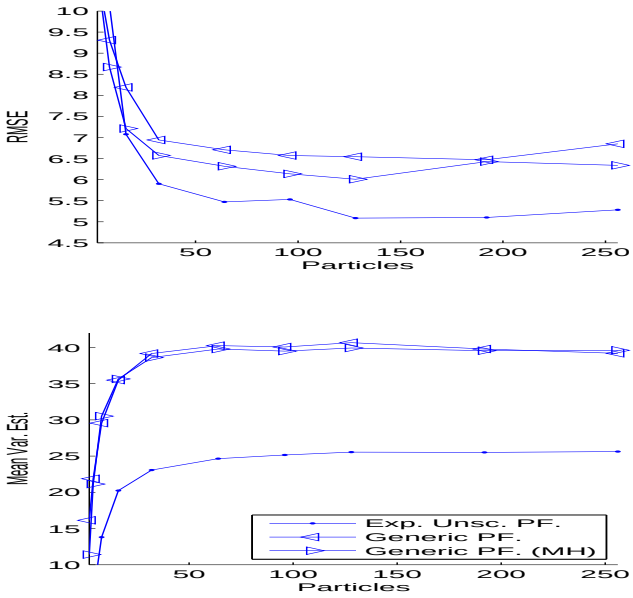

Figure 5. *Two graphs depicting the effect of varying the particle count per update on the benchmark system. The mean RMSE and mean variance estimates are seen to converge rather quickly to relatively stationary values at around 100 particles per time update.*

equation, y_k.

$$\begin{aligned} x_{k+1} &= \frac{1}{2}x_k + \frac{25x_k}{1+x_k^2} + 8\cos(1.2k) + v_n \\ y_k &= x_k + w_k; \end{aligned} \tag{12}$$

Note that, both the noise sources, v_k and w_k, are zero mean Gaussian white noise with variances of 10.0 and 1.0 respectively. As was the case with the small robot model, a Monte Carlo series of simulations was made with a variety of estimation algorithms. Two examples of the appearance of a simulation can be found in figure 4.

The result of the Monte Carlo simulation can be seen in table 1. The Kalman filter type algorithms were simulated 1000 times and the means of the root mean square errors (RMSE) were found as well as the means of the variance estimates. The Particle Filter types were simulated 100 times with 200 particles per time update in all filters.

Finally, in order to compare the precision of the three particle filters as a function of the number of particles per time update, a Monte Carlo series of simulations

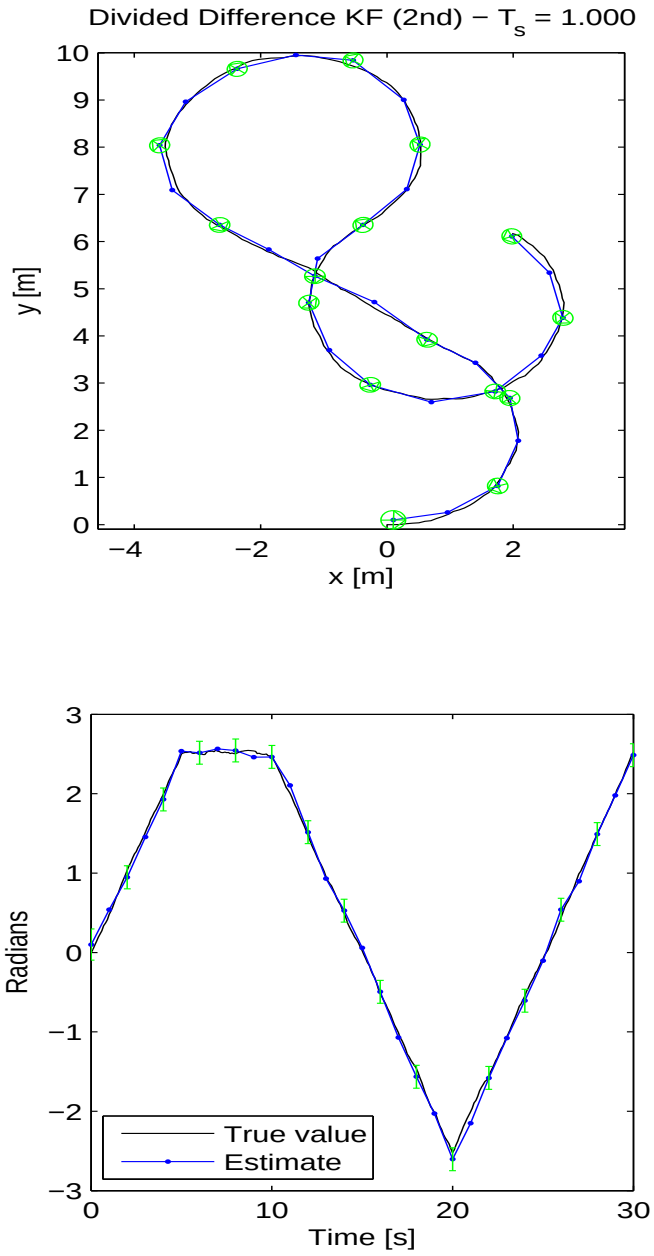

Figure 6. *A path traced by the small unicycle robot. The estimation routine employed is the 2nd order Divided Difference filter with a sampling frequency of 1 Hz. At every other estimated state, the 95% confidence intervals are drawn as ellipses or bars respectively. The estimate is at no point outside the confidence intervals.*

Algorithm	Mean RMSE	Mean Var. Est.	Time
C.D. EKF	0.9573	0.9206	1.000
Std. UKF	0.9472	0.9238	0.126
Scl. UKF	0.9503	0.9247	0.151
DD1	0.9417	0.9221	0.133
DD2	0.9260	0.9238	0.137
Exp. PF	0.9513	0.9165	2.067
Gen. PF	4.2326	31.595	5.917
PF (MH)	4.0238	28.165	8.543

Table 1. *Table of results for a Monte Carlo series of simulations on the discrete nonlinear and noisy system.*

was made using the benchmark system. The results can be seen in figure 5. The series consisted of 100 runs per particle count, from 2 to 256 particles in increasing steps. The algorithms converge rather quickly as the particle count increases.

5.2 Dead-reckoning

Algorithm	Max. RMSE	Max. Var. Est.	Time
C.D. EKF	0.06799	0.005717	1.000
Std. UKF	0.07399	0.006779	2.678
Scl. UKF	0.07331	0.006693	3.673
DD1	0.07251	0.006779	2.640
DD2	0.07177	0.006781	2.658
Exp. PF	0.07591	0.006479	16.86
Gen. PF	0.09698	0.039829	17.82
PF (MH)	0.08960	0.058690	18.53

Table 2. *Small mobile robot, worst value of mean estimate (x,y,θ) and maximum mean variance estimate of 100 Monte Carlo simulations. The table is split into Kalman filter variants (top) and particle filters (bottom). The particle filters all used 200 particles.*

The next example is a continuous time system and is a very simple model of a dead-reckoning guidance for a small mobile robot (see equation (1)). In Figure 6 the results of a simulation using the Divided Difference (2nd order) as estimator can be seen. The integral of the control signal, ω, is seen in the lower panel below the path traced by the robot (upper panel).

In order to compare a range of techniques implemented in the framework, accuracy results are given in table 2. Attempting to find a fair estimate of the accuracy, 100 runs were made with each algorithm and the average values were found. The particle filters all used 200 particles per time update. The table contains the "worst case" values for the three states.

Also listed in the table is the computational burden of each algorithm. The latter is given as a relative number compared to the runtime of a continuous-discrete extended Kalman filter (C.D. EKF). The times are relative, as other processor speeds and types will yield different absolute results. Furthermore, the algorithms and their runtimes may well benefit from numerical optimizations in application specific implementations. The algorithms used a fixed step integration (Matlab, Dormand-Prince, order 5) to solve equation 1. The standard Unscented Kalman filter (Std. UKF) performs very well, while it's scaled version gives a lower mean RMSE and a slightly lower mean variance estimates. The DD1 and DD2 both give low mean RMSE and consistent variance estimates - in this case, the second order parts of the DD2 does not yield much.

5.3 Simultanuous location and mapping I

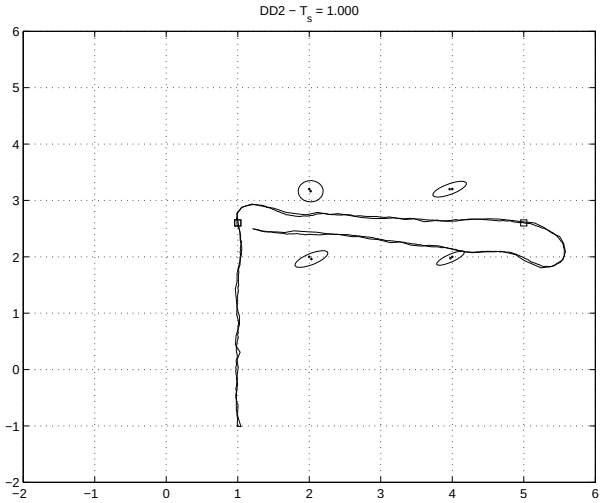

Figure 7. *Navigation of a mobile robot through a door opening. The map is build simultanuouly while controlling the robot. The control is based on the location i.e. the estimation of the position and orientation of the robot. The positions and their incertainty are anotated by 99% confidence areas (ellipsoids).*

The next two examples are related to location while a map of the guide marks is build. The dynamics involved is the AGV given in (1) with a sensor fusing between the odometry (dead-reckoning) and the relatiove postioning of the guide marks. Both the location and mapping is based on the oberservation equation, (2), where the guide mark is the actual guide mark under observation. The robot is assumed to have an active view sector in front which is 90 degree wide and has a range of 4 m. The active guide marks are the guide marks visible within the robot view sector.

In this context the map consists of a database containing the estimated locations of the guide marks and their respective uncertainty. Besides the database the location and mapping consists of a routine for handling the information related to the active guide marks.

In the first example related to simultaneous location and mapping the task is to navigate the robot along a wall and drive through the door opening and return. The door opening is defined in terms of two set of guide marks. The navigation is performed by means of way points located in in front and behind the door opening. The positions of the way points are assumed to be known. The control implementation is described in (Bak 2000), but is beyond the scope of this paper.

The results are illustrated in Figure 7 where the applied estimation technique is based on the DD2 method described in section 4.

5.4 Simultanuous location and mapping II

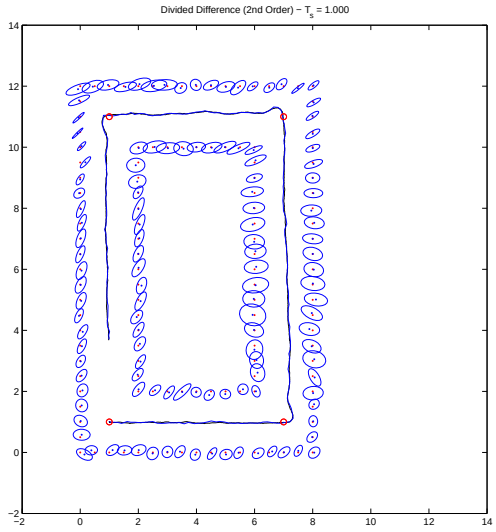

Figure 8. *Navigation of a mobile robot along a corridor with guide marks located on the walls. The map is build simultanuouly while controlling the robot.*

This example is quite similar to the previous example, except that in this case it is a bit more complex and the robot has to follow a coridor equipped with guide marks. The results can be seen in Figure 8. The true robot path (which is known due to the simulation) and the estimated are indicated by solid lines. The location of the 4 way points are also indicated.

As the map is build the position of the guide marks are introduced. The estimated positions and their uncertainties are indicated with a dot and a 99% confidence area (ellipsoids). Notice, that in some case the correct position of a guide mark is outside the confidence area.

6 Conclusion

In this paper we have presented a simulation platform for simultaneous location and mapping. The platform is an extention of the toolbox KALMTOOL ver. 2 which a set of MATLAB tools for state estimation for nonlinear systems. It contains functions for extended Kalman filtering as well as for the two new filters the DD1 filter and the DD2 filter. It also contains functions for Unscented (standard and scaled) Kalman filter as well as three versions of particle filters.

The paper contain a few examples to illustrate the methods and the results maninly based on divided difference approach (DD2) to nonlinear estimation.

In this work we have applied an earth fixed coordiante system in which both position (and orientation) of the robot and the guide marks are related. The result is positions of robot and guide marks in an absolute scale. However, the dynamic is related to the robot only. Another approach is to apply a robot fixed coordinate system. Then the position of the robot and guide marks are relative. In a robot fixed coordinate system the process equation for the guide marks are no longer the identity but a result of the movement of the robot (and the coordinate system).

Acknowledgment

The support from the Danish Center for Scientific Computing (DCSC) (under grant CPU-1101-30) is gratefully acknowledged.

References

Arras, K. O. (2004), The cas robot navigation toolbox, quick guide, Technical report, CAS, KTH.

Arulampalam, M., Maskell, S., Gordon, N. & Clapp, T. (2002), 'A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking', *IEEE Transactions on Signal Processing* **50**(2), 174–188.

Bak, M. (2000), Control of Systems with Constraints, PhD thesis, IAU, DTU.

Julier, S. (2002), 'The scaled unscented transformation', *Proceedings of the American Control Conference* pp. 4555–4559.

Julier, S. & Uhlmann, J. (2004), 'Unscented filtering and nonlinear estimation', *Proceeding of the IEEE* **92**(3), 401–422.

Netto, A., Gimeno, L. & Mendes, M. (1978), 'A new spline algorithm for non-linear filtering of discrete time systems', *Proceedings of the 4th IFAC Symposium on Identification and System Parameter Estimation, Tbilisi, U.S.S.R.* pp. 2123–2130.

Nørgaard, M., Poulsen, N. K. & Ravn, O. (2000), 'New development in state estimation for nonlinear systems', *Automatica* **36**, 1627–1638.

Nørgaard, M., Poulsen, N. K. & Ravn, O. (2003), Kalmtool for use with matlab, *in* '13th IFAC Symposium on System Identification, SYSID03, Rotterdam', IFAC, Rotterdam, pp. 1490–1495.

van der Merwe, R. (2004), Quick-start guide for rebel toolkit, Technical report, Oregon Health and Science University.

van der Merwe, R., Doucet, A., de Freitas, N. & Wan, E. (2000), The unscented particle filter, Technical report, Cambridge University Engineering Department.

Wan, E. & van der Merwe, R. (2000), The unscented kalman filter for nonlinear estimation, *in* 'Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium', pp. 153–158.