Derivative free Kalman filtering used for orchard navigation

Søren Hansen, **Enis Bayramoglu**,
Jens Christian Andersen, **Ole Ravn**,
Nils Axel Andersen
DTU Electrical Engineering
The Technical University of Denmark
sh, eba, jca, or, naa@elektro.dtu.dk

Niels Kjølstad Poulsen
DTU Informatics
The Technical University of Denmark
nkp@imm.dtu.dk

Abstract – *In this paper the use of derivative free filters for mobile robot localisation is investigated. Three different filters are tested on real life data from an autonomous tractor running in an orchard environment. The localisation algorithm fuses odometry and gyro measurements with line features representing the surrounding fruit trees. The line features are created on basis of 2D laser scanner data by a least square algorithm. The Matlab® toolbox Kalmtool is used for easy switching between different filter implementations without the need for changing the base structure of the system.*

Keywords: Robot navigation, State estimation, Autonomous mobile robots, Localisation, Sensor fusion.

1 Introduction

Autonomous navigation of mobile platforms solving agricultural tasks has a great potential. Especially in organised environments like orchards, where the same task has to be conducted many times through the season. The orchard environment is a relatively organised environment, which is ideal for autonomous robots like the automated HAKO tractor (figure 1). Trees are systematically planted, which makes laser navigation based on the surroundings interesting, since the dense vegetation of the rows of trees make a surface suitable for the laser scanner. Odometry, gyro and features extracted from the laser scanner can be fused to perform the localisation. An existing solution using an Extended Kalman filter (see [5]) for localisation and navigation has been developed for the HAKO tractor. In this paper an enhancement of this solution is investigated by using three derivative filters. The benefit is that instead of a local approximation of the nonlinear equations describing the robot and its sensors, a regional approximation is done and hence a smaller likelihood for errors is obtained. As a supplement the sometimes tedious work of finding Jacobians of complex functions are avoided.

When designing and building complex systems good tools are essential for success. A good tool support the user on the different levels of abstraction. Typically this ranges from mathematical formulation and simulation of the algorithms over numerical implementation to verification and validation of the actual device in real-time. The filter implementations outlined in this paper are designed using the tool, Kalmtool [6].

Kalmtool is a collection of MATLAB implementations for estimation and simulation in connection with nonlinear dynamic systems. The development of the toolbox has been driven by the application, which is navigation of mobile robots. In this context location and mapping are corner stones.

Figure 1: The HAKO tractor.

Several toolboxes have been proposed for mobile robot navigation over the years but a transparent and powerful tool is still in demand. ReBEL (Recursive Bayesian Estimation Library) [11] is a MATLAB toolkit of functions and scripts, designed to facilitate sequential Bayesian inference (estimation) in general state space models. The CAS Robot Navigation Toolbox [1] is a tool for doing off-line off-board localisation and SLAM for mobile robots. The design of the CAS toolbox decouples robot model, sensor models, features and algorithms used giving the user the possibility to

modify the toolbox. The OpenSLAM project [9] initiative, is a user driven website, which gathers algorithms and toolboxes.

The paper describes how three different derivative free filters can be used for the task of making the HAKO tractor drive autonomously through the orchard. The localisation solution uses the tree rows as measurements to correct the pose estimated by the filters. The paper is divided into the following parts. Firstly, the theory behind the Kalman filtering is outlined with focus on derivative free solutions. Secondly, the application of mobile robot localisation in an agricultural setting is discussed and the solutions are tested with real life data.

2 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

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

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 independent and identically distributed (i.i.d.) stochastic vectors.

The objective is to estimate x_k from measurements

$$y_k = g_k(x_k, e_k) \qquad \in \mathbb{R}^m \tag{2}$$

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 probability distribution function (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:

$$p(x_k|Y_{0:k-1}) = \int_{\Omega_x} p(x_k|x_{k-1})p(x_{k-1}|Y_{0:k-1})dx_{k-1} \tag{3}$$

and

$$p(x_k|Y_{0:k}) = \frac{p(y_k|x_k)}{p(y_k|Y_{0:k-1})}p(x_k|Y_{0:k-1}) \tag{4}$$

These two recursions are related to the dynamic ((3)) and the inference ((4)) step, respectively and can only in special cases be solved analytically. In the linear Gaussian case the pdf. can be parametrised 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

$$\hat{x}_{k+1|k} = A\hat{x}_{k|k} + Bu_k \tag{5}$$

$$P_{k+1|k} = AP_{k|k}A^T + R_1 \tag{6}$$

and the inference recursion

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - \hat{y}_{k|k-1}) \tag{7}$$

$$P_{k|k} = P_{k|k-1} - K_kCP_{k|k-1} \tag{8}$$

where:

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

and

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

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

In this case, the prediction in (3) results in (5) and can also be found as an application of calculus for linear operations on Gaussian vectors. The inference recursion in (7) emerge from (4) 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 (7)-(8) is simply through

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

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

2.1 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

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

This approximation applies both to the process equation (f) and the measurement equation (g). In fact, the only changes with respect to (5)-(8) 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, (6) and (8), are unchanged (except for the state dependent system matrices).

2.2 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 [7] and figure 2 for illustration).

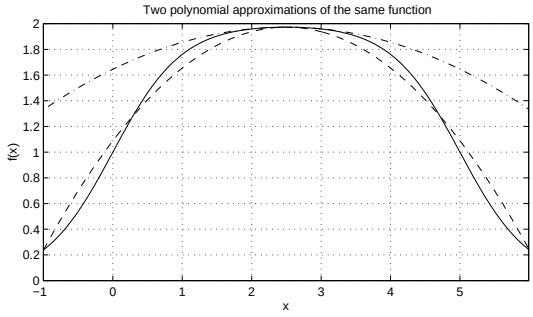

Figure 2: Comparison of a second-order polynomial approximation obtained with the Taylor (dot-dashed) and the Stirling method (dashed) of a non-quadratic function.

Let again, x be a stochastic variable and $x \in \mathbf{N}\left(\hat{x}, S_xS_x^T\right)$. 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

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

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

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 [7]).

Introduce the notation

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

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.

2.3 The Unscented Kalman filter

The Unscented Kalman filter is based on the (unscented) transformation of a stochastic variable, x, through a nonlinear function, $F(x)$ (see [3]). Assuming again the mean of x is $\hat{x}$ and the variance matrix is $P_x = S_xS_x^T$, then the sigma points are defined as:

$$\begin{aligned} x^{(0)} &= \hat{x} & w_0 &= \frac{\kappa}{n_x+\kappa} \\ x^{(i)} &= \hat{x} + \sqrt{(n_x+\kappa)}S_{x,i} & w_i &= \frac{1}{2(n_x+\kappa)} \\ & & & i = 1, \ldots n_x \\ x^{(j+n_x)} &= \hat{x} - \sqrt{(n_x+\kappa)}S_{x,j} & w_{j+n_x} &= \frac{1}{2(n_x+\kappa)} \\ & & & j = 1, \ldots, n_x \end{aligned}$$

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 [2] or [12] for details).

2.4 The linear regression Kalman filter

The linear regression Kalman filter (LRKF) is as the name indicated based on a linear regression (see [10]). The non linear function F is approximated by an affine function

$$F(x^{(i)}) = b + Ax^{(i)} + \varepsilon_i$$

in a least squares sense, i.e.

$$(\hat{A}, \hat{b}) = arg \min_{A,b} \sum_{i=1}^{n_x} \varepsilon_i^T \varepsilon_i$$

Furthermore the matrix

$$Q^* = \sum_{i=1}^{n_x} \varepsilon_i \varepsilon_i^T$$

is a measure of goodness of fit and can be added to the covariance matrices, R_1 and R_2 in the process equation and the measurement equation, respectively.

3 Autonomous orchard navigation

In order to make the robot navigate through the orchard based on measurements by the laser scanner, the raw scan data needs to be interpreted as features. In the orchard case line features are used. These features are matched with the corresponding features in a map. If this is possible one can use the difference between where a known feature is measured to be and where it should be according to a map, to update the position estimate of the robot. Since the robot is driving in an orchard environment, it is natural to use trees and hedges as features.

In the orchard experiments the HAKO tractor - see figure 1 is used. The tractors sensors are described in detail, together with the models used in the filters, in the subsections of this chapter.

3.1 HAKO tractor

The HAKO tractor has the following sensors attached in the setup used for orchard navigation.

- **RTK-GPS**
  A high precision RTK-GPS receiver logs the position in UTM coordinates. This is used as the ground truth in comparison with the localisation algorithm. The GPS system delivers an accuracy of $1 \text{ cm} + 2 \text{ ppm}$ in the horizontal direction and $2 \text{ cm} + 2 \text{ ppm}$ in the vertical direction, according to the manufacturer.

- **Odometry**
  A quadrature encoder is attached to the tractors drive shaft. This measures the revolutions of the shaft and thereby a measure of the length driven by the tractor is found. In addition the angle of the front wheels are measured. By using these measurements the tractors position can be found by odometry.

- **Gyro**
  A fibre optic gyro measuring the rate of the tractors heading is also attached.

- **Laser scanner**
  The laser scanner attached is a SICK LMS-200 and it has a $180°$ degree field of view, with a $0.5°$ resolution. The scanner is placed at the front of tractor 0.5 m over ground.

3.2 Map of orchard

In order to relate measurements from the laser scanner to tree rows in the orchard, a map is needed. A simple map which contains starting and ending points of rows is used. The map is formed in the UTM coordinate system. From the map information a straight line representation ($Ax + By + C = 0$) of the orchard rows can be found easy. The fruit tree rows are mapped as well as two boundary hedges, one in the southern region and one to the east.

3.3 Localisation

The localisation algorithm is based on the state estimate $\mathbf{x}_k$ and odometry measurement $\mathbf{u}_k$ given by

$$\mathbf{x}_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} \qquad \mathbf{u}_k = \begin{bmatrix} L_k \\ \theta_{SA,k} \end{bmatrix} \tag{9}$$

where x_k, y_k and θ_k is the robot pose in the global coordinate system. The control signals L_k and $\theta_{SA,k}$ is the distance driven in one sample and the tractors steering angle. The state- and measurement transition is done by a pair of non-linear stochastic difference equations like equation 1 and equation 2 on page 2.

3.4 Vehicle model

The model used to derive the tractors odometry is called tricycle drive and is seen in further detail on figure 3. This is a simplification of the Ackermann steering geometry used on the HAKO tractor. Odometry is the method of calculating the position of a vehicle from heading and speed changes. On the HAKO tractor, the speed is measured using an encoder on the gearbox. The heading is measured by a gyro and can also be calculated from the steeringangle, which is measured by an absolute encoder, returning the steering angle.

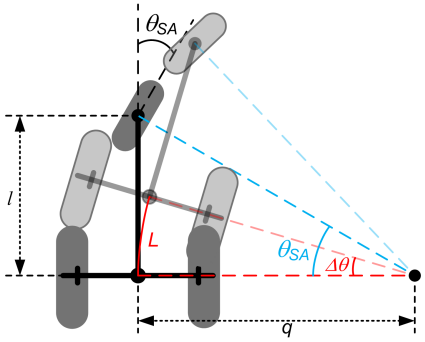

Figure 3: Figure showing the change in heading $\Delta\theta$ when moving the distance L with the steering angle θ_{SA}.

On the robot the steering angle θ_{SA} is given. The change in heading angle is derived from figure 3. The equation is:

$$\Delta\theta = \frac{L_k \tan(\theta_{SA})}{l} \tag{10}$$

where L_k is the distance the robot has driven in one sample and l is the distance between the front and rear axle of the HAKO tractor.

The robots new position and angle can now be found as:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \begin{bmatrix} L_k \cos\left(\theta_k + \frac{\Delta\theta}{2}\right) \\ L_k \sin\left(\theta_k + \frac{\Delta\theta}{2}\right) \\ \Delta\theta \end{bmatrix} \tag{11}$$

where the process noise is added through the control signals L_k and θ_k as described in the next section.

3.5 Process noise model

The process noise is the noise originating from the measurement of the control signals - in this case - the odometry. The process noise is the covariance of the control signal $\mathbf{u}_k$ and is denoted $\mathbf{Q}_k$.

$$\mathbf{Q}_k = \begin{bmatrix} \sigma_L^2 & 0 \\ 0 & \sigma_{\theta,SA}^2 \end{bmatrix} \tag{12}$$

For the error covariance to be dependent of the travelled distance, the variances must be proportional to the travelled distance.

The variance for the linear displacement is given by [4]:

$$\sigma_L^2 = \epsilon_L^2 |L| \tag{13}$$

where ϵ_L is the standard deviation error from one meter of travel.

The variance of the heading θ is found using equation 14 below. The equation is based on equation 10, which determines the change in direction.

$$\sigma_{\theta,SA}^2 = \left(\frac{\tan(\epsilon_A + \epsilon_{SA}|\theta_{SA}|)}{l}\right)^2 |L| \tag{14}$$

ϵ_A is the angle error standard deviation caused by driving one meter straight ahead and ϵ_{SA} is the contribution from the steering angle while turning.

3.6 Feature extraction

In this section the feature extraction will be presented. The SICK laser scanner returns 361 different range measurements spread out in a $180°$ fan originated in the centre of the scanner. The principle is illustrated in figure 4

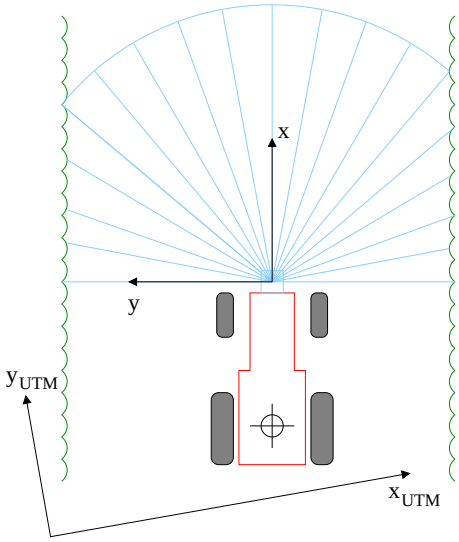

Figure 4: The HAKO tractor driving between rows in the orchard. The laser scanner returns distances from the origo of the $\{x, y\}$ coordinate system to the tree rows. The maximal distance measured is 8.1 m.

By using knowledge about the tractors pose the range scans are transformed into global UTM coordinates. These laser measurements are pre-processed by calculating if the numeric distance to the predicted line, based on the tractor position and the map. If the distance is below a pre-defined threshold, the measurement is marked (see figure 5) and used.

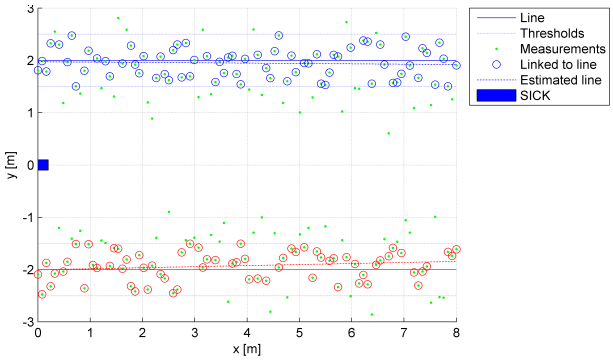

Figure 5: Principle of matching scan data to a priori map to estimate line from measurements.

The distance d_l between the laser scan range measurement point-representation and predicted line is calculated using equation 15.

$$d_l = \frac{Ax_{UTM} + By_{UTM} + C}{\sqrt{A^2 + B^2}} \tag{15}$$

where the equation involving A, B, C is the normal equation of the a priori line and x_{UTM}, y_{UTM} is the range measurement point-representation.

All the measurements which are inside the interval on either side of the predicted line are now used, to find an estimate of the measured line using linear regression. For the random test data this is visualised on figure 5. When the line is found further checks are performed to see if the line estimate is sufficiently accurate to be used by the EKF.

3.7 Line features

A laser scan usually contains several line features, and all of them are passed on for further processing. The description below concerns directly the lines extracted with a priori knowledge. To simplify the Kalman filter the features are represented in polar coordinates.

Feature vector
The features used are the rows that make up the orchard. If the distance from the centre of the robot to the row is denoted by d_m and the bearing of the row in UTM coordinates is denoted θ_m the feature vector is given by:

$$f(z_k) = f_k^1, f_k^2, ... = \left\{ \begin{bmatrix} d_{m,k}^1 \\ \theta_{m,k}^1 \end{bmatrix}, \begin{bmatrix} d_{m,k}^2 \\ \theta_{m,k}^2 \end{bmatrix}, \dots \right\} \tag{16}$$

Figure 6 shows the definition of the distance and angle measured.

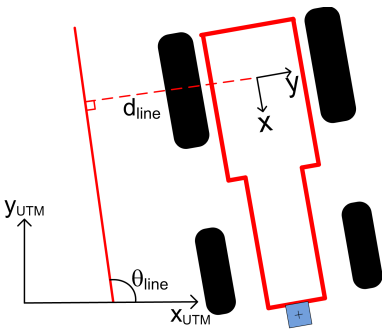

Figure 6: Measured line shown in robot and UTM coordinate system.

Since the rows found are represented as $A_{m,i}x_{UTM} + B_{m,i}y_{UTM} + C_{m,i} = 0$ initially the distance and angle of each feature is found as:

$$\begin{bmatrix} d_m \\ \theta_m \end{bmatrix} = \begin{bmatrix} \frac{A_m x_{UTM} + B_m y_{UTM} + C_m}{\sqrt{A_m^2 + B_m^2}} \\ \arctan\left(\frac{-A_m}{B_m}\right) - \theta \end{bmatrix} \tag{17}$$

The θ_m is subtracted with the state estimate of the angle θ to relate the line to the current state. Since the same is done to the measurement vector, this does not result in a difference in the update vector.

4 Results

The results presented in this section are based on recorded data from the HAKO tractor, during an autonomous drive in the orchard. The log is spanning a tour of four rows (see figure 7), which makes is possible to show the position update when re-finding the rows at headland turns. The data shown is recorded while the tractor runs autonomously. The green 'maplines' shown on the figure are the straightline representation of the orchard tree rows used as for position update.

Three derivative-free filters have been tested for the localisation task. The RTK-GPS measurements are used as ground truth for the position estimates, which all are kept in a global reference frame for easy comparison. The tested filters are

- Second order divided difference filter (dd2).
- Linear regression filter (linreg).
- Unscented Kalman filter (ukf).

All three filters are able to complete the run through the orchard in a satisfactory way. They are able to keep the robot in track and close to the ground truth. In figure 7 the full run of the orchard is shown for the dd2 filter. The robot starts in the upper left corner and ends in the upper right corner of the run. In the end it is seen that the odometry estimate has drifted due to wheel slippage and therefore ends about 5 meters from true robot position. However the state estimate is only about 20 cm from the correct end position.

This is seen better in the close up in figure 8 where the robots end position is shown. The end position estimate of the dd2 and the linear regression filters are almost identical (8 cm apart) hence the dd2 is not visible in the figure.

Figure 9 shows the time development of the standard deviations of the states, estimated by the three filters under similar noise initialisations. They show remarkable similarity. The deviation on the estimate of y, σ_y, is raising except in the lower corners around sample 0.9×10^4 and sample 2.8×10^4 where the robot is in the most southern part of the orchard. At these two points a tree row perpendicular to the others ensure that measurements in the y-direction is possible (this is marked as the lowest green line in figure 7).

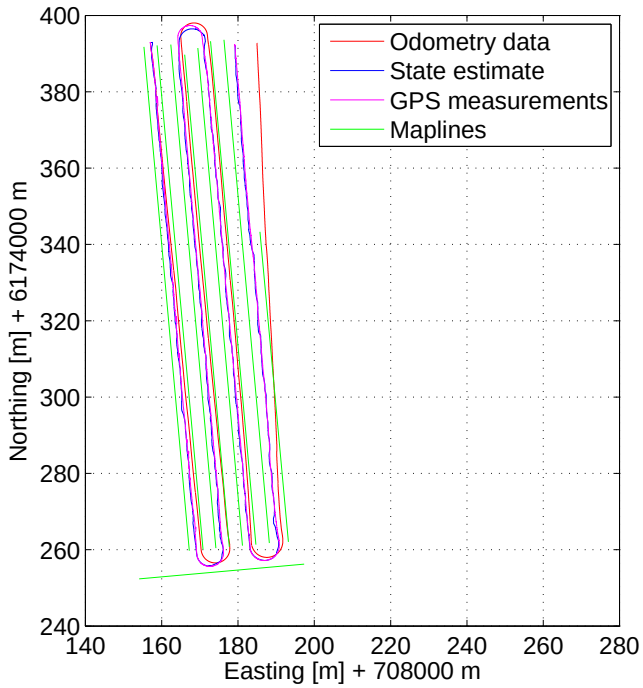

Figure 7: Plot of the full 4-row run through the orchard. The state is estimated by the dd2 filter.

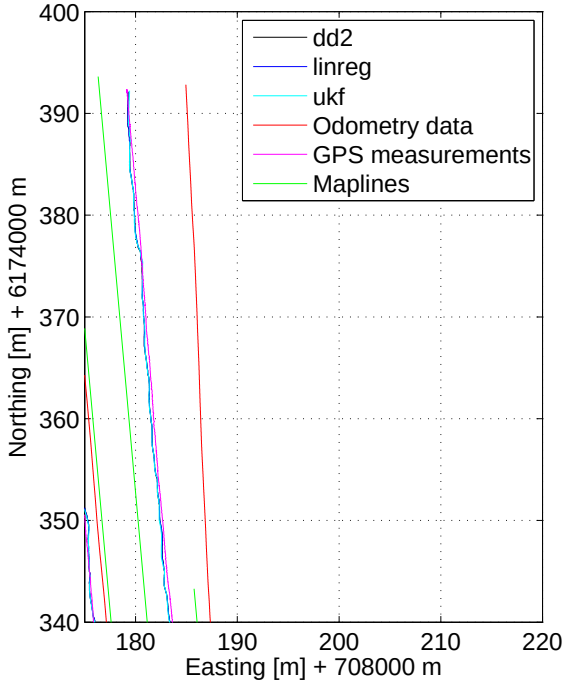

Figure 8: Plot of the end position measured and estimated by the three filters.

In table 1 the average computational time for the three filters is listed. This is the time it takes each filter to iterate through 2500 samples corresponding to a 50 meters drive with two

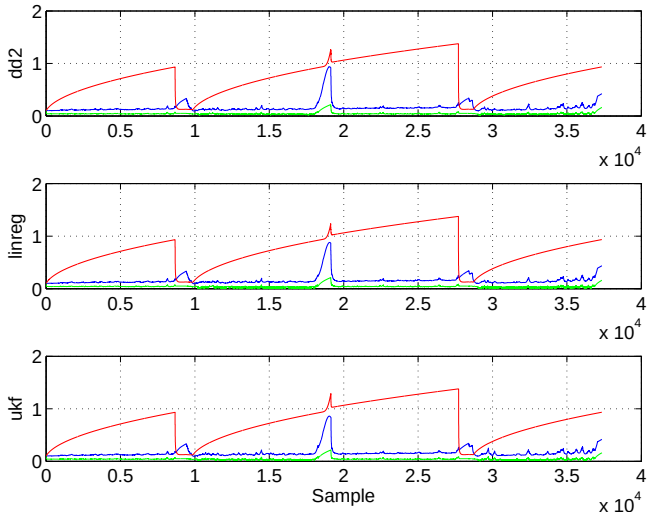

Figure 9: Square root of covariances of the three estimated states for the filters.
Legend: σ_x Blue σ_y Red σ_θ Green

Filter type	Avr. CPU time	Sum of GT. distance
dd2	74.7825 s	154.008
linreg	74.5950 s	151.700
ukf	75.9567 s	163.098

Table 1: Average computational time and sum of distance deviation from ground truth.

rows visible by the laser scanner. The time is an average over 50 experiments. The filters use almost the same amount of time with the ukf being marginally slower. In comparison an Extended Kalman filter uses in average 71 s for the same run. This shows that the different filter algorithms are not the main CPU time consumer. It is rather the data processing involved with map look-up and feature matching that takes time. This is identical for all three implementations, and the difference in time shown in the table stems from the filter algorithms.

In the last column of table 1 the sum of the distance deviation between the state estimate and the ground truth for the complete orchard run is shown. The Unscented Kalman filter has again the highest deviation. In figure 10 the position estimate of the ukf is compared with the linear regression filter. Both filters distort the turn compared to the GPS. However, the linear regression filter estimates a trajectory closer to the GPS.

5 Conclusion

A localisation solution using three derivative free filters has been implemented, enabling the HAKO tractor to drive autonomously through the orchard using laser scanner features to correct the pose estimate. The three filters are the sec-

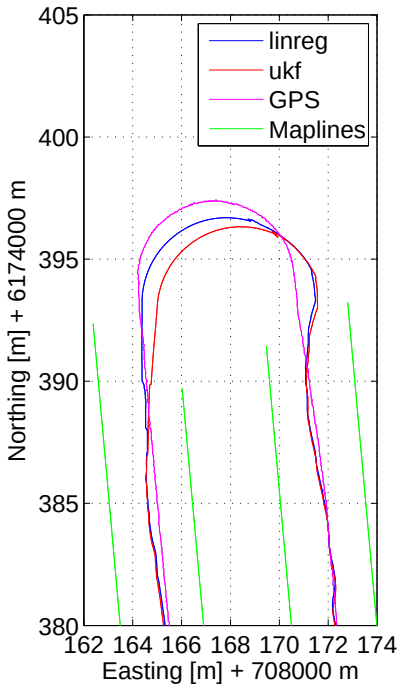

Figure 10: Comparison between Unscented Kalman filter and linear regression filter at the northern turn of the orchard.

ond order divided difference filter, the Linear Regression Kalman filter and the Unscented Kalman filter. All three filters performs satisfactory both with regard to precision and computational time, however the Unscented Kalman filter seems to perform marginally inferior compared to the others.
The new filters matches an existing solution using an Extended Kalman filter in performance, but utilises better approximations and is more flexible towards changes in the system and measurement model descriptions.

Using the Kalmtool toolbox makes the comparison between filters easy and the visualisation features makes the debugging less tedious.

The toolbox is available for download at: *http://server.elektro.dtu.dk/personal/or/kalmtool/*

Acknowledgement

Copenhagen University, Faculty of Life Sciences is gratefully acknowledged for cooperation and use of the HAKO tractor for implementation and data collection. Especially the research group under Associate Professor Hans-Werner Griepentrog which is very involved in the ongoing cooperation.

References

[1] Kai O. Arras. The cas robot navigation toolbox, quick guide. Technical report, CAS, KTH, January 2004.

[2] S. Julier. The scaled unscented transformation. *Proceedings of the American Control Conference*, pages 4555–4559, May 2002.

[3] Simon Julier and Jeffrey Uhlmann. Unscented filtering and nonlinear estimation. *Proceeding of the IEEE*, 92(3):401–422, 2004.

[4] Lindsay Kleeman. Advanced sonar and odometry error modeling for simultaneos localisation and map building. *Proceedings of the 2003 IEEE International Conference on Intelligent Robots and System, Las Vegas*, 2003.

[5] L. V. Mogensen, S. Hansen, J. C. Andersen, O. Ravn, N. A. Andersen, M. Blanke, and N. K. Poulsen. Kalmtool used for laser scanner aided navigation. In *15th IFAC Symposium on System Identification (SYSID)*, jul 2009.

[6] L. V. Mogensen, S. Hansen, O. Ravn, and N. K. Poulsen. Comparing mobile robot localisation algorithms using kalmtool. In *15th IFAC Symposium on System Identification (SYSID)*, jul 2009.

[7] Magnus Nørgaard, Niels K. Poulsen, and Ole Ravn. New development in state estimation for nonlinear systems. *Automatica*, 36:1627–1638, 2000.

[8] Magnus Nørgaard, Niels Kjølstad Poulsen, and Ole Ravn. Kalmtool for use with matlab. In *13th IFAC Symposium on System Identification, SYSID03, Rotterdam*, pages 1490–1495, Rotterdam, 2003. IFAC.

[9] Cyrill Stachniss, Udo Frese, and Giorgio Grisetti. Open slam - give your algorithm to the community. Web site - www.openslam.org, October 2008. Authors assosiated with University of Freiburg.

[10] Herman Bruyninckx Tine Lefevbre and Joris De Schutter. Kalman filters for non-linear systems: a comparison of performance. *International Journal of Control*, 77(7):639–653, 2004.

[11] Rudolph van der Merwe. Quick-start guide for rebel toolkit. Technical report, Oregon Health and Science University, February 2004.

[12] Eric Wan and Rudolph van der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium*, pages 153–158, 2000.