Computers and Electronics in Agriculture 157 (2019) 261–269

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag

Original papers

Robot navigation in orchards with localization based on Particle filter and Kalman filter

Pieter M. Blok[a,b,*], Koen van Boheemen[a], Frits K. van Evert[a], Joris IJsselmuiden[b], Gook-Hwan Kim[c]

[a] Wageningen University & Research, Agrosystems Research, Droevendaalsesteeg 1, 6708 PB Wageningen, the Netherlands
[b] Wageningen University & Research, Farm Technology Group, Droevendaalsesteeg 1, 6708 PB Wageningen, the Netherlands
[c] Rural Development Administration, National Academy of Agricultural Science, 166 Nongsaengmyeong-ro, Wanju-gun, Jeollabuk-do 55365, Republic of Korea

ARTICLE INFO

Keywords:
Probabilistic localization
Autonomous robot navigation
Particle filter
Kalman filter
Orchard

ABSTRACT

Fruit production in orchards currently relies on high labor inputs. Concerns arising from the increasing labor cost and shortage of labor can be mitigated by the availability of an autonomous orchard robot. A core feature for every mobile orchard robot is autonomous navigation, which depends on sensor-based robot localization in the orchard environment. This research validated the applicability of two probabilistic localization algorithms that used a 2D LIDAR scanner for in-row robot navigation in orchards. The first localization algorithm was a Particle filter (PF) with a laser beam model, and the second was a Kalman filter (KF) with a line-detection algorithm. We evaluated the performance of the two algorithms when autonomously navigating a robot in a commercial Dutch apple orchard. Two experiments were executed to assess the navigation performance of the two algorithms under comparable conditions. The first experiment assessed the navigation accuracy, whereas the second experiment tested the algorithms' robustness. In the first experiment, when the robot was driven with 0.25 m/s the root mean square error (RMSE) of the lateral deviation was 0.055 m with the PF algorithm and 0.087 m with the KF algorithm. At 0.50 m/s, the RMSE was 0.062 m with the PF algorithm and 0.091 m with the KF algorithm. In addition, with the PF the lateral deviations were equally distributed to both sides of the optimal navigation line, whereas with the KF the robot tended to navigate to the left of the optimal line. The second experiment tested the algorithms' robustness to cope with missing trees in six different tree row patterns. The PF had a lower RMSE of the lateral deviation in five tree patterns. In three out of the six patterns, navigation with the KF led to lateral deviations that were biased to the left of the optimal line. The angular deviations of the PF and the KF were in the same range in both experiments. From the results, we conclude that a PF with laser beam model is to be preferred over a line-based KF for the in-row navigation of an autonomous orchard robot.

1. Introduction

Each year, more than 675 million metric tons of fruit are produced worldwide (Statista, 2018). The production of fruit is typically labor intensive, as it depends on several manual tasks, such as crop maintenance and selective harvest. Unfortunately, increasing labor costs and shortage of labor cause a pressure on the supply of labor in orchards. It is expected that this labor problem will only worsen in the future, due to increased urbanization and lack of farm successors. To overcome these problems, many research efforts have been devoted to the development of robotic systems in fruit orchards. Agricultural robots have the potential to replace human labor in times of labor scarcity. A core feature for every mobile orchard robot is autonomous navigation, which comprises the safe and autonomous guidance of the robot in the orchard environment. In turn, navigation depends on autonomous localization, which is the process of determining the robot's position and orientation (pose) in the fruit orchard using sensors and software algorithms. More specifically, the data from the sensors is used by the software algorithms to provide information on the robot's location with respect to the surrounding environment. Based on this information, the robot can autonomously navigate between the tree rows to execute autonomous tasks in the orchard.

To our knowledge, there are at least four autonomous orchard robots commercially available at this moment. The autonomous

* Corresponding author at: Wageningen University & Research, Agrosystems Research, Droevendaalsesteeg 1, 6708 PB Wageningen, the Netherlands.
E-mail addresses: pieter.blok@wur.nl (P.M. Blok), koen.vanboheemen@wur.nl (K. van Boheemen), frits.vanevert@wur.nl (F.K. van Evert), joris@track32.nl (J. IJsselmuiden), meceng93@korea.k (G.-H. Kim).

https://doi.org/10.1016/j.compag.2018.12.046
Received 16 October 2018; Received in revised form 18 December 2018; Accepted 23 December 2018

navigation of two of these robots fully depends on Global Navigation Satellite System (GNSS) receivers (Greenbot, 2018; Precision-Makers, 2018), whereas the other two robots use a combination of different sensors. These two robots both use a GNSS receiver and a Laser Imaging Detection And Ranging (LIDAR) scanner for autonomous navigation (ASI-Robots, 2018; NAIO-Technologies, 2018); one of these two robots additionally uses a camera (NAIO-Technologies, 2018). When focusing on the applicability of these sensors for autonomous localization in orchard environments, Li et al. (2009) stated that robot localization based on GNSS receivers is susceptible to operation failure when the GNSS signals are blocked by the tree canopy. Vision-based robot localization using cameras proved to be an alternative, however Shalal et al. (2013) highlighted that varying outdoor light conditions, for example direct sunlight or shadow, had a negative impact on the navigation performance. LIDAR scanners proved to be more robust in outdoor environments (Weiss and Biber, 2011) and are considered as a predominant sensor for robot localization in orchards (Shalal et al., 2013). A LIDAR scanner emits laser beams from several scan angles and measures the time of flight for each beam to return after being reflected by an object. Supplementary sensors, such as inertial measurement units (IMUs), gyroscopes and wheel encoders are often integrated in mobile orchard robots to track the relative changes of the robot's pose.

The vast majority of the localization algorithms presented in analogous research, were based on data obtained from LIDAR scanners (Andersen et al., 2010; Barawid et al., 2007; Bergerman et al., 2012; Blok et al., 2018; Freitas et al., 2012; Hansen et al., 2009; Hiremath et al., 2014; Jæger-Hansen et al., 2012; Libby and Kantor, 2010; Marden and Whitty, 2014; Shalal et al., 2015; Zhang et al., 2014). The presented methodologies differed in the way they processed the LIDAR data. Several algorithms were based on line-detection methods that estimated a row of trees by fitting a line through the observed laser scan points. Using the position and orientation of the line that resembled the tree row, an estimation was made on the robot's pose with respect to this line. Commonly used line-detection algorithms for robot navigation in orchards are the Hough Transform (Barawid et al., 2007), Random Sample Consensus (RANSAC) (Marden and Whitty, 2014) and least squares line fitting (Andersen et al., 2010; Bergerman et al., 2015). However, Hiremath (2013) indicated that robot navigation solely based on these line-detection algorithms could be negatively influenced by dynamic and unpredictable situations, for instance overhanging tree branches, debris, fixed obstacles (fruit crates) and moving obstacles (animals or human personnel). Another methodology combines simultaneous localization and mapping (SLAM) within an unknown environment. LIDAR-based SLAM is a promising method for outdoor localization (Christiansen et al., 2011; Lepej and Rakun, 2016), however SLAM requires more processing time and computing requirements (Shalal et al., 2015). In addition, Chen et al. (2018) stated that LIDAR-based SLAM is less reliable when the laser beams returned from tree branches, canopies or other objects that change throughout the growing season.

Another approach, based on probabilistic calculus, proved to be more suitable for the use in dynamic and unpredictable situations (Thrun et al., 2005). Instead of relying on a single estimation, probabilistic algorithms represent information by probability distributions with multiple hypotheses. As such, these algorithms tend to be more robust to deal with sensor limitations and sensor noise in dynamic environments, such as orchards. Two commonly used probabilistic algorithms for robot localization in orchards are the Kalman filter (KF) (Andersen et al., 2010; Bergerman et al., 2015; Hansen et al., 2011; Shalal et al., 2015) and the Particle filter (PF) (Bergerman et al., 2012; Blok et al., 2018; Kurashiki et al., 2010). The first uses Gaussian distributions for its localization, while the second uses random sampling with multiple particles to estimate the robot's pose. Shalal et al. (2013) stated that the KF is the predominant algorithm for robot localization in row-crops and orchards. In addition, we found that the majority of the presented KF algorithms used line-detection algorithms in their measurement update step. These line-based algorithms use feature-extraction to generate a few lines from hundreds of laser scan points, and thereby drastically reduce the amount of data used for robot localization. Another localization method assessed individual laser beams by combining an environment model and a laser beam model in a PF (Blok et al., 2018; Hiremath et al., 2014). This probabilistic approach allowed extensive data analysis for robot localization, as the two models incorporated the interaction of individual laser beams with the environment. However, this PF algorithm required more a-priori information about the orchard environment and was more computational intensive than a line-based KF (Hiremath, 2013).

Despite previous efforts, research to date did not validate the different probabilistic localization algorithms under comparable situations in a commercial orchard. The objective of this paper was to identify the most applicable probabilistic localization algorithm for in-row robot navigation in orchards. Based on the findings of Blok et al. (2018) and Hiremath et al. (2014), we hypothesized that a PF with laser beam model outperformed a line-based KF for robot navigation due to the higher degree of a-priori information and the extensive data analysis. The aim of this research was to evaluate the navigation accuracy and robustness of the two probabilistic localization algorithms when autonomously navigating a robot in a commercial fruit orchard. As such, this research contributes a novel validation of two probabilistic localization algorithms under comparable circumstances to allow accurate and robust robot navigation in orchards. In addition, our research proposes a new method to evaluate the navigation robustness of robots in real-world orchard conditions. This research focused on the in-row navigation of an orchard robot and did not investigate autonomous turning and the detection of obstacles and headlands.

2. Materials and methods

2.1. Robot platform

An open-source software controlled robot (Husky A200, Clearpath Robotics, Kitchener, Canada) was used as the robot platform to test the autonomous robot navigation in a commercial orchard [Fig. 1]. The robot consisted of a rigid frame with four fixed wheels at a wheel base of 0.54 m and a track width of 0.67 m. The robot used skid-steering for turning. Three sensors were integrated with the robot to provide information about the robot's state and the environment. The first was a 2D LIDAR scanner (LMS-111, Sick AG, Waldkirch, Germany). This scanner had a field of view of 270° with an angular resolution of 0.50° and yielded 541 measurements, all updated at 50 Hz. Each measurement consisted of the travelled distance of a laser beam when returned by an object and the intensity of the returned beam. The latter is influenced by the distance and the reflectance properties of the object. The systematic error in the distance measurement was approximately

Fig. 1. Husky A200 robot with 2D LIDAR scanner used for orchard navigation.

0.03 m (Sick Sensor Intelligence, 2016). The LIDAR scanner was placed in a horizontal direction on the robot's front at 0.50 m above the ground. We limited the scanner's detection range up to 4.0 m to be able to observe only one tree row on each side of the robot. The second sensor was an IMU (UM6, CH Robotics, Box Hill North, Australia). This sensor outputted the roll and pitch with a 2° accuracy and the yaw with 5° accuracy, all updated at 20 Hz (CH, 2013). The last sensors used for navigation were the on-board wheel encoders that measured wheel odometry. The encoders had a resolution of 78.000 revolutions per travelled meter (Clearpath Robotics, 2016).

In addition to these sensors, we equipped the robot with a Real Time Kinematic (RTK) GNSS receiver (HiPer Pro, Topcon, Tokyo, Japan) to obtain ground truth position information. The RTK-GNSS receiver was placed in the middle of the robot at 0.50 m above the ground. The receiver generated a horizontal position estimate with a 0.02 m accuracy updated at 2 Hz. A custom built computer was placed in the loading bay of the robot. This computer consisted of an Intel Core i7-3770 T CPU with 8 GB of DDR3 RAM. A pre-configured version of Ubuntu Linux 14.04LTS from Clearpath Robotics was installed on the computer, together with the Robot Operating System (ROS). An object-oriented software program was built in Python to process the sensor data, get the ground truth measurements, run the localization algorithms and to control the robot.

2.2. Particle filter algorithm with laser beam model

A Particle filter (PF) algorithm was developed to estimate the robot's pose with respect to the tree rows. The centerline that was midway between two tree rows was used as optimal navigation line. Two pose variables were used in the state belief of the robot [Fig. 2]. The first variable was the lateral deviation (d), an estimate of the positional deviation perpendicular to the centerline. This lateral deviation was negative when the robot was left of the centerline and positive when the robot was right of the centerline. The second variable was the deviation in orientation (α), which was defined as the difference between the robot's direction of travel (heading) and the orientation of the two tree rows. A negative angular deviation was observed when the robot was oriented left of the centerline and positive when oriented right of the centerline. Two environment variables were used in the robot's state

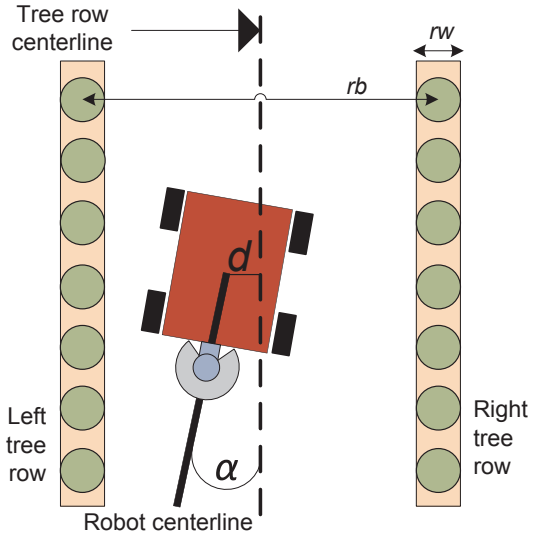

Fig. 2. Graphical representation of the robot's state belief for in-row localization. The circles represent the tree trunks and the sensor in front of the robot represents the LIDAR scanner. d depicts the robot's lateral deviation to the centerline, while α is the robot's angular deviation from the heading of the centerline. The environment parameters of the PF were the tree row width (rw) and the inter-row distance between two tree rows (rb).

belief to provide the PF algorithm some flexibility to cope with changing conditions in the orchard. The first environment variable was the inter-row distance between two trees rows (rb), and the second variable the tree row width (rw) [Table 1].

At algorithm initialization, 125 particles were randomly generated. These particles represented randomized and discrete simulations of the four state variables. With these randomized simulations, the PF algorithm generated multiple hypotheses about the possible pose of the robot and the orchard environment. Then, the PF algorithm estimated the robot's pose by two sequential repetitive steps: the predict step and the measurement update step. In the predict step, the previous beliefs for the angular and lateral deviation were updated using the IMU and wheel encoder data (step 2 in [Fig. 3]). When the robot moved, the change in orientation and travelled distance from the last estimated pose were measured by these two sensors to predict the new pose of the robot.

In the measurement update step, we combined a laser beam model with an environment model of the orchard to calculate probabilities for individual laser beams to hit a tree trunk or to pass further. The laser beam model and the environment model were inspired by the work of Hiremath et al. (2014), who used this approach for robot localization in maize fields. We modified their laser beam model and their environment model to allow robot localization in an orchard environment. Specifically, we excluded the regions of overhanging leaves on both sides of the maize stem that were presented by Hiremath et al. (2014). As a result, our environment model was based on three regions on each side of the robot [Fig. 4]. R_1 was the region in which laser beams passed before they reached the area that contained tree trunks. R_2 was the region that contained the tree trunks. R_3 was the region beyond R_2.

The distance a laser beam had to travel to reach a region, depended on the four state variables obtained from the particle ($\hat{d}_p$, $\hat{\alpha}_p$, $\hat{r}_b$, $\hat{r}_w$), and the angle at which a laser beam was emitted by the 2D LIDAR scanner (Φ) [Fig. 4]. The distance covered by a laser beam was calculated. b_1 represented the travelled distance of a laser beam to the end of region R_1 (Eq. (1)). b_2 represented the laser beam distance to the end of region R_2 (Eq. (2)). b_{max} was the maximum distance a laser beam could travel. However, as laser beams could pass on and hit a tree trunk in an adjacent row, the maximum distance was limited to the inter-row distance (Eq. (3)).

$$b_1(\phi) = \frac{\left(\frac{\hat{rb}_p - \hat{rw}_p}{2}\right) - \hat{d}_p}{\sin(\phi + \hat{\alpha}_p)} \quad [\text{m}] \tag{1}$$

$$b_2(\phi) = \frac{\left(\frac{\hat{rb}_p + \hat{rw}_p}{2}\right) - \hat{d}_p}{\sin(\phi + \hat{\alpha}_p)} \quad [\text{m}] \tag{2}$$

$$b_{max}(\phi) = \frac{\hat{rb}_p}{sin(\phi + \hat{\alpha}_p)} \quad [\text{m}] \tag{3}$$

Using the distance calculations (Eqs. (1)–(3)), we evaluated the correspondence of every particle with the distances measured by the LIDAR scanner. Per particle, 135 laser beams were analyzed using a 2° laser beam interval for the sake of computation speed. For each of the 135 laser beams, a probability density function was obtained that represented the hit and miss probabilities of the laser beam as a function of its travelled distance x. We used four equations from Hiremath et al. (2014) to obtain the PDF's (Eqs. (4)–(7)). In region R_1 [Fig. 4], assuming that there are no obstacles, the only cause for a beam returning is measurement noise (Eq. (4)). We assumed a 1% probability ($\lambda = 0.01$) of measurement noise per travelled meter of the laser beam.

$$P_1(x) = \lambda \cdot e^{-\lambda x} \quad \text{with} \quad x \in [0, b_1] \quad [-] \tag{4}$$

In region R_2, the probability for a laser beam return was influenced by three aspects (Eq. (5)). First, there is the probability that a laser beam passes through region R_1 without hitting an object before

Table 1
State variables of the particle filter.

State variable	Symbol	Value at initialization sampled from an uniform distribution (min, max)	Unit
Lateral deviation	$\hat{d}_p$	$U(-0.1, 0.1)$	m
Angular deviation	$\hat{\alpha}_p$	$U(-10, 10)$	°
Inter-row distance between two tree rows	$\widehat{rb}_p$	$U(-0.5, 0.5)$ + inter-row distance	m
Tree row width	$\widehat{rw}_p$	$U(0.05, 0.5)$	m

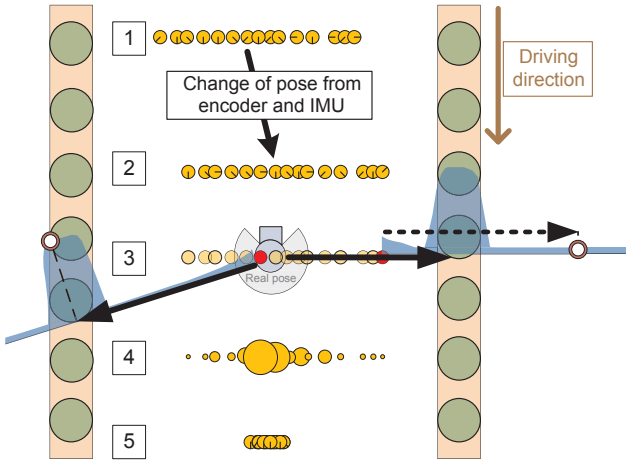

Fig. 3. A schematic explanation of the Particle filter. The Particle filter started with the initialization of random particles (1), followed by the predict step (2). The measurement update step (3) obtained particle weights using probability density functions (PDF). A higher weight was assigned to the particles that corresponded to the actual measured laser beam distance by the 2D LIDAR scanner (solid arrow on the left). The dashed arrow depicts the actual measured LIDAR distance inserted in the PDF of a less probable particle, resulting in a lower weight (4). In the resampling step all weights were used to obtain a new set of particles from which the lateral and angular deviation were estimated (5). Afterwards, step 2–5 were sequentially executed in each processing cycle.

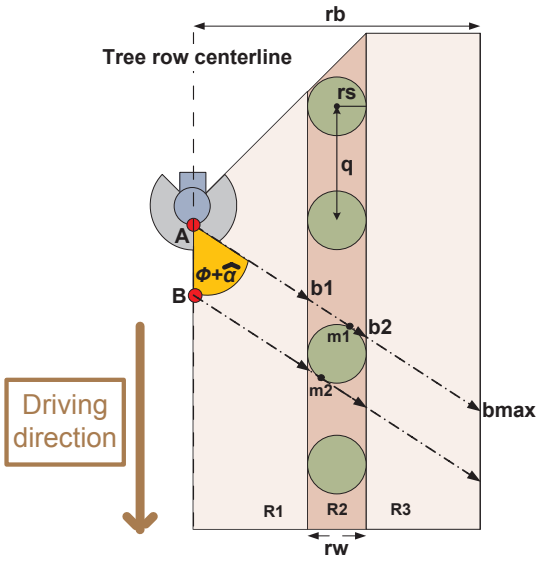

Fig. 4. Orchard environment model consisting of regions R_1, R_2 and R_3 on each side of the robot. b_1, b_2 and b_{max} depict the distance a laser beam had to travel to reach these regions. Between the points A and B, a laser beam from scan angle ϕ was expected to hit a tree trunk in region R_2. m_1 is the first point of a hit return and m_2 the last. rs represents the tree trunk radius and q the intra-row distance.

reaching region R_2. This was modelled by $\psi_1 = (1 - P_1(b_1))$. Second, there is a probability that a laser beam hits a tree trunk inside region R_2. This probability, expressed by p_{hit}, depended on the intra-row distance (q) and the tree trunk radius (rs) (Eq. (6)). Third, there is a probability of measurement noise that was estimated by a 1% probability per travelled meter of the laser beam ($\lambda = 0.01$).

$$P_2(x) = \psi_1 \cdot \left(\frac{p_{hit}}{b2 - b1} + (1 - p_{hit}) \cdot e^{-\lambda(x - b_1)} \right) \text{ with } \in [b_1, b_2] \ [-] \quad (5)$$

$$p_{hit} = \frac{2 \cdot rs}{q \cdot \sin(\phi + \hat{\alpha}_p)} \ [-] \quad (6)$$

In region R_3, the probability for a laser beam return was influenced by two aspects. First, there is the probability that a laser beam passes through regions R_1 and R_2 without hitting an object before reaching region R_3. This was expressed by ψ_1 and $\psi_2 = (1 - P_2(b_2))$. Second, there is a probability of measurement noise that was estimated by a 1% probability per travelled meter of the laser beam ($\lambda = 0.01$).

$$P_3(x) = \psi_1 \cdot \psi_2 \cdot \lambda \cdot e^{-\lambda(x - b2)} \text{ with } x \in [b_2, b_{max}] \ [-] \quad (7)$$

For each of the 135 laser beams, we calculated the probabilities P_1, P_2 and P_3, which were then combined into one probability density function (PDF). Then, the observed distance by the LIDAR scanner was inserted in the PDF to obtain a likelihood for this laser beam to be returned (step 3 in [Fig. 3]). By multiplying the likelihoods of the 135 analyzed beams, a total weight per particle was obtained (step 4 in [Fig. 3]). Particles with a high weight had a high probability to represent the actual pose of the robot. The filtering of highly probable particles was done by low variance resampling proportionally to the particle's weight, similar to the method described by Thrun et al. (2005). The resampling was executed in each measurement update step to obtain a new set of 125 particles. From the new set a posterior belief on the lateral and angular deviation was calculated as input for the robot navigation (step 5 in [Fig. 3]). After the resampling, random noise was added to the newly obtained set of particles to prevent rapid particle convergence and to increase particle diversity for robust robot localization.

2.3. Kalman filter algorithm with line-detection

The KF algorithm estimated the same two pose variables as the PF; the robot's lateral deviation (d) and the angular deviation (α) from the centerline between two tree rows. The KF modelled both state beliefs by unimodal Gaussian distributions. Similar to the PF, two events influenced the state beliefs sequentially and repeatedly: the predict step and the measurement update step. The predict step was similar to the PF to allow equal comparison. The measurement update step differed and was built with a line-detection method, using k-means clustering and least squares line fitting. First, the k-means (MacQueen, 1967) clustered the observed laser points into two groups to represent the two tree rows; one tree row on the left and one on the right side of the robot. These two cluster groups were produced by iteratively minimizing the total intra-cluster variance (J) using an Euclidean distance function (Eq. (8)).

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} ||x_i^{(j)} - c_j||^2 \quad [\text{m}] \tag{8}$$

where k is the number of clusters, n is the number of laser scan points, $x_i^{(j)}$ is the position of a laser scan point i in cluster j and c_j is the position of the centroid of cluster j.

After the clustering, an ordinary least squares procedure was used to fit lines through the laser scan points. Eq. (9) was used to determine the level of fit of a proposed line. The line with the highest R^2 was selected and fitted through the cluster of scan points as an estimation of the tree row. This was done on each side of the robot, resulting in two lines. A centerline was calculated by averaging the start and the end points of both lines. From this approximated centerline, the angular and lateral deviation were determined using the principle of [Fig. 2]. To model the state beliefs by Gaussians, a standard deviation was chosen as 1-R^2, which was obtained from Eq. (9).

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (x_i - \widehat{x_i})^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \quad [-] \tag{9}$$

where n is the number of laser scan points, $x_i$is the horizontal position of a laser scan point in the dataset, $\widehat{x_i}$ is the horizontal position obtained from the fitted linear equation and $\bar{x}$ is the average horizontal position in the cluster group.

The KF estimated the posterior belief by combining the Gaussian from the prior belief with the Gaussian from the measurement update step using the Bayes rule product (Eqs. (10) and (11)). The lateral and angular deviation were obtained from the mean of the resulting Gaussian and used as basis for robot navigation.

$$\mu' = \frac{\tau^2 \mu + \sigma^2 \nu}{\tau^2 + \sigma^2} \quad [\text{m}, °] \tag{10}$$

$$\sigma^{2'} = \frac{1}{\frac{1}{\tau^2} + \frac{1}{\sigma^2}} \quad [\text{m}, °] \tag{11}$$

where (μ, σ^2) are the mean and variance of the Gaussian of the prior belief after the predict step, (ν, τ^2) are the mean and variance of the Gaussian after the measurement update step and $(\mu', \sigma^{2'})$ are the mean and variance of the Gaussian of the posterior state belief.

2.4. Robot control

Both localization algorithms were processed at fixed cycles of 0.5 s to allow a synchronized robot control with the RTK-GNSS ground truth measurements. The last available laser scan data was used by both algorithms to localize the robot. From the obtained lateral and angular deviation, we calculated the desired steering angle of the robot (θ) using Eq. (12) that was derived from Hague and Tillett (1996). In Eq. (12), the robot's velocity was taken into account to critically damp the steering control at all velocities. We introduced an additional scaling factor w_α that was set to 0.5 to force the robot to respond stronger to an observed lateral deviation than an angular deviation. In addition, the scaling weight w_d was set to 0.25 to generate a steering angle that was considered safe for this robot when the lateral or angular deviation approached their maximum values. The desired steering angle was inserted into a proportional-integral–derivative (PID) control function to calculate the commanded angle that was sent to the robot. The PID-control gains were obtained by an iterative trial-and-error search with different combinations of control gains. We found that the combination of $K_p = 0.75$, $K_i = 0.05$ and $K_d = 0.40$ yielded the desired steering response for this robot. The steering controls and gains were the same during the field experiments.

$$\theta(t) = -w_d \cdot \hat{d} - w_\alpha \cdot \sqrt{4 \cdot w_d \cdot \text{v}} \cdot \hat{\alpha} \quad [\text{rad}] \tag{12}$$

where w_d is the scaling weight for the lateral deviation, w_α the scaling weight for the angular deviation and v the velocity of the robot.

2.5. Field experiments

Two experiments were executed in a commercial apple orchard in February 2017 to evaluate the navigation performance of the robot when using either the PF or the KF algorithm. The orchard, which was located in Randwijk (The Netherlands), had a flat surface and consisted of straight tree rows with an intra-row distance of 1.0 m and an inter-row distance of 3.0 m. The first experiment assessed the navigation accuracy when driving the robot autonomously in the orchard. The accuracy was determined by the lateral and angular deviation of the robot when navigating at two velocities, 0.25 m/s and 0.50 m/s. We used two orchard paths of 100 m that were driven by the robot in the same direction to allow equal comparison. The first 10 m travelled by the robot was not taken into account to make sure that the starting pose of the robot did not influence the outcome of the accuracy experiment. At the end of the orchard path, the robot was stopped by the human operator.

The second experiment assessed the navigation robustness of both algorithms. The robustness was determined by the lateral and angular deviation of the robot when it drove autonomously between six tree row patterns with missing trees. Trees in an orchard can be absent for multiple reasons, for instance when there is a maintenance path perpendicular to the tree rows or when diseased trees are removed. As trees could not be physically removed in the commercial orchard, the following procedure was used to virtually remove trees from the laser scan data. Highly reflective tape was attached to the tree trunks, which caused the intensity of the laser beams returned from these trees to be much higher than the beams returned from trees that were not covered by the tape. Typical intensities of laser beam returns from the high reflective tape were in the range 800–1000, while beam intensities from trees and branches were in range 200–600. The software removed the laser scan points with intensity higher than 700 from the measurement data, making these tree trunks invisible for the robot. Six different patterns of missing trees were tested [Fig. 5], when driving the robot 15 m with a velocity of 0.25 m/s. Each pattern was repeated two times.

In both experiments, we used the RTK-GNSS receiver to track the robot's trajectory that was stored on the robot's computer for offline processing. The driven trajectories were processed in such a way that only the paths were assessed where a RTK-fix signal was guaranteed. As such, all ground truth locations measured by the RTK-GNSS receiver were believed to have an accuracy up to 0.02 m from the real-world location. The lateral deviation of the robot (d) was calculated by taking the shortest perpendicular distance from the robot's position to the centerline (Eq. (13)). The angular deviation (α) was obtained by subtracting the robot's heading from the fixed heading of the ground truth centerline. The ground truth centerlines were constructed by connecting the start and end points of the paths, which were obtained by manually placing the RTK-GNSS receiver midway between the two tree rows of the driven paths.

$$d(t) = \frac{(y_2 - y_1)x_t - (x_2 - x_1)y_t + x_2 y_1 - y_2 x_1}{\sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}} \quad [\text{m}] \tag{13}$$

where (x_t, y_t) are the coordinates of the robot's position at time t, (x_1, y_1) the coordinates of the start point of the ground truth centerline and (x_2, y_2) the coordinates of the end point of the ground truth centerline.

We used the root mean square error (RMSE) as a metric to evaluate the navigation performance in both experiments. The RMSE was calculated as a measure for the average magnitude of the angular and lateral deviation and assigned high weights to large deviations without considering their direction (Eq. (14)).

$$RMSE = \sqrt{\frac{\sum_{t=1}^{n} (p_t - a_t)^2}{n}} \quad [\text{m}, °] \tag{14}$$

where p_t is the predicted value at time t, a_t the actual value at time t and n the number of observations.

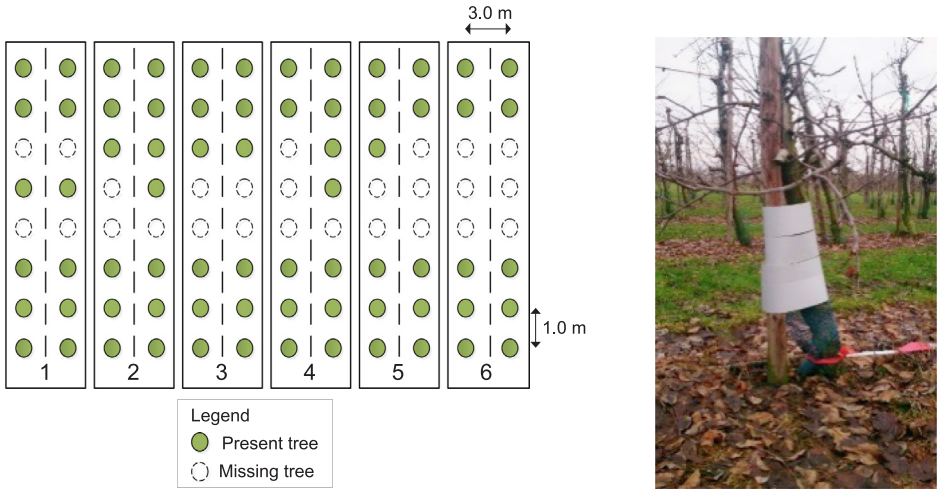

Fig. 5. The six tree row patterns that were used to test the algorithms' robustness to cope with missing trees in the row (left), and an image of a tree that was covered by high reflective tape to virtually remove the tree trunk from the laser scan data (right).

3. Results and discussion

In both experiments, we observed a non-bumpy robot navigation without collisions or malfunctions. In total, the robot navigated 1200 m autonomously in eight different trials. The first experiment, with its two trials, assessed the navigation accuracy when the robot navigated with both algorithms at 0.25 m/s and 0.50 m/s. A RMSE of the lateral deviation of 0.055 m was observed when the robot navigated with the PF algorithm at 0.25 m/s [Table 2]. A RMSE of 0.087 m was observed with the KF algorithm at the same velocity. At 0.50 m/s, the RMSE was 0.062 m with the PF algorithm and 0.091 m with the KF algorithm [Table 2]. The first two boxplots on the left of [Fig. 6] indicate that 50% of the PF's lateral deviation was within 0.05 m from the centerline at both velocities. When the robot navigated with the KF, less than 25% of the lateral deviation was within 0.05 m from the centerline.

When the robot navigated with the KF at 0.25 and 0.50 m/s, more than 75% of the lateral deviation was biased left to the optimal line. [Fig. 6]. When we analyzed this suboptimal localization performance, we observed that the least squares line fitting was susceptible to remote clusters of laser scan points that were returned from tree branches. Because the least squares line fitting only generated one best line fit for a cluster group of laser scan points, these remote scan points caused the lines to shift from the real tree trunks. This effect was particularly evident with clusters close to the robot, causing the creation of non-parallel lines [Fig. 7]. As a consequence, the centerline was sometimes wrongly estimated causing an unjustified robot steering and a lateral deviation that was biased to one side of the centerline, causing the RMSE of the lateral deviation to increase. Because the robot travelled mostly on one side of the centerline with the KF, we also observed a 0.62° and 0.28° lower RMSE of the angular deviation at 0.25 and 0.50 m/s velocity [Table 2].

In the second experiment we observed that the PF had a lower RMSE of the lateral deviation in five out of six tree row patterns with missing trees [Table 2]. The boxplots of [Fig. 6] highlight that 50% the PF's lateral deviation was within 0.05 m from the centerline for all tree row patterns. The KF achieved the same in only three patterns. In addition, the robot was biased to the left of the centerline in patterns 2, 3 and 6 when using the KF for robot localization. This bias resulted in a higher RMSE of the lateral deviation when compared to the PF [Table 2]. When investigating the KF's suboptimal localization, we found some erroneous line shifts caused by laser scan data of tree branches that were distant from the tree trunks. These line shifts were particularly severe at patterns 3 and 6 that contained adjacent missing trees on both sides of the robot [Fig. 7]. The RMSE's of the angular deviation of the PF and KF were in the same range [Table 2]. Both the PF's and the KF's angular deviations were equally distributed to both sides of the optimal line [Fig. 8].

We evaluated the two algorithms with the same robot platform using identical steer control parameters and fixed data processing cycles of 0.5 s. The last allowed the synchronization with the RTK-GNSS ground truth measurements, however it also caused a latency when controlling the robot. This control latency was especially high with the faster line-based KF. We acknowledge that a real-time control, in which the robot is directly controlled after finishing the algorithm's calculation, could have resulted different navigation performances or conclusions in this research. Yet, we expect these effects to be marginal, because the robot navigated at low velocities (≤ 0.5 m/s) in both experiments.

By comparing the two algorithms in similar conditions, we were also able to compare two different world models of the orchard

Table 2
The root mean square error (RMSE) of the lateral and angular deviation with respect to the centerline when navigating the robot with the Particle filter and Kalman filter.

			RMSE of the lateral deviation [m]		RMSE of the angular deviation [°]	
Trial	Experiment	Navigation test	Particle filter	Kalman filter	Particle filter	Kalman filter
1	Accuracy	v = 0.25 m/s	0.055	0.087	3.235	2.615
2	Accuracy	v = 0.50 m/s	0.062	0.091	2.155	1.871
3	Robustness	Tree row pattern 1	0.044	0.039	2.671	2.470
4	Robustness	Tree row pattern 2	0.033	0.086	2.592	2.403
5	Robustness	Tree row pattern 3	0.041	0.081	2.473	2.428
6	Robustness	Tree row pattern 4	0.033	0.042	2.552	2.120
7	Robustness	Tree row pattern 5	0.036	0.058	2.776	3.045
8	Robustness	Tree row pattern 6	0.044	0.070	2.516	2.848

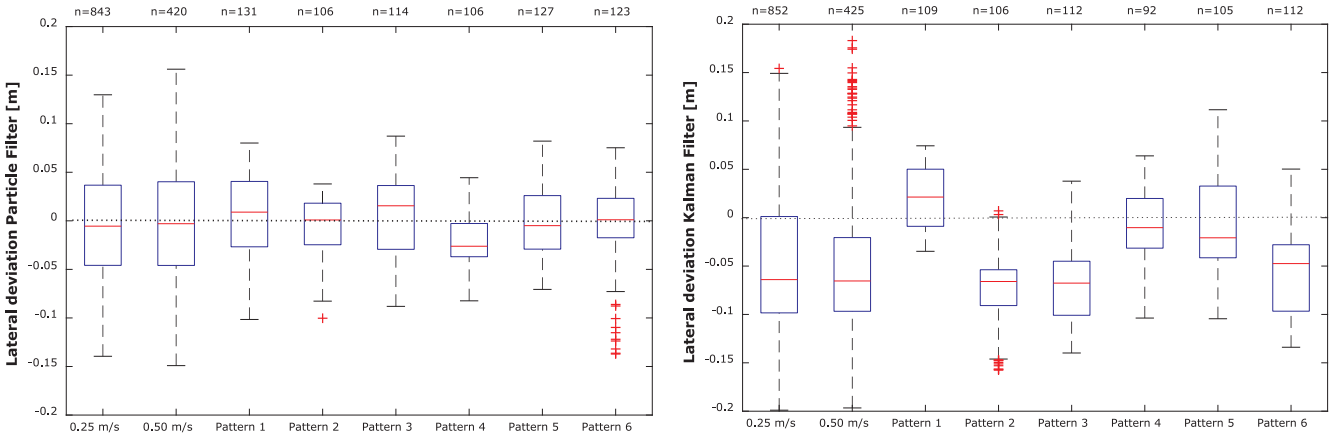

Fig. 6. Boxplots of the signed lateral deviation from the centerline (dashed line) at both navigation experiments when the robot navigated with the Particle filter (left) and the Kalman filter (right). The value *n* corresponds to the number of uniquely stored RTK-GNSS locations.

environment. Both world models were based on the assumption of straight tree rows, however the PF's world model used more a-priori information about the orchard environment. This higher degree of a-priori information in combination with the assessment of individual laser beams, allowed the PF algorithm to navigate the robot with a lower RMSE of the lateral deviation in seven out of eight trials in this orchard environment. To allow robot navigation in orchards with straight tree rows but different spacing's or configurations, it is required to enter this a-priori information in advance in the PF algorithm. The simplicity of the KF's world model, which is based on one straight tree row on each side of the robot, does not require user input when the robot is used in an orchard with different intra- or inter-row distances.

The navigation results were obtained in winter time, which means a stagnated growth of grass and weeds and the absence of leaves, blossoms and fruits on the trees. In other seasons, we expect more laser beam returns from tall grass, weeds, leaves and tree branches that bend by the weight of fruits and leaves. These environmental changes might require an alternation of both world models to guarantee accurate robot localization and navigation in orchards. The PF's world model could be extended with two "tree branch" regions, one in front of the tree trunk region and one beyond, comparable to the method presented in Hiremath et al. (2014). The applicability of this method was proven by Hiremath et al. (2014), who showed a RMSE of the lateral deviation of 0.04 m when navigating a robot between maize plants. The KF's world model could be extended with a more sophisticated line-detection algorithm that assigns a higher probability when the two lines are parallel to each other. We also believe that the RANSAC algorithm might be a better alternative for line detection, as this algorithm attempts to exclude outliers from its linear fit. Marden and Whitty (2014) observed a RMSE of the lateral deviation of 0.04 m when combining RANSAC-based SLAM with an Extended Kalman filter (EKF). Another promising line-detection algorithm that is based on refinements of the PEARL algorithm (RUBY), outperformed the traditional RANSAC algorithm when tested on simulated LIDAR data with outliers (Malavazi et al., 2018). In addition, the influence of outliers in the location estimate can be reduced by increasing the detection range of the LIDAR scanner or by applying data fusion of different sensors. Shalal et al. (2015) investigated an EKF fusion of LIDAR data and camera images that yielded a RMSE of the lateral deviation of 0.089 m when navigating a robot midway between two tree rows. A RMSE of 0.094 m was observed when the robot navigated close to one tree row for close-range crop monitoring (Shalal et al., 2015). Although these RMSE's are slightly higher than the ones obtained from our research, we believe that the fusion of LIDAR data with camera images is promising to exclude tree branches or non-tree objects from the location estimate.

4. Conclusions

Based on our findings, we concluded that the Particle filter (PF) with the laser beam model outperformed the line-based Kalman filter (KF) on navigation accuracy. Regarding navigation robustness, it was concluded that the PF outperformed the KF in five out of six trials. When the orchard robot navigated with the KF, the lateral deviations were biased to the left of the centerline in five out of eight trials. The angular deviations of the PF and the KF were in the same range and equally distributed to both sides of the optimal line. From both experiments we concluded that a PF with laser beam model had better navigation accuracy and was more robust to deal with missing trees than a line-based KF.

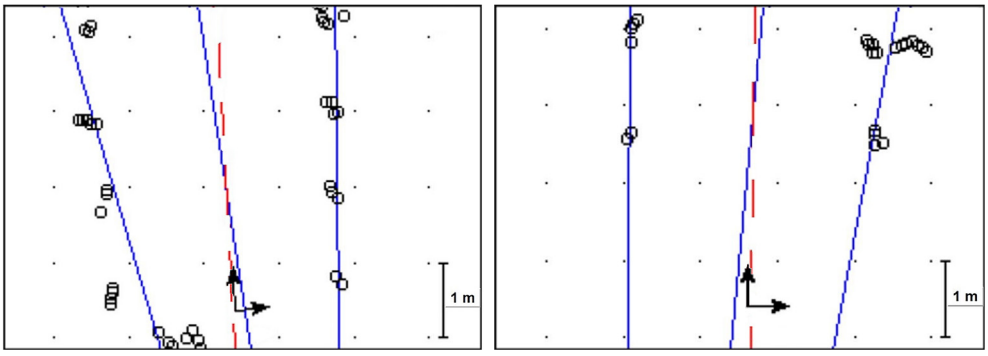

Fig. 7. The left figure depicts laser scan points (circles) from the first experiment and the right figure scan points from the second experiment. Both figures indicate an incorrect line shift by the least squares line fitting in the Kalman filter (solid lines) caused by clusters of laser scan returns from tree branches. The arrows represent the ground truth pose of the robot and the dashed lines the pose estimates of the Particle filter.

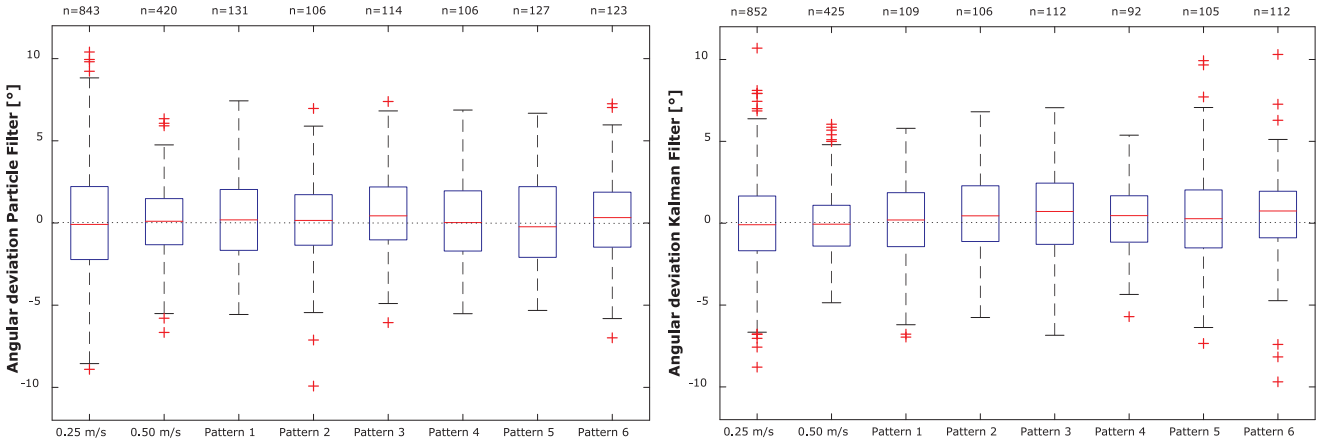

Fig. 8. Boxplots of the signed angular deviation from the centerline (dashed line) at both navigation experiments when the robot navigated with the Particle filter (left) and the Kalman filter (right). The value n corresponds to the number of uniquely stored RTK-GNSS locations.

Nomenclature

Symbol	Value	Units	Description
a_t		m,°	The actual value at time t
b_1		m	The travelled distance of a laser beam to the end of region R_1 of the orchard environment model
b_2		m	The travelled distance of a laser beam to the end of region R_2 of the orchard environment model
b_{max}		m	Maximum distance a laser beam could travel in the orchard environment modelmodel
c_j		m	The position of the centroid of cluster j
d		m	Lateral deviation of the robot to the centerline midway between two tree rows
J		m	Total intra-cluster variance between two cluster groups of laser points
k		–	The number of clusters used in the k-means
K_d	0.40	–	Derivative gain of the PID control
K_i	0.05	–	Integral gain of the PID control
K_p	0.75	–	Proportional gain of the PID control
m_1		m	Location of the first edge point of a tree trunk that can return a laser beam
m_2		m	Location of the last edge point of a tree trunk that can return a laser beam
n		–	Number of observations
P_1		–	Probability of a laser beam return from region R_1 of the orchard environment model
P_2		–	Probability of a laser beam return from region R_2 of the orchard environment model
P_3		–	Probability of a laser beam return from region R_3 of the orchard environment model
p_{hit}		–	Probability of a laser beam return from a tree trunk inside region R_2 of the orchard environment model
p_t		m,°	The predicted value at time t
q		m	Intra-row distance between trees in the row
R^2		–	The level of fit of a proposed line generated by the ordinary least squares procedure
rb		m	Inter-row distance between two tree rows
rs		m	Tree row radius
rw		m	Tree row width
U		–	Uniform distribution
v		m/s	Velocity
x		m	The travelled distance of a laser beam
$\bar{x}$		m	The average horizontal position of a set of laser points
$\widehat{x_i}$		m	The horizontal position of a laser point i obtained from the fitted linear equation using ordinary least squares
x_1		m	Horizontal coordinate of the start point of the ground truth centerline between two tree rows
x_2		m	Horizontal coordinate of the end point of the ground truth centerline between two tree rows
x_i		m	The horizontal position of a laser scan point i
x_i^j		m	The position of a laser scan point i in cluster j
x_t		m	Horizontal coordinate of the robot's position at time t
y_1		m	Vertical coordinate of the start point of the ground truth centerline between two tree rows
y_2		m	Vertical coordinate of the end point of the ground truth centerline between two tree rows
y_t		m	Vertical coordinate of the robot's position at time t
w_d	0.25	–	Scaling weight for the lateral deviation
w_α	0.50	–	Scaling weight for the angular deviation
α		°	Angular deviation of the robot from the centerline between two tree rows
θ		rad	Desired steering angle of the robot
λ	0.01	–	Probability of a laser beam return from measurement noise
μ		m,°	Mean of the Gaussian of the prior belief after the predict step
μ'		m,°	Mean of the Gaussian of the posterior state belief
ν		m,°	Mean of the Gaussian after the measurement update step
σ^2		m,°	Variance of the Gaussian of the prior belief after the predict step
$\sigma^{2\prime}$		m,°	Variance of the Gaussian of the posterior state belief
τ^2		m,°	Variance of the Gaussian after the measurement update step
Φ		°	The angle at which a laser beam was emitted by the 2D LIDAR scanner
ψ_1		–	Probability that a laser beam passes through region R_1 without a hit return before reaching region R_2
ψ_2		–	Probability that a laser beam passes through regions R_1 and R_2 without a hit return before reaching region R_2

Acknowledgements

This work was supported by the Cooperative Research Program for Agricultural Science & Technology Development [grant number PJ012289] of the Rural Development Administration (Republic of Korea).

Appendix A. Supplementary material

Supplementary data to this article can be found online at https://doi.org/10.1016/j.compag.2018.12.046.

References

Andersen, J.C., Ravn, O., Andersen, N.A., 2010. Autonomous rule-based robot navigation in orchards. IFAC Proc. 43 (16), 43–48.

ASI-Robots, 2018. Orchard and vineyard automation. Available: https://www.asirobots.com/farming/orchard-vineyard/ (Accessed 07-12-2018).

Barawid Jr, O.C., Mizushima, A., Ishii, K., Noguchi, N., 2007. Development of an autonomous navigation system using a two-dimensional laser scanner in an orchard application. Biosyst. Eng. 96 (2), 139–149.

Bergerman, M., Maeta, S.M., Zhang, J., Freitas, G.M., Hamner, B., Singh, S., Kantor, G., 2015. Robot farmers: autonomous orchard vehicles help tree fruit production. IEEE Robot. Autom. Mag. 22 (1), 54–63.

Bergerman, M., Singh, S., Hamner, B., 2012. Results with autonomous vehicles operating in specialty crops. In: 2012 IEEE International Conference on Robotics and Automation, pp. 1829–1835.
Blok, P.M., Suh, H.K., Boheemen, K.v., Kim, H.-J., Kim, G.H., 2018. Autonomous in-row navigation of an orchard robot with a 2D LIDAR scanner and particle filter with a laser-beam model. J. Inst. Control, Robot. Syst. 24 (8), 726–735 (in Korean with English abstract).
CH, R., 2013 UM6 Ultra-Miniature Orientation Sensor Datasheet. Available: http://www.chrobotics.com/docs/UM6_datasheet.pdf (Accessed 05-02-2018).
Chen, X., Wang, S.a., Zhang, B., Luo, L., 2018. Multi-feature fusion tree trunk detection and orchard mobile robot localization using camera/ultrasonic sensors. Comput. Electron. Agric. 147, 91–108.
Christiansen, M.P., Jensen, K., Ellekilde, L.-P., Jørgensen, R.N., 2011. Localization in orchards using Extended Kalman Filter for sensor-fusion-A FroboMind component.
Clearpath Robotics, 2016. Husky Unmanned Ground Vehicle - User Manual. Available: https://www.clearpathrobotics.com/husky-user-manual/ (Accessed 22-6-2016).
Freitas, G., Zhang, J., Hamner, B., Bergerman, M., Kantor, G., 2012. A low-cost, practical localization system for agricultural vehicles. In: International Conference on Intelligent Robotics and Applications. Springer, Montreal, Canada, pp. 365–375.
Greenbot, 2018. Greenbot. Available: https://www.greenbot.nl/ (Accessed 07-12-2018).
Hague, T., Tillett, N.D., 1996. Navigation and control of an autonomous horticultural robot. Mechatronics 6 (2), 165–180.
Hansen, S., Bayramoglu, E., Andersen, J.C., Ravn, O., Andersen, N., Poulsen, N.K., 2011. Orchard navigation using derivative free Kalman filtering. In: Proceedings of the 2011 American Control Conference. IEEE, San Francisco, United States of America, pp. 4679–4684.
Hansen, S., Blanke, M., Andersen, J.C., 2009. Autonomous tractor navigation in orchard - diagnosis and supervision for enhanced availability. IFAC Proc. Volumes 42 (8), 360–365.
Hiremath, S.A., 2013. Probabilistic methods for robotics in agriculture. Unpublished PhD Ph.D. dissertation, Wageningen University, Wageningen, The Netherlands.
Hiremath, S.A., van der Heijden, G.W.A.M., van Evert, F.K., Stein, A., ter Braak, C.J.F., 2014. Laser range finder model for autonomous navigation of a robot in a maize field using a particle filter. Comput. Electron. Agric. 100 (1), 41–50.
Jæger-Hansen, C.L., Griepentrog, H.W., Andersen, J.C., 2012. Navigation and tree mapping in orchards. In: International Conference of Agricultural Engineering (CIGR- Ag Eng 2012),Valencia, Spain, pp. 1–6.
Kurashiki, K., Fukao, T., Ishiyama, K., Kamiya, T., Murakami, N., 2010. Orchard traveling UGV using particle filter based localization and inverse optimal control. In: System Integration. IEEE, pp. 31–36.
Lepej, P., Rakun, J., 2016. Simultaneous localisation and mapping in a complex field environment. Biosyst. Eng. 150, 160–169.
Li, M., Imou, K., Wakabayashi, K., Yokoyama, S., 2009. Review of research on agricultural vehicle autonomous guidance. Int. J. Agric. Biol. Eng. 2 (3), 1–16.
Libby, J., Kantor, G., 2010. Accurate GPS-free positioning of utility vehicles for specialty agriculture. In: 2010 ASABE Annual International Meeting, Pittsburgh. American Society of Agricultural and Biological Engineers, United States of America, pp. 1–14.
MacQueen, J., 1967. Some methods for classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, Oakland, United States of America, pp. 281-297.
Malavazi, F.B.P., Guyonneau, R., Fasquel, J.-B., Lagrange, S., Mercier, F., 2018. LiDAR-only based navigation algorithm for an autonomous agricultural robot. Comput. Electron. Agric. 154, 71–79.
Marden, S., Whitty, M., 2014. GPS-free localisation and navigation of an unmanned ground vehicle for yield forecasting in a vineyard. In: Proceedings of the 13th International Conference on Intelligent Autonomous Systems, pp. 1–10.
NAIO-Technologies, 2018. Ted Robot. Available: https://www.naio-technologies.com/en/category/ted-robot/ (Accessed 07-12-2018).
Precision-Makers, 2018. X-Pert. Available: https://www.precisionmakers.com/en/x-pert/ (Accessed 13-12-2018).
Shalal, N., Low, T., McCarthy, C., Hancock, N., 2013. A review of autonomous navigation systems in agricultural environments. In: 2013 Society for Engineering in Agriculture Conference: Innovative Agricultural Technologies for a Sustainable Future: Engineers Australia, pp. 10.
Shalal, N., Low, T., McCarthy, C., Hancock, N., 2015. Orchard mapping and mobile robot localisation using on-board camera and laser scanner data fusion–Part B: mapping and localisation. Comput. Electron. Agric. 119 (1), 267–278.
Sick Sensor Intelligence, 2016. Sick LMS111-10100 Datasheet. Available: https://www.sick.com/media/pdf/2/42/842/dataSheet_LMS111-10100_1041114_nl.pdf (Accessed 16-8-2016).
Statista, 2018. Fruit Production - Statistics & Facts. Available: https://www.statista.com/topics/1621/fruit-production/ (Accessed 06-09-2018).
Thrun, S., Burgard, W., Fox, D., 2005. Probabilistic Robotics. MIT Press.
Weiss, U., Biber, P., 2011. Plant detection and mapping for agricultural robots using a 3D LIDAR sensor. Rob. Auton. Syst. 59 (5), 265–273.
Zhang, J., Maeta, S., Bergerman, M., Singh, S., 2014. Mapping orchards for autonomous navigation. In: 2014 ASABE and CSBE/SCGAB Annual International Meeting. American Society of Agricultural and Biological Engineers, Montreal, Canada, pp. 1–9.