Deployment of a Point and Line Feature Localization System for an Outdoor Agriculture Vehicle

Jacqueline Libby and George Kantor

***Abstract*— This paper presents a perception-based GPS-free approach for localizing a mobile robot in an orchard environment. An extended Kalman filter (EKF) algorithm is presented that uses a wheel odometry prediction step and laser rangefinder update steps. There are two update steps, one that uses measurements to reflective point features and one that uses measurements to linear features formed by tree rows. The features are associated to landmarks in previously surveyed maps. The practical issues of dealing with uncertainty both from the environment and the on-board sensors are discussed and accounted for. The resulting algorithm is demonstrated in over 20km of online operation in a variety of real orchard environments.**

I. INTRODUCTION

Localization of mobile vehicles is important in precision agriculture for a variety of reasons. Data collected from these vehicles can be geo-registered into maps, which field managers and scientists can use alike. This allows the same vehicle to return to specific locations and perform tasks such as spraying in a more targeted manner, thereby saving valuable resources. Localization is also critical for automated or semi-automated vehicles that can improve productivity for agricultural applications and fulfill the growing demand for labor.

The use of GPS for localization has many drawbacks for specialty crop settings. In orchards such as the ones used in this work (Fig. 1), the line-of-sight to satellites can get occluded by tree canopies. This occlusion problem does not occur in broad-acre crops, where GPS has been successfully used for many years [1]. Even without signal interference, GPS systems that provide sub-meter accuracy are prohibitively expensive for most specialty agriculture applications. Additionally, GPS does not provide information about the orientation of the vehicle, which is necessary for automated steering, as well as for determining the position of objects observed by vehicle-mounted sensors.

This research is part of the CASC project (Comprehensive Automation for Specialty Agriculture), funded by the USDA, to provide new technologies for specialty crops that are reliable and affordable [2]. Our role is to automate a robotic utility vehicle which can drive up and down orchard rows with a variety of sensors to intelligently make sense of itself and the environment. This paper describes efforts to date to localize the vehicle in real time without the use of GPS. Our approach uses sensors already on the vehicle for other purposes, thereby adding no cost or infrastructure to the platform. Wheel encoders, which are already being used for vehicle control, are used to provide a preliminary pose estimate based on dead reckoning. 2D lasers fixed to the vehicle, which are already being used for obstacle avoidance, are used to detect point and line features. These features are used in an extended Kalman filter (EKF) for pose estimate corrections.

Fig. 1. Fruit wall in a typical orchard environment. Tall tree canopies make GPS unreliable. Shrubs, low-hanging branches, and thin trunks make feature extraction difficult.

The orchards we test in have many environmental challenges (Fig. 1). The rows of trees are called fruit walls, resembling vines that grow along wires. The trees are closely spaced and the trunks have very small diameters, with branches that often hang low to the ground. This environment is carefully engineered to maximize light interception to the canopy. Our robot is limited in what it can sense with its fixed 2D lasers. Tree trunks are narrow and often occluded by leaves, ruling out the possibility of using them as point features. Line features can be fit to the straight rows of trees, but these lines are very noisy due to the organic shape of the canopy.

The work presented here uses a practical approach to deal with the constraints of our platform and the environment. We use a combination of naturally-existing line features that are formed by the tree rows and a small number of artificial point features. As the robot drives down a straight row, noisy lines fitted to the canopy are used to correct for crosstrack error. Reflective tape placed only at the ends of rows are used to correct for downtrack error when the robot nears the end of the row, as well as to correct for error as the robot makes tight turns from one row to the next. This allows the robot to traverse entire orchard blocks, and we demonstrate this with multiple endurance runs, where the robot is running online localization algorithms used as inputs for autonomous navigation.

II. RELATED WORK

Laser localization for autonomous ground vehicles in outdoor environments has made many recent advances. Kelly et. al. [3] use GPS for global pose estimates, and deal with GPS drop-outs by using a suite of sensors for local pose estimates

J. Libby is a PhD student in the Robotics Institute, School of Computer Science, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA, 15213, USA jlibby@cmu.edu

G. Kantor is Faculty in the Robotics Institute, School of Computer Science, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA, 15213, USA kantor@ri.cmu.edu

and obstacle avoidance. The winners of the DARPA Urban Grand Challenge [4] used an Applanix system for an initial pose estimate, and then lasers to detect the reflectance from road markers to correct for this estimate. Madhavan and Durrant-Whyte [5] use wheel encoders and lasers to localize in unstructured environments, using a combination of reflective landmarks and polylines to structures with clearly defined edges and corners. Guivant et. al. [6] use wheel encoders and lasers to perform EKF Simultaneous Localization and Mapping (SLAM) in park settings, using the trunks of large trees as point features.

Using line features for robot pose estimation and navigation has been developed primarily in indoor environments, where clean, man-made surfaces can be detected. Sack and Burgard [7] learn line models by extracting line segments from laser range scans, and then integrate them into a global map. Lu and Milios [8] perform laser odometry by matching lines that are fit to consecutive scans. Leonard et. al. [9] extract linear features from sonar data, which are then used for mapping and localization.

Autonomous navigation in outdoor agricultural environments has also been explored to some extent. Barawid et. al. [10] navigated an autonomous tractor down straight tree rows, using a 2D laser scanner to fit lines to the rows. Stentz et. al. [11] developed a safe and reliable tractor system predominantly used in orange groves, where GPS was used for localization, and a suite of cameras was used for sophisticated path tracking and obstacle detection.

In this paper, we build on this body of work by presenting a GPS-free solution to the localization problem with limited sensing in difficult outdoor environments. Our approach uses an EKF to combine point feature measurements with noisy line feature measurements created by organic structure (trees) instead of man-made walls. The algorithm has been implemented and validated through extensive experimentation in real-world settings.

III. EXPERIMENTAL PLATFORM

The platform for this work is an electric vehicle, equipped with brake and steering motors that allow for either autonomous or manual control (Fig. 2). Encoders on the rear differential and steering wheel measure distance traveled and steering angle, respectively, at 250 Hz. The black box frames the two lasers used in this work. Both have a 2D scan plane oriented horizontally. The lower laser is a SICK LMS 291, with a 35 Hz scanning frequency, an 80 m scanning range, a 1° angular resolution, and a 180° field of view. The upper is a SICK LMS 111, with a 50 Hz scanning frequency, a 20 m scanning range, a 0.5° angular resolution, and a 270° field of view. Each beam in a scan measures the range, r, and bearing, ϕ, to the nearest object in its path. Each beam also returns an intensity value, which measures the reflectivity of that object.

Fig. 2. Robotic utility vehicle used as the platform for the work presented. The black box frames the two horizontal forward-facing laser rangefinders used in this work.

A ruggedized laptop along with two embedded computers share the computational load of interfacing with sensors and processing data. The laptop runs a GUI for high level control. The hardware drivers and algorithms are nodes within a distributed communication framework built on top of ROS [12], an open-source Robot Operating System. In addition to seamless communication between modules, ROS provides a playback mechanism that allows us to test new algorithms on previously collected data, thereby saving hardware resources.

An onboard Applanix POS 220 LV high-accuracy positioning system is used to benchmark performance, as well as for some preprocessing steps. The Applanix fuses RTK corrected GPS, IMU, and distance-traveled encoder measurements to provide a 6 DOF pose estimate, accurate to within a few cm in position and 0.05° in orientation.

IV. TECHNICAL APPROACH

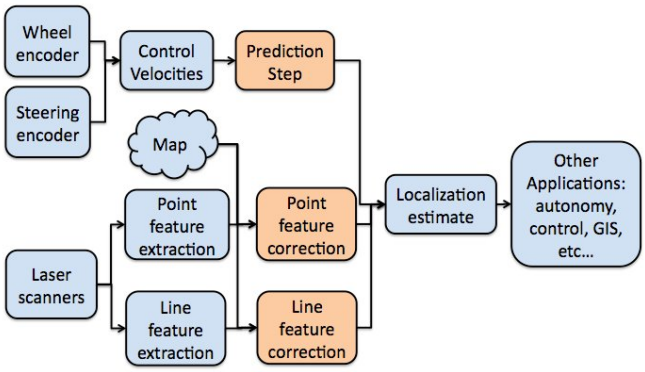

Fig. 3. Block diagram of system. The three steps of the EFK filter are colored orange.

A feature-based EKF is used to estimate the state of the vehicle. The state is the pose of the vehicle with respect to the world, with an estimate $\hat{\mathbf{q}} = \left[\hat{x}, \hat{y}, \hat{\theta}\right]^\mathsf{T}$ and associated covariance matrix $\mathbf{P}$. The filter uses an odometry-based prediction step and two types of laser-based update steps. The first uses measurements to point features and the second measurements to line features. Fig. 3 is a block diagram of the system architecture, with the three filter steps colored in orange.

A. Mapping Step

The EKF localization filter here requires an *a priori* map. The map consists of line features formed by rows of trees in the orchard and point features consisting of reflective tape placed at the ends of the rows (Fig. 4). To survey the map, we drive around the reflective tape with the vehicle. The lasers give high intensity returns when the tape is detected, and we use the vehicle pose from the Applanix to generate point clouds of these returns in the world frame. We cluster the point clouds and use the mean of each cluster as point

Fig. 4. Reflective tape placed around posts at the ends of rows. Arrows point to tape.

features in the map. The map, m, contains a list of 2D coordinates for each point feature, i, with respect to the world, w: $\mathbf{p}^w_{m,i} = \left(x^w_{m,i}, y^w_{m,i}\right)$.

Line features are generated by connecting the pair of endpoints along each row. Fig. 5 shows a top-down view of a test site, where the black dots represent these point features, and the green line segments connecting points down the length of a row are segments from the line features in the map. (The lengthwise and crosswise segments together create polygons for each row, which are used for data association, described in Section IV-D.) We define a line in polar coordinates, with respect to some frame a, as $\mathbf{l}^a = [d^a, \alpha^a]^\mathsf{T}$, where d^a is the perpendicular distance from the origin of frame a to the line, and α^a is the angle between the x-axis of frame a and the vector that runs along that perpendicular. We can then write a list of map coordinates for each line feature, j, with respect to the world frame, $\mathbf{l}^w_{m,j} = \left[d^w_{m,j}, \alpha^w_{m,j}\right]^\mathsf{T}$.

B. Prediction Step

We use a point and shoot model for the EKF prediction step:

$$\hat{\mathbf{q}}_t = f\left(\hat{\mathbf{q}}_{t-1}, \mathbf{u}_t\right) = \begin{bmatrix} \hat{x}_{t-1} + \triangle t v_t \cos\hat{\theta}_{t-1} \\ \hat{y}_{t-1} + \triangle t v_t \sin\hat{\theta}_{t-1} \\ \hat{\theta}_{t-1} + \triangle t \omega_t \end{bmatrix}, \quad (1)$$

where $\hat{q_t}$ is the current estimate of the pose, $\triangle t$ is the time difference between timesteps $t-1$ and t, and $u_t = [v_t, w_t]^\mathsf{T}$ are the forward and angular velocity inputs, computed using the encoder measurements. The noise on these velocity readings is modeled with the covariance matrix $\mathbf{U}$. (Refer to IV-E for noise modeling.) The state covariance prediction step is given by

$$\mathbf{P}_t = \mathbf{F}\mathbf{P}_{t-1}\mathbf{F}^\mathsf{T} + \mathbf{W}\mathbf{U}\mathbf{W}^\mathsf{T}, \quad (2)$$

where $\mathbf{F} = \frac{\partial f}{\partial \mathbf{q}}$ and $\mathbf{W} = \frac{\partial f}{\partial \mathbf{u}}$ are the Jacobians of the motion model given in (1).

C. Point Feature Correction Step

At a specific moment in time, the laser will detect a high intensity return to a point feature in the map, and give us a range and bearing point measurement, $\mathbf{z} = [r, \phi]^\mathsf{T}$. We write now more specifically the pose estimate as $\hat{\mathbf{q}} = \hat{\mathbf{q}}^w_v$, the pose of the vehicle with respect to the world. Our measurement model, $h\left(\hat{\mathbf{q}}^w_v, \mathbf{p}^w_{m,i}\right) = \left[\hat{r}, \hat{\phi}\right]^\mathsf{T}$, is the measurement we would expect to get at the pose estimate, $\hat{\mathbf{q}}^w_v$, to the landmark $\mathbf{p}^w_{m,i}$. h is a composite function,

$$h\left(\hat{\mathbf{q}}^w_v, \mathbf{p}^w_{m,i}\right) = h^*\left(g\left(\hat{\mathbf{q}}^w_v\right), \mathbf{p}^w_{m,i}\right), \quad (3)$$

where $g\left(\hat{\mathbf{q}}^w_v\right) = \hat{\mathbf{q}}^w_s$ transforms the world pose of the vehicle to the world pose of the sensor using the fixed pose of the sensor with respect to the vehicle, $\hat{\mathbf{q}}^v_s$. We then define h^* as the expected range and bearing between the sensor pose and the landmark:

$$h^*\left(\hat{\mathbf{q}}^w_s, \mathbf{p}^w_{m,i}\right) = \begin{bmatrix} \sqrt{\left(\hat{x}^w_s - x^w_{m,i}\right)^2 + \left(\hat{y}^w_s - y^w_{m,i}\right)^2} \\ \text{atan2}\left(\frac{\hat{y}^w_s - y^w_{m,i}}{\hat{x}^w_s - x^w_{m,i}} - \hat{\theta}^w_s\right) \end{bmatrix} \quad (4)$$

To choose the landmark, $\mathbf{p}^w_{m,i}$, we use standard chi-squared gating to associate to the one with the closest Mahalanobis distance, using the covariance of our current estimate, $\mathbf{P}$, and the covariance of the sensor noise, $\mathbf{R}_p$ (refer to Section IV-E). We calculate the Jacobian $\mathbf{H} = \frac{\partial h}{\partial \mathbf{q}}$ using the chain rule on our composite function. We can then plug $\hat{\mathbf{q}}^\mathbf{w}_\mathbf{v}$, $h\left(\hat{\mathbf{q}}^\mathbf{w}_\mathbf{v}, \mathbf{p}^w_{m,i}\right)$, $\mathbf{z}$, $\mathbf{P}$, $\mathbf{H}$ and $\mathbf{R}_p$ into the standard EKF update equations to correct for the state. (Refer to [13] for these equations.)

D. Line Feature Correction Step

A line measurement is obtained by fitting a line to a point cloud of laser returns. When the robot is operating in an orchard row, lines can be fit to the length of trees along the left and right sides of the vehicle. A Hough transform is used for an initial fit, and then a low pass filter gets rid of noisy outliers. (Refer to [14] for more details.) If both a left and right line measurement are received at the same time, we run a separate update step of the filter for each line before moving onto the next time step. $\mathbf{l}^s$ represents the polar coordinates of a line measurement in sensor frame. We model the uncertainty of the line measurement with covariance matrix $\mathbf{R}_l$.

We now define a function that maps a line in frame a, $\mathbf{l}^a$, to the equivalent line in frame b: $\mathbf{l}^b = \lambda^*\left(\mathbf{q}^a_b, \mathbf{l}^a\right)$, where $\mathbf{q}^a_b$ is the pose of frame b with respect to frame a. λ^* is a composite function,

$$\lambda^*\left(\mathbf{q}^a_b, \mathbf{l}^a\right) = \lambda^*_2\left(\lambda^*_1\left(\mathbf{q}^a_b, \mathbf{l}^a\right)\right). \quad (5)$$

λ^*_1 initially maps the line into the new frame,

$$\lambda^*_1\left(\mathbf{q}^a_b, \mathbf{l}^a\right) = \overline{\mathbf{l}}^b = \begin{bmatrix} \bar{d}^b \\ \bar{\alpha}^b \end{bmatrix} = \begin{bmatrix} d^a - x^a_b \cos\alpha^a - y^a_b \sin\alpha^a \\ \alpha^a - \theta^a_b \end{bmatrix}, \quad (6)$$

but since $\alpha \in (-\pi, \pi]$, we avoid redundancy by enforcing d to be positive:

$$\lambda^*_2\left(\overline{\mathbf{l}}^b\right) = \mathbf{l}^b = \begin{cases} \left[d^b, \alpha^b\right]^\mathsf{T} & \text{if } d > 0, \\ \left[-d^b, \alpha^b + \pi\right]^\mathsf{T} & \text{if } d < 0. \end{cases} \quad (7)$$

We can then use λ^* to define our line feature measurement model, $\lambda\left(\hat{\mathbf{q}}^w_v, \mathbf{l}^w_{m,j}\right)$, which is the line measurement we would expect to get at our current pose estimate, $\hat{\mathbf{q}}^w_v$, to the closest line feature in the map, $\mathbf{l}^w_{m,j}$. λ is a composite of two functions,

$$\lambda\left(\hat{\mathbf{q}}^w_v, \mathbf{l}^w_{m,j}\right) = \lambda^*\left(g\left(\hat{\mathbf{q}}^w_v\right), \mathbf{l}^w_{m,j}\right), \quad (8)$$

where $g\left(\hat{\mathbf{q}}^w_v\right) = \hat{\mathbf{q}}^\mathbf{w}_\mathbf{s}$ again transforms the world pose of the vehicle to the world pose of the sensor, and λ^* transforms a map line from the world frame to the sensor frame, $\lambda^*\left(\hat{\mathbf{q}}^w_s, \mathbf{l}^w_{m,j}\right) = \hat{\mathbf{l}}^\mathbf{s}$. Note that λ is a composite of three functions total. When solving for the Jacobian, $\mathbf{\Lambda} = \frac{\partial \lambda}{\partial \mathbf{q}}$, it is necessary to solve separately for the two cases in λ^*_2.

Data association is made easy with the use of the polygons (Fig. 5). We assume the state estimate is good enough to tell us which polygon the vehicle is in. If the pose is outside of a polygon, then the robot is at the end of a row and the line fit is probably bad, so we throw the measurement out. (This is OK because when outside of a polygon, the robot is near the point landmarks.) Otherwise we determine which polygon the pose is in, whether the measurement is to the left or right of the robot using α, and then associate to the corresponding line in that polygon. Note that this constrains us to only use line detections from the current row. Sometimes the lasers see through the trees to other rows in the block, and we throw out these measurements by gating on their d values. We gate the innovation separately on d and α with appropriate values based on sensor characterization (see Section IV-E.) This allows us to make the appropriate decisions in the data association, as well as to further rule out noisy outliers and bad detections.

E. Noise Modeling

The three steps of the filter each have covariance matrices that model the noise on the sensors being used. To determine the parameters for these models, we manually drove the robot up and down the rows of an orchard site that had already been mapped, while collecting sensor measurements and ground truth poses from the Applanix. Note that we only have to recalculate these parameters when the hardware on the vehicle is altered. Using the values from one orchard site generalizes to all other sites used in experimentation.

For the prediction step, we model the noise on our forward and angular velocities, $\mathbf{U}$. We computed the velocities by differentiating both the encoder readings and the Applanix poses. We then computed the covariance of the error between the two sets of values. The motion of the vehicle is separated into two cases: 1) when it is driving straight down a row, and 2) when it is turning at the end of a row. We compute the covariance separately for these cases, giving us $\mathbf{U_1}$ and $\mathbf{U_2}$. The variance on the angular velocity was much higher for the turning case, which is to be expected. Whether the vehicle is inside or outside of a polygon tells us whether the vehicle is going straight or turning. This is used here in modelling the noise, as well as in the filter itself for deciding whether to use $\mathbf{U_1}$ or $\mathbf{U_2}$ for each prediction step.

For the point feature measurement model, we collected range and bearing measurements, $\mathbf{z}$, at each time step. We mapped our Applanix poses through h to get the expected measurements at these time steps. We then computed the covariance of the innovations, $\left(\mathbf{z} - h\left(\mathbf{q}_v^w, \mathbf{p}_{m,i}^w\right)\right)$, between the two sets of values to use for the covariance matrix $\mathbf{R}_p$.

For the line measurement model, we collected line measurements, $\mathbf{l}^s$, at each time step. We mapped our Applanix poses through λ to get the expected measurements. We then computed the covariance of the innovations, $\left(\mathbf{l}^s - \lambda\left(\mathbf{q}_v^w, \mathbf{l}_{m,j}^w\right)\right)$, to use for the noise model $\mathbf{R}_l$.

V. EXPERIMENTS AND RESULTS

The point and line feature based EKF localization algorithm was implemented on our robotic vehicle and tested in a variety of orchard sites. For each test block, we place reflective tape at the end posts of the rows (refer to 4.) The experimental procedure was to first collect a large data set that could be used to build the map. After this, experiments were run to test the localization algorithm running online.

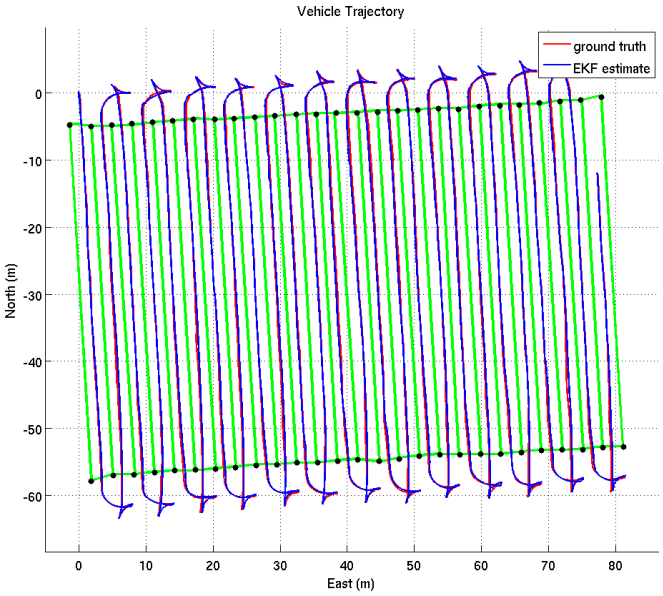

Fig. 5. Top down view of a trajectory, as the robot drives through an orchard block. The blue line shows the localization estimate, and the red line (barely visible) shows the Applanix estimate for ground truth comparison. Black dots at the end of the rows designate the point feature landmarks in the map. Green line segments connecting these points designate line features, as well as polygons for data association. (Sunrise Orchards, Rock Island, WA, 07/23/2010)

The robot is initialized at a known starting location, and then driven up and down the orchard rows, making k-turns to go from one row to the next. For some of the experiments the robot was manually driven by a human operator, and in other experiments the robot was driving autonomously. The localization module was abstracted from the system so that it could run seamlessly in either mode.

Fig. 5 shows a top-down plot of one of our trials. The blue line shows the localization estimate of the vehicle trajectory, and the red line (barely visible) shows the corresponding ground truth values. The black dots show the placement of the reflective landmarks and the green line segments show the corresponding line features and data association polygons.

Fig. 6 shows the same trial as Fig. 5, except with the dead reckoning estimate plotted as well. The dead reckoning is what we would get if we just ran the prediction steps of the filter without the measurement update steps. The dead reckoning relies on information only from the encoders, and it can be seen here that small errors in the steering wheel encoder are magnified over distances. This is meant to give the reader an appreciation for how the software algorithms presented here overcome the challenges of interfacing with cheap sensing hardware.

To evaluate performance, our primary metric for error is the Euclidean distance between the estimate and the ground truth at each time step. We look separately at the crosstrack and downtrack dimensions of this error. Fig. 7 shows histograms of these distributions for a typical run. When a robot drives in a straight line, its dead reckoning crosstrack error will usually accumulate more quickly than the downtrack, due to the propagation of small errors in heading. In our filter, the crosstrack error is corrected by line measurements, and therefore remains small. The downtrack error accumulates to some extent as the robot drives down

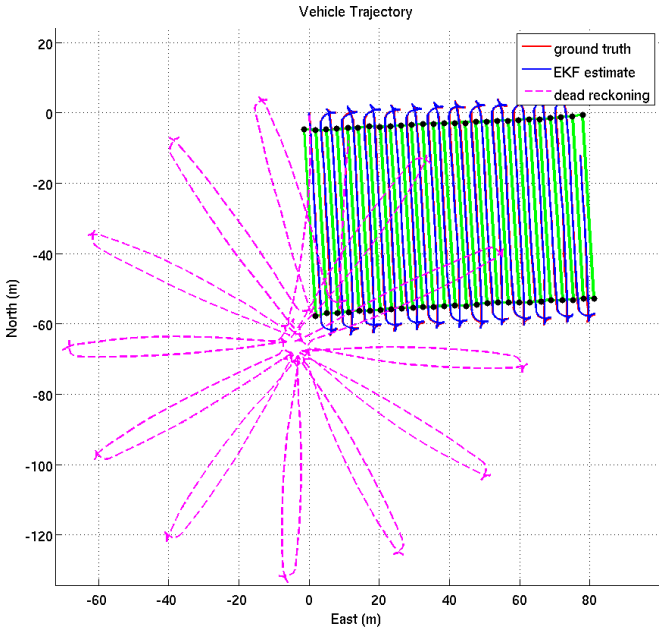

Fig. 6. Same trajectory shown in Fig. 5, except zoomed out and plotted with the dead reckoning estimate as well. This is meant to give the reader an appreciation for the limitations of the wheel encoder measurements, and how the filter corrections drastically improve the performance.

a row, but then gets corrected when observations to point features are made near the end of the row. The point features continue to help the robot as it makes the k-turn at the end of the row, and then both point and line features help the entry into the next row run smoothly.

Fig. 8 shows an example of an extreme case of downtrack error accumulation, as the vehicle is nearing the end of a very long row. As it nears the end, it picks up a reading

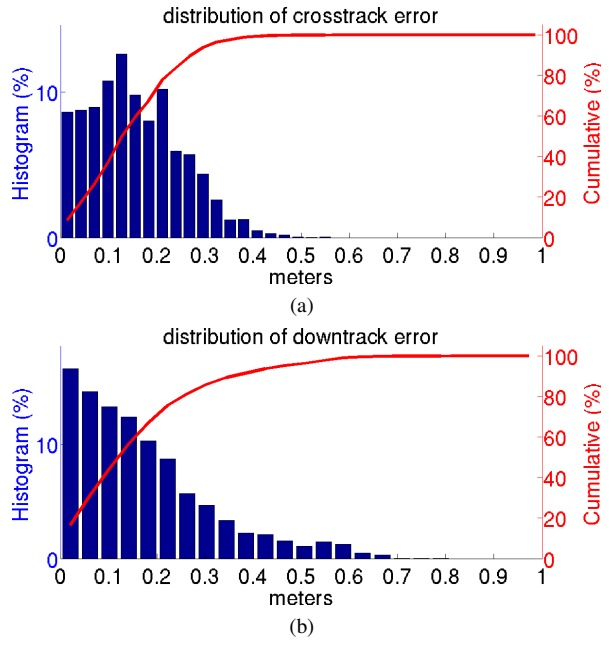

Fig. 7. Distributions of position errors of our localization estimate compared with ground truth from the Applanix. (A typical run, from Sunrise Orchards, Rock Island, WA, 07/23/2010)

to a reflective landmark one row over to the right. Because the filter estimate has so much error, the innovation in the measurement will be very large, and has the potential not to pass the data association gate. However, because the noise was modelled correctly, the state covariance ellipse will characterize the downtrack error, and the Mahalobis distance of the innovation will pass under the gate.

The experiment was repeated in multiple orchards with varying characteristics, and the results of these experiments are summarized in Table I. In total, sub-meter mean performance was demonstrated in over 20km of in-orchard operations. It is evident that the downtrack error increases as the row length increases, due to the dead reckoning accumulation in the middle of the row. The filter is still able to correct for itself at the ends of the rows with observations to point features. The orchards used in this testing represent the varying environments that are common in the apple industry. Canopies ranged from neatly manicured fruit-walls to uneven bushy rows. Row lengths varied from 53 to 345 meters. Terrain varied from hilly to flat. Two of the trials were conducted autonomously, and in the rest the vehicle was driven manually. In spite of all these differences, the EKF algorithm performed well in all environments without any need to tune the various EKF parameters.

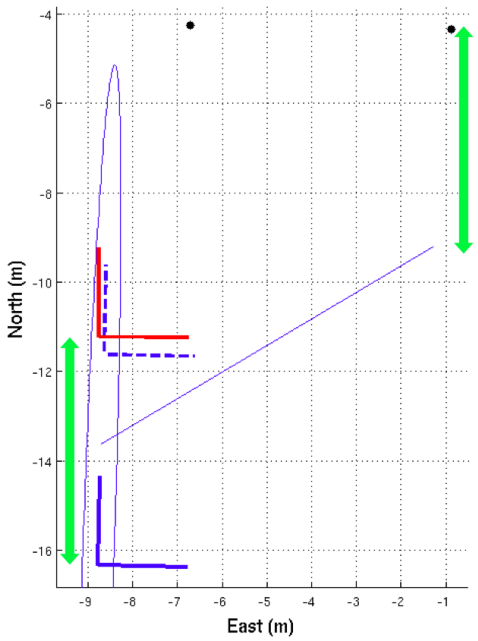

Fig. 8. This diagram shows an extreme case of downtrack error, where correct noise modeling allows for data association to still be successful. The solid blue and red L shapes are coordinate frames of the filter estimate and ground truth poses, respectively, at time t. The dashed blue frame is the filter estimate at time $t+1$, after a point feature update step. The black dots are landmarks in the map. The thin blue line is a laser ray, ending at a reading to a landmark, as observed by the filter estimate at time t. The green arrow on the bottom left highlights the downtrack error at time t. The same green arrow is translated up to the laser reading, to highlight the difference between the actual landmark and where the filter expects to see a landmark. The top half of an ellipse is centered around the blue pose estimate at time t. This is a 3σ ellipse representing the state covariance $\mathbf{P}$. The fact that the red ground truth frame lies within this ellipse demonstrates that the noise has been modeled correctly to represent the state uncertainty. This in turn allows the landmark reading to pass the Mahalanobis gate during the data association step.

TABLE I

STATISTICS FOR EXPERIMENTS OVER THE SUMMER OF 2010

Test site	Total distance (km)	Mean crosstrack error (m)	Mean downtrack error (m)	3σ crosstrack error (m)	3σ downtrack error (m)	Row length (m)
FREC[a]	2.05	0.18	0.28	0.78	1.13	125
FREC[a]	2.00	0.14	0.34	0.60	1.82	125
Sunrise[b]	1.38	0.15	0.17	0.54	0.65	53
Sunrise[b]	1.78	0.18	0.16	0.62	0.61	53
Sunrise[b]	1.59	0.17	0.25	0.60	1.01	53
Skyline[c]	2.94	0.23	0.66	0.67	3.16	345
Skyline[c]	10.21	0.22	0.73	0.64	3.48	345

[a]FREC stands for Fruit Research and Extension Center, Biglerville, PA

[b]Sunrise is short for Sunrise Orchards, Rock Island, WA

[c]Skyline is short for Skyline East Orchards, Royal City, WA. In these trials the robot was driving autonomously.

VI. CONCLUSIONS AND FUTURE WORK

In this work, we demonstrated a point and line feature based EKF localization filter running online in challenging outdoor environments. We developed a GPS-free solution, with minimal sensing from two wheel encoders and two fixed lasers operating in horizontal planes. We used the Applanix system in preprocessing steps to model the noise parameters and build the map, as well as to benchmark our results. Our online algorithms worked separately from the Applanix, without any GPS inputs. We demonstrated our solutions through extensive experimentation in a variety of real-world conditions. By rigorously modeling the noise on our sensors and our platform, we were able to attain robust solutions without retuning any parameters between trials.

It should be noted that this 2D filter performs very well even in hilly terrain. The update steps use measurements detected to features within a local vicinity, so we can assume that the coordinates of the vehicle and these features lie on a flat plane, even if this plane is tilted due to a slope in the terrain. On a more global scale, hills could affect the dead reckoning estimate, but these errors can be corrected by the udpate step, as long as the hills are not extreme enough to affect the data association.

In future work, we plan to extend this to a full SLAM solution, where the map is not registered ahead of time. Now that we have modeled the challenging characteristics of our platform and the surrounding environment, we will be able to tackle this problem more robustly. We also plan to incorporate other sensors to help with natural point feature extraction from the environment. This could include cameras as well as spinning lasers or fixed lasers rotated at different angles to give us more 3D information. Currently, we wrap reflective tape around endposts, but other sensors could help with detecting those endposts without the tape. Of course, all of these techniques involve putting more infrastructure on the platform, which incurs cost and complexity.

Laser odometry is another area we plan to explore. We have done some preliminary work in matching features extracted from consecutive scans, but this involves having a very clean environment (i.e. unoccluded tree trunks) where the uncertainty of the data does not render this registration obsolete. We are optimistic about performing laser odometry at the end of rows, where either the end posts or the entire ends of rows can be used as features to improve the accuracy of the EKF prediction step, resulting in improved performance for the overall filter.

VII. ACKNOWLEDGMENTS

We thank all the members of our project, Comprehensive Automation for Specialty Crops. We would also like to thank the owners and managers of Soergel Orchards, PA, Skyline East Orchards, WA, the orchards at Penn State's Fruit Research and Extension Center, and Sunrise Orchards at Washington State's Tree Fruit and Research Commission, for allowing us to test in their facilities.

This work is supported by the US Department of Agriculture under the Specialty Crop Research Initiative, award number 2008-51180-04876.

REFERENCES

[1] M. O'Connor, T. Bell, G. Elkaim, and B. Parkinson, "Automatic Steering of Farm Vehicles Using GPS," in *3rd International Conference on Precision Agriculture*, 1996.

[2] S. Singh, T. Baugher, M. Bergerman, K. Ellis, B. Grocholsky, B. Hamner, J. Harper, G.-a. Hoheisel, L. Hull, V. Jones, G. Kantor, B. Kliethermes, H. Koselka, K. Lewis, J. Libby, W. Messner, H. Ngugi, J. Owen, J. Park, C. Seavert, W. Shi, and J. Teza, "Automation for Specialty Crops: A Comprehensive Strategy, Current Results, and Future Goals," in *The 4th IFAC International Workshop on Bio-Robotics, Information Technology and Intelligent Control for Bioproduction Systems (2009 IFAC Bio-Robotics IV Workshop)*, 2009.

[3] A. Kelly, O. Amidi, H. Herman, T. Pilarski, P. Rander, A. Stentz, N. Vallidis, and R. Warner, "Toward Reliable Off Road Autonomous Vehicles Operating in Challenging Environments," *The International Journal of Robotics Research*, vol. 25, pp. 5–6, 2006.

[4] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, B. Salesky, Y.-w. Seo, S. Singh, J. Snider, A. Stentz, W. R. Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, and D. Ferguson, "Autonomous Driving in Urban Environments : Boss and the Urban Challenge," *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[5] R. Madhavan and H. F. Durrant-Whyte, "Terrain-aided localization of autonomous ground vehicles," *Automation in Construction*, vol. 13, no. 1, pp. 83–100, 2004.

[6] J. Guivant, E. Nebot, and H. F. Durrant-Whyte, "Simultaneous Localization and Map Building Using Natural features in Outdoor Environments," *Robotics and Autonomous Systems*, vol. 20, no. 2-3, pp. 79–90, 2002.

[7] D. Sack and W. Burgard, "A Comparison of Methods for Line Extraction from Range Data," in *In Proc. of the 5th IFAC Symposium on Intelligent Autonomous Vehicles (IAV)*, Lisbon, Portugal, 2004.

[8] F. Lu and E. Milios, "Robot Pose Estimation in Unknown Environments by Matching 2D Range Scans," *Journal of Intelligent and Robotic Systems*, vol. 18, pp. 249–275, 1997.

[9] J. J. Leonard, P. M. Newman, R. J. Rikoski, J. Neira, and J. D. Tardos, "Towards Robust Data Association and Feature Modeling for Concurrent Mapping and Localization," *Springer Tracts in Advanced Robotics (STAR)*, vol. 6, pp. 7–20, 2003.

[10] O. C. J. Barawid, A. Mizuzhima, K. Ishii, and N. Noguchi, "Development of an Autonomous Navigation System using a Two-dimensional Laser Scanner in an Orchard Application," *Biosystems Engineering*, vol. 96, no. 2, pp. 139–149, 2007.

[11] A. Stentz, C. Dima, C. Wellington, H. Herman, and D. Stager, "A System for Semi-Autonomous Tractor Operations," *Autonomous Robots*, vol. 13, no. 1, pp. 87–103, 2002.

[12] Robot Operationg System (ROS) is open source software. [Online]. Available: http://www.ros.org/wiki/

[13] H. M. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementation*. MIT Press, 2005.

[14] B. Hamner, S. Singh, and M. Bergerman, "Improving Orchard Efficiency with Autonomous Utility Vehicles," in *ASABE Annual International Meeting*, Pittsburgh, PA, 2010.