Title
Row-sensing templates: A generic 3D sensor-based approach to robot localization with respect to orchard row centerlines

Journal
Journal of Field Robotics, 39(6)

ISSN
1556-4959

Authors
Fei, Zhenghao
Vougioukas, Stavros

Publication Date
2022-09-01

DOI
10.1002/rob.22072

Row-sensing Templates: A Generic 3D Sensor-based Approach to Robot Localization with Respect to Orchard Row Centerlines

Abstract:

Accurate robot localization relative to orchard row centerlines is essential for autonomous guidance where satellite signals are often obstructed by foliage. Existing sensor-based approaches rely on various *features* extracted from images and point clouds. However, any selected features are not available consistently, because the visual and geometrical characteristics of orchard rows change drastically when tree types, growth stages, canopy management practices, seasons, and weather conditions change.

In this work, we introduce a novel localization method that doesn't rely on features; instead, it relies on the concept of a *row-sensing template*, which is the expected observation of a 3D sensor traveling in an orchard row, when the sensor is anywhere on the centerline and perfectly aligned with it. First, the template is built using a few measurements, provided that the sensor's true pose with respect to the centerline is available. Then, during navigation, the best pose estimate (and its confidence) is estimated by maximizing the match between the template and the sensed point cloud using particle-filtering. The method can adapt to various orchards and conditions by re-building the template. Experiments were performed in a vineyard, and in an orchard in different seasons. Results showed that the lateral mean absolute error (MAE) was less than 3.6% of the row width, and heading MAE was less than 1.72°. Localization was robust, as errors didn't increase when less than 75% of measurement points were missing. The results indicate that template-based localization can provide a generic approach for accurate and robust localization in real-world orchards.

Keywords: Probabilistic, localization, orchards, agriculture

1 Introduction

Labor cost and an increasing farm labor shortage are two main drivers for developing and deploying mechanization and automation technologies in orchards and vineyards (Zhang, 2017; Charlton et al., 2019). A third, significant driver is the need to implement precision horticulture, i.e., executing operations such as spraying, thinning, pruning, and harvesting while taking into consideration an orchard's or vineyard's spatial and temporal variability. Precision horticulture can increase the efficiency of resources, and consequently reduce cost and negative ecological impact (Zude-Sasse et al., 2016).

Agricultural robots can help ease orchard labor shortages by either replacing workers in labor-intensive tasks like harvesting (e.g., Williams et al., 2020) or assisting human workers in various orchard production activities (e.g., harvesting, pruning, spraying, and mowing) to increase working efficiency (Zhang, 2017). For example, the utilization of an autonomous utility vehicle resulted in efficiency gains of up to 58% (Bergermann et al., 2015) for tasks such as pruning conducted on the top part of trees when compared with the same task performed on ladders. Agricultural robots can also provide advanced sensing, computation, and actuation that facilitates precision horticulture. Example applications include selective spraying, where chemical inputs are reduced dramatically (Zhang, 2017; Asaei et al., 2019) and selective pruning (Botterill et al., 2017).

Robot operation in orchards relies on accurate localization inside the rows of orchard blocks, and at the end-of-block headland spaces (Figure 1a). Inside orchard rows, robots travel along the row centerlines, and therefore, auto-guidance requires knowledge of the robot's *position along the row's centerline*, and its *lateral displacement and heading offsets relative to this centerline*. In the headlands, robots execute appropriate turning maneuvers to move to another row or move to another orchard block.

Unlike localization in open fields where cm-level accurate GNSS (Global Navigation Satellite System) signals are available, accurate and robust localization inside orchard rows cannot rely solely – or at all - on GNSS. The reason is that the foliage of tall trees (Figure 1a, b) often blocks GNSS signals or introduces multipath effects, rendering satellite-based localization with a moving GNSS receiver impossible or unreliable. This effect may not be as severe in the headlands between orchard blocks, although it can be present. Therefore, localization methods must utilize sensors that take local measurements of the surrounding environment (trees, ground, sky, irrigation lines, etc.) Visual cameras, depth or RGB-D cameras, and LiDARs are "3D sensors” that are commonly used in orchards to collect reflected energy from the surrounding environment, with high spatial resolution.

Figure 1. a) Left: Rows of high-density trellised apple trees belonging to an orchard block; headland space is shown in the front (Lodi, California, 2018); b) Right: Rows of almond trees (Winton, California, 2015, courtesy of UC ANR).

The rows of modern, commercial orchard blocks are characterized – to a large extent - by flat terrain and uniformity in tree types, shapes and sizes, and placement/spacing. When the sensor that is used for localization is near the ground level and below the treetops, the spatial distributions of the sensor measurements of the surrounding environment (referred to as ‘sensor readings’ for brevity) exhibit two main characteristics.

The first characteristic is that sensor readings do not change significantly when the sensor’s reference frame translates along/parallel to the orchard centerline (horizontal translation “invariance” of sensor readings). A major consequence of this characteristic is that absolute localization along the centerline - with respect to a reference entry point of the row - does not seem feasible/practical using local, near-ground sensing (see Shalev & Degani, (2020) for a different approach). For this reason, in the absence of reliable GNSS signals or artificial landmarks that could be used for absolute positioning, existing literature (Section 2) shows that researchers have used various forms of odometry (e.g., wheel, inertial, or vision-based) for localization along the centerline. The robot position corresponds to the distance traveled by the robot from a reference starting point on the centerline, i.e., the robot’s entry point in the current orchard row. It should be stressed that this characteristic becomes weaker as the robot approaches the end/edge of the current row, when its sensor can "see" and detect .

The second characteristic is that sensor readings depend on the sensor frame's rotation or lateral translation (offset) with respect to the centerline. This characteristic suggests that lateral and rotational localization with respect to the centerline are possible using only local sensing. In fact, the published methods for such localization rely on a stronger assumption on this characteristic, i.e., that selected features of the orchard environment are distributed *symmetrically* about the centerline, and that these features can be detected and localized reliably (see detailed literature review in section 2). These features are used to estimate the robot's lateral displacement and heading offsets relative to the centerline. Proposed features include the tree trunks on the left and right sides of the row, the intersections between trunks and ground, the sky region, the orchard floor, or the planes of flat, fruiting-wall type tree canopies.

The main shortcoming of feature-based methods is that they have not been shown to generalize well or be robust enough in different orchard settings. Orchards constitute diverse, complex, and dynamic environments. Tree type, age, placement, and architecture, as well as canopy and orchard floor management practices can severely affect the presence, appearance, and symmetry (about the centerline) of features. Seasonal variation (e.g., dormant vs. blooming trees), weather, and illumination conditions, missing trees also introduce variability in feature appearance, symmetry, or even availability. To the authors' knowledge, a general approach that does not rely on specific features, or centerline symmetry assumptions, and can adapt easily to a large range of orchard environments is not available.

The main contributions of this paper toward this goal are the following:
1) It introduces the concept of the 'row-sensing template' and utilizes it to develop a new generic and robust sensor-based method to estimate a vehicle's heading and lateral offset with respect to the row centerline in orchards; the method capitalizes on characteristic #1, but does not rely on features and symmetry assumptions.
2) It presents extensive experimental results in two different orchards and various seasons to evaluate the proposed method's localization accuracy and robustness, and also analyzes the

method's performance as the robot approaches the end of the row, where characteristic #1 is violated to increasing extent, i.e., sensor readings start varying under translation along the centerline.

Our localization method adopts a probabilistic framework, uses raw 3D point clouds, instead of features extracted from the raw data, and is shown to be generic and robust. The proposed method consists of two stages.

In its first stage, the method capitalizes on characteristic #1 (sensor data invariance under translation along an orchard row's centerline) and builds a uniform 3D grid that stores in each voxel the probability that this voxel is occupied, when the sensor is anywhere on the row's centerline and aligned to it. Essentially, the 3D grid – referred to as a "row-sensing template" or "template" – represents what the sensor expects to "sense" if it is placed at any point on the centerline, with its frame aligned with it; it is an occupancy grid (Elfes, 1989) for the space inside the sensor's field of view (not the world). The template is built using a small set of point cloud measurements. The major requirement during the template-building stage is that the sensor's lateral and heading offsets (ground truth) are known.

In its second stage, the method capitalizes on characteristic #2 and uses the template for Monte-Carlo localization in real-time. A static measurement model is implemented that returns the probability of the current point-cloud measurement given the template and a proposed pose. The vehicle pose is estimated as the pose that maximizes the probability of getting the observed measurement. The space of possible poses is searched by generating uniformly distributed random poses or by integrating the measurement model with a particle filter framework.

The rest of this paper is organized as follows: In section 2, we present the related works, and in section 3, we discuss in detail our proposed approach. Next, in section 4, we present the experimental platform and methods used for the experimental evaluation of the method. The results from our experiments are presented and discussed in section 5, and in section 6, we summarize our conclusions and discuss future work.

2 Related Work

Sensor-based localization inside orchard rows has been addressed by many researchers, with cameras (monocular and stereo) and LiDARs (2D and 3D) being the most commonly used sensors. The main idea behind the existing methods is to utilize specific visual or geometrical features or structures to estimate directly or indirectly the row's centerline and localize the sensor (robot) with respect to it. Barawid et al. (2007) used a 2D LiDAR scanner to detect tree trunks and the Hough-transform to extract the left and right tree lines independently; then, he computed the corresponding centerline to determine the pose of the vehicle. Similarly, He et al. (2010) proposed a machine vision algorithm to detect tree trunks and *the boundaries between the trunks and the ground, to* estimate tree row lines, and the corresponding row centerline. Hamner et al. (2011) used a 2D LiDAR to detect tree trunks and the Hough transform to detect right and left lines that are constrained to be parallel; the centerline was computed from them. Marden et al., (2014) estimated grapevine trunk lines using the RANSAC method and used these lines as features in a line-based EKF-SLAM framework; their method can simultaneously do localization and mapping (line map). Bell et al., (2016) used a 3D laser scanner is used to measure the positions of posts and trunks in pergola-structured orchards (e.g., for kiwis) and calculate the row direction and centerline. Lyu et al., (2018) also proposed a method to detect the boundaries between trunks and the ground and used a naive Bayesian classifier for the free space centerline detection. Durand-Petiteville et al., (2018) presented a stereo vision-based method to find tree trunks by detecting their "shadows," i.e., concavities in the range component of the obtained point cloud.

Other researchers have used the *ground*, *sky*, or *tree foliage* as features, and segmented them in image space to estimate the row's centerline. Subramanian et al., (2006) proposed to use RGB thresholding to segment the tree canopy in the image and find boundary lines; their method also combined a 2D lidar to detect a path using distance thresholding to increase their system robustness. Torres-Sospedra et al., (2011) used a multi-layer feedforward neural network to segment land/soil, sky, tree crown, and trunk areas in an image and then applied a Hough transformation on the borders between land and trees to determine the centerline of the path. Sharifi et al., (2015) improved the segmentation method by using the mean-shift algorithm to do clustering, along with a novel classification technique based on graph partitioning theory to classify clusters. Radcliffe, Cox & Bulanon (2018) used an upward looking camera to detect sky and tree canopy features for localization.

Zhang et al., (2013) proposed a method that utilizes 3D point clouds for localization. They divided the 3D point cloud into a left and a right set. Then, they randomly selected points in both sets to compute multiple pairs of parallel-line features and used RANSAC to get the pair with the smallest number of outliers. The heading was directly computed from the best pair of the parallel lines. However, the method needed an additional step to segment tree trunks and large branches from the point cloud to accurately determine the lateral offset, because trunks and branches generate denser and more stable LiDAR returns than leaves and grass.

All the above methods are applicable when the corresponding features they rely on are visible/detectable. As it was explained in Section 1, this may not be true in many cases. For example, tree trunks or trunk-ground intersections may not be visible; the sky may not be visible, or the orchard floor may be covered (Figure 2a, 2b, 2c). Blok et al., (2019) used a 2D LiDAR and a particle filtering approach for localization in orchard rows, without relying on features. The methods was found to be accurate and robust when some trees were missing. However, their probabilistic 2D LiDAR sensor model relied heavily on an *a-priori* model of the orchard structure (row and tree spacing and trunk sizes) at each side of the robot, which would have to be re-developed for different tree architectures and orchard spacings. Furthermore, approaches that use a 2D LiDAR as the main sensor can only get single-plane information in space. Because tree canopies are three dimensional and irregular, and ground can be uneven and have grass, methods using 2D LiDAR are not as robust.

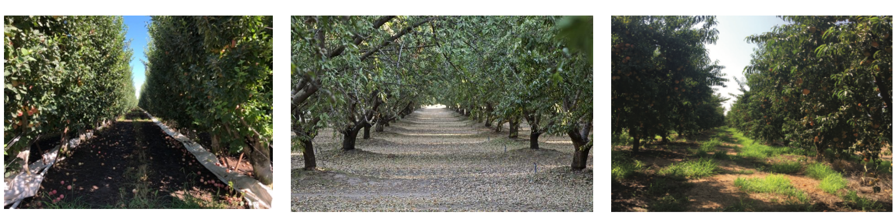


Figure 2. Left: The figure shows that apple tree trunks are hidden by foliage, and trunk-ground intersections are partially occluded by reflective tarps (Lodi, California, 2019); Center: The figure shows that the sky is not visible in an almond orchard (California, 2017); Right: The figure shows that the orchard floor is covered with patches of grass (Vougioukas, 2019).

As a conclusion, accurate and robust methods for localization with respect to orchard row centerlines are still needed. Such methods should not depend on over-simplifying, extensive or unrealistic assumptions about orchard structure or the presence of features, and should be applicable and easily adaptable in different types of orchards, and different seasons.

3 Template-based Localization

The proposed template-based localization method consists of two main stages. During the first template-building stage, a 3D sensor template **T** is constructed that represents what a 3D sensor would expect to perceive if it was placed on the row's centerline with zero lateral and heading offset. During this phase, point cloud measurements with corresponding known poses are needed to build the template. In the second phase, a novel template-based sensor measurement model is used in a particle filter framework for localization in real-time.

We also make the assumption that the orchard ground is relatively flat, the ground plane can be extracted in the point cloud measurement, so that the vehicle's roll and pitch can be easily recovered.

Future sensor measurements can correctly align with the template, even with some variations, because template is built in a probabilistic way, and the overall geometrical shape of the sensor measurement is taken into account.

Next, basic terms, symbols, and coordinate frames are introduced that will be used extensively in the rest of the paper. Let $\{R\}$ be the frame of the currently traversed orchard row. Its origin lies on the centerline of the row, between the first pair of trees at the beginning of the row (Figure 3). Its x-axis is the centerline of the row and points toward the other end of the row (forward), and its z-axis is perpendicular to the ground and points upward. $\{V\}$ is the vehicle frame, with its x-axis pointing forward, and the z-axis pointing upward. Let $\{C\}$ be the 3D sensor frame. In this work, for simplicity, $\{C\}$ coincides with $\{V\}$, and both may be used interchangeably. (In general, $\{C\}$ and $\{V\}$ are connected via a known rigid body transformation.)

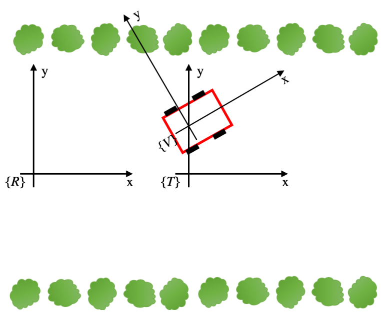


Figure 3 The figure shows the orchard row frame {R}, the vehicle frame {V}, and the template frame {T}.

An orchard row template **T** represents the expected 3D sensor measurement, when the sensor is on the centerline, and aligned to the row frame $\{R\}$'s x axis. Therefore, the row template frame $\{T\}$ – by definition - has the same orientation as $\{R\}$, and its origin lies on the centerline, and at the same x coordinate as $\{V\}$ with respect to $\{R\}$.

The template is represented as a uniform 3D grid made up of voxels defined over a continuous space in a template frame $\{T\}$; each voxel has a value that represents the occupancy frequency of that voxel. Each measurement is a point cloud triangulated from a pair of stereo images. At time t, let the nth point of the point cloud be denoted as ${}^{\{V\}}z_t^n$, and the set of all points, i.e., the point cloud itself be ${}^{\{V\}}Z_t$. Our goal is to estimate the vehicle's pose with respect to the row frame ${}^{\{R\}}X_t = [{}^{\{R\}}x_t, {}^{\{R\}}y_t, {}^{\{R\}}z_t, {}^{\{R\}}\alpha_t, {}^{\{R\}}\beta_t, {}^{\{R\}}\theta_t]$, where α, β, and θ are roll, pitch, and yaw, respectively. The distance along the centerline ${}^{\{R\}}x_t$ can be obtained by odometry and hence is outside the scope of this work. The quantities ${}^{\{R\}}z_t$, ${}^{\{R\}}\alpha_t$, ${}^{\{R\}}\beta_t$ are estimated by finding the ground plane as in section 3.1. The main focus of this paper is to estimate ${}^{\{R\}}y_t$ and ${}^{\{R\}}\theta_t$, using our template-based method. Next, the two stages of our method are presented in detail.

Stage 1: Given a set of point cloud measurements ${}^{\{V\}}Z_{t_1,t_2,\cdots t_n}$ with corresponding known lateral offsets ${}^{\{R\}}y_{t_1,t_2,\cdots t_n}$ and headings ${}^{\{R\}}\theta_{t_1,t_2,\cdots t_n}$ with respect to the centerline, build an row-sensing template $\mathbf{T}$.

Stage 2: Given a sequence of point cloud measurements ${}^{\{V\}}Z_{t_1,t_2,\cdots t_n}$, visual odometry information ${}^{\{R\}}u_{t_1,t_2,\cdots,t_n}$, and the pre-built orchard row template $\mathbf{T}$, compute the vehicle's lateral offset ${}^{\{R\}}y_t$ and heading ${}^{\{R\}}\theta_t$ with respect to the row's centerline, at each time step.

The computational pipelines of both stages are shown in Figure 4. Each individual module is explained in detail in this section.

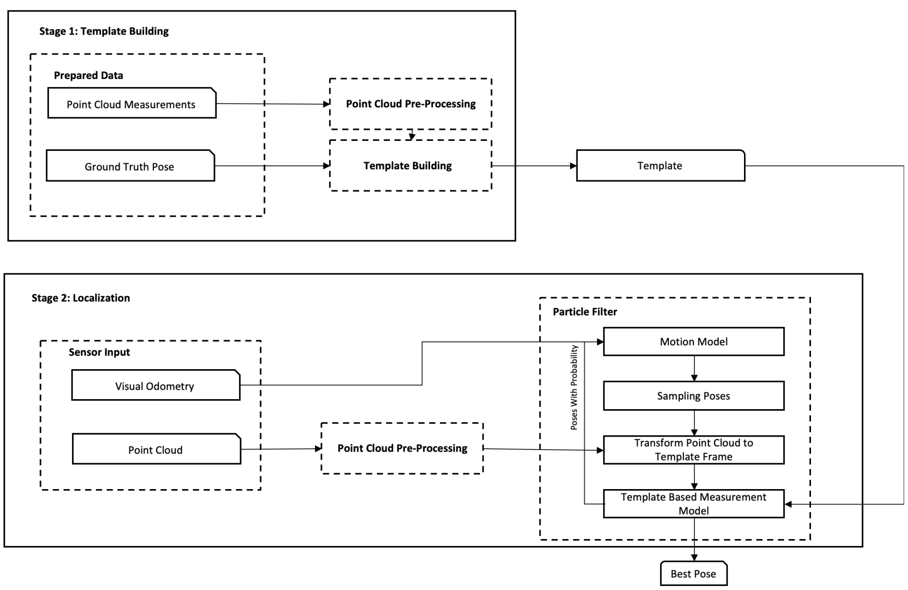


Figure 4 The figure shows two stages of the template-based localization approach ("Template Building" and "Localization") and their corresponding computational pipelines.

3.1 Stage 1: Template Building

The intuition behind using an orchard row template is that different tree-rows in the same orchard at a specific season share a tunnel-like structure that is invariant along the centerline. So the template T represents – in a 3D grid - what the sensor would expect to perceive if it were placed on the row's centerline with zero lateral and heading errors. Here we propose a way to build an orchard row template – starting from an empty template - using a set of measurements ${}^{\{C\}}Z_{t_1,t_2,\cdots t_n}$ with corresponding known lateral offsets ${}^{\{R\}}y_{t_1,t_2,\cdots t_n}$ and headings ${}^{\{R\}}\theta_{t_1,t_2,\cdots t_n}$. As a first step, each point cloud measurement ${}^{\{C\}}Z_t$ is input to the Point Cloud Pre-Processing module (Figure 5). In this module, the point cloud is first down-sampled using a voxel down-sampling method (Rusu et al., 2011). All points belonging to a voxel are represented by a single point – their centroid. And the point cloud is transferred into $\{V\}$.

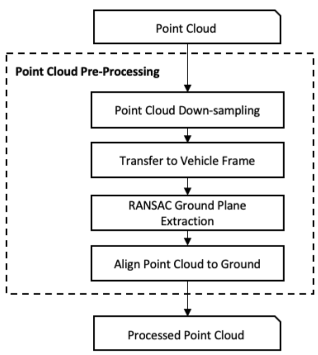


Figure 5 Point cloud pre-processing module

Then, RANSAC (Bolles et al., 1981) is used to estimate the ground plane and the vehicle's roll α_t, pitch β_t, and z states in $\{R\}$. Using the states estimated by RANSAC, the measurements ${}^{\{C\}}Z_t$ is transformed to the template frame $\{T\}$.

As a second step, all the points that do not belong to the current tree row and are outside the sensor's range are discarded by applying a spatial rectangular cutoff filter (**CutoffFilter**()) of appropriate dimensions set by the row width, height and sensor maximum range.

The third step (implemented in **UpdateTemplate**()) updates the contents of the template's voxels using the point cloud ${}^{\{T\}}Z_t$ by incrementing the current value of each voxel whose volume contains a point from ${}^{\{T\}}Z_t$. Then, the occupancy *frequency* of each voxel is computed as an estimate of the voxel's occupancy probability. The region out of the row range is undefined, there might be another row next to the current or not, so we fill all voxels within the template but not in the current row a no information frequency (*noInfoFrequency*). The entire process is given in Algorithm 1. Figure 6 shows the visualization of a row template.

Algorithm 1 BuildRowTemplate

Input: ${}^{\{C\}}Z_{t_1,t_2,\cdots t_n}$, ${}^{\{R\}}[y,\theta]_{t_1,t_2,\cdots,t_n}$,
templateResolution, templateRange, rowRange, noInfoFrequency
Output: template **T**
Create an all zero 3D grid template **T** with templateResolution and templateRange
${}^{\{V\}}Z_{t_1,t_2,\cdots t_n}$, ${}^{\{R\}}[\alpha,\beta,z]_{t_1,t_2,\cdots,t_n}$ = **PointCloudPreProcess**(${}^{\{C\}}Z_{t_1,t_2,\cdots t_n}$)
for $t = 1 : n$ **do**
  $R_t =$ **RotationMatrixFromEuler**($\alpha_t, \beta_t, \theta_t$)
  $t_t = [0, {}^{\{R\}}y_t, {}^{\{R\}}z_t]$ {translation vector}
  ${}^{\{V\}}T_{\{T\}} \in SE(3) = [R_t|t_t]$
  ${}^{\{T\}}Z_t = {}^{\{T\}}T_{\{V\}}{}^{\{V\}}Z_t = {{}^{\{V\}}T_{\{T\}}}^{-1}{}^{\{V\}}Z_t$
  ${}^{\{T\}}Z_t =$ **CutoffFilter**(${}^{\{T\}}Z_t$, rowRange)
  T = **UpdateTemplate**(T, ${}^{\{T\}}Z_t$)
end for
T = $\frac{T}{n}$ {convert count map to frequency map}
Fill out of rowRange region of the **T** with noInfoFrequency
return T

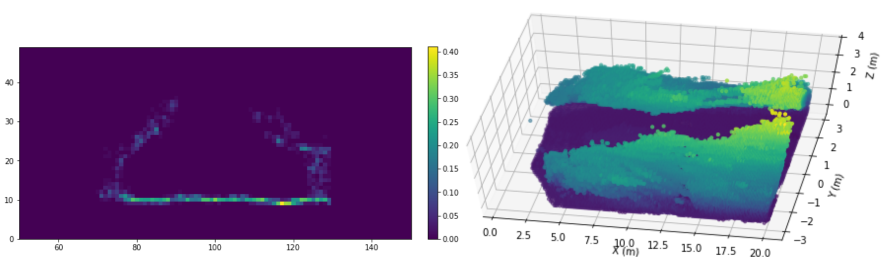


Figure 6 Left: An example of a cross-section (y-z) of a template T at distance from the sensor, x = 12 m; the color of each voxel represents the occupancy frequency; Right: 3D visualization of the template with voxel occupancy frequency larger than 0.02.

3.2 Stage 2: Localization using a template

3.2.1 Measurement Model

The measurement model (or sensor model) is the probability $P({}^{\{V\}}Z \,|X, \mathbf{T})$ of getting a point cloud measurement ${}^{\{V\}}Z$, given the vehicle pose X and the template $\mathbf{T}$ (Thrun et al., 2005). This probability can be obtained as the product of probabilities of all individual points under the assumption that individual point measurements are independent, given the vehicle pose X and the template $\mathbf{T}$.

$$P({}^{\{V\}}Z|X, \mathbf{T}) = \Pi_{k=1}^{n} P({}^{\{V\}}z^{k}|X, \mathbf{T}) \quad (1)$$

The template $\mathbf{T}$ is built in a way that each voxel is an estimate of the probability of the sensor to get a measurement in that voxel in frame $\{T\}$. So $\mathbf{T}$ is used as a likelihood field that can be indexed to get the individual measurement probability $P({}^{\{T\}}z^{k}|\mathbf{T})$. Each individual measurement probability $P({}^{\{V\}}z|X, \mathbf{T})$ can be calculated by transforming ${}^{\{V\}}z$ to ${}^{\{T\}}z$ and performing a table lookup for the probability in $\mathbf{T}$. The algorithm of this measurement model is shown in Algorithm 2. Figure 7 shows $P({}^{\{V\}}z^{k}|X, \mathbf{T})$ for each point in the measurement given a good pose proposal X and a bad pose proposal; the overall point measurement probability is higher (brighter) for a good pose proposal.

Algorithm 2 MeasurementModel

Input: measurement: ${}^{\{C\}}Z_t$, template $\mathbf{T}$, pose: ${}^{\{R\}}X$, range
Output: likelihood of ${}^{\{R\}}X$
${}^{\{V\}}Z_t$, ${}^{\{R\}}[\alpha, \beta, z]_t$ = **PointCloudPreProcess**(${}^{\{C\}}Z_t$)
R_t = **RotationMatrixFromEuler**(${}^{\{R\}}\alpha_t$, ${}^{\{R\}}\beta_t$, ${}^{\{R\}}\theta_t$)
$t_t = [0, {}^{\{R\}}y_t, {}^{\{R\}}z_t]$ {translation vector}
${}^{\{V\}}T_{\{T\}} \in SE(3) = [R_t|t_t]$
${}^{\{T\}}Z_t = {}^{\{T\}}T_{\{V\}}{}^{\{V\}}Z_t = {}^{\{V\}}T_{\{T\}}{}^{-1}\,{}^{\{V\}}Z_t$
${}^{\{T\}}Z_t$ =**CutoffFilter**(${}^{\{T\}}Z_t$, range)
q = 1
for $k = 1 : n$ **do**
  p = retrieve probability of the point ${}^{\{T\}}Z_t{}^{k}$ from $\mathbf{T}$
  q = q * p
end for
return q

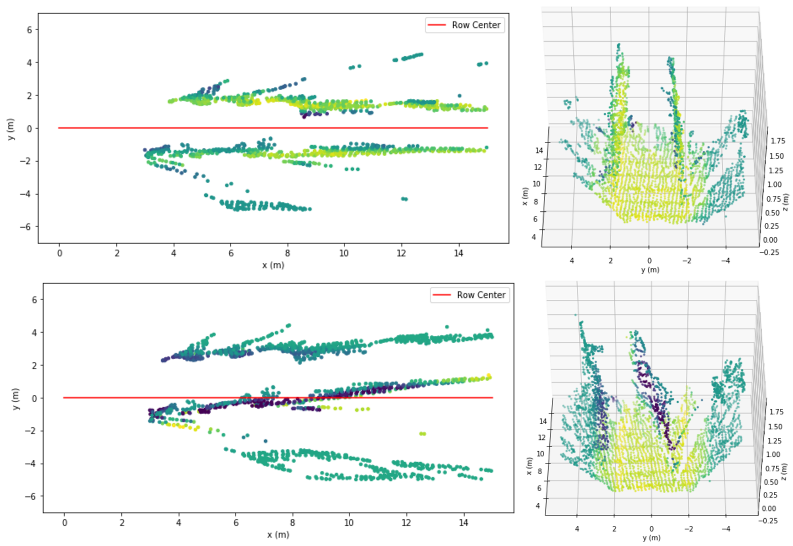


Figure 7 Top: Top-down view (left) and 3D view (right) of $P\left({}^{\{V\}}z^k \middle| X, T\right)$ *for all the points in* ${}^{\{V\}}Z$ *of a good pose proposal. Bottom: Top-down view (left) and 3D view (right) of* $P\left({}^{\{V\}}z^k \middle| X, T\right)$ *for all the points in* ${}^{\{V\}}Z$ *of a bad pose proposal. The brighter a point, the more likely this measurement point is correct.*

3.2.2 Monte Carlo Localization

Given the measurement model $P\left({}^{\{V\}}Z \middle| X, \mathbf{T}\right)$, a template $\mathbf{T}$ and a measurement ${}^{\{V\}}Z_t$, the Monte Carlo (aka Particle Filter) Localization framework is used to estimate vehicle pose X_t (Thrun et al., 2005). Under this framework, *n* multiple possible poses $X_t^{[i]}$ are sampled from a distribution $\mathbf{D}$, to generate the set of sampled poses $\mathbf{X}_t = \left[X_t^{[1]}, X_t^{[2]} \ldots, X_t^{[n]}\right]$. The likelihood of each possible pose in this set is evaluated, and the pose with maximum likelihood is selected.

$$\hat{X}_t = \underset{X \in \mathbf{X}_t}{arg\,max} P\left({}^{\{V\}}Z_t \middle| X, \mathbf{T}\right) \tag{2}$$

An obvious choice for **D** is a uniform distribution in frame $\{T\}$ (e.g., $y \in \mathrm{U}[-0.8m, 0.8m]$, $\theta \in \mathrm{U}[-0.6rad, 0.6rad]$). Figure 8 shows an example of the likelihood field of $X \in \mathbf{D}$, where **D** is a uniform distribution.

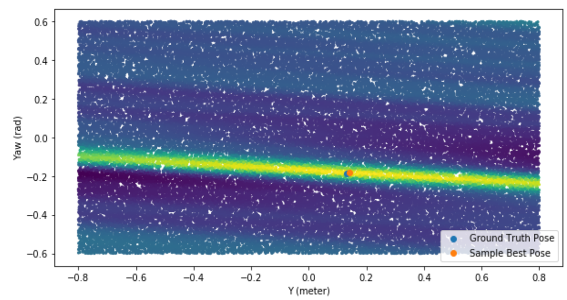


Figure 8 The figure shows a pose likelihood field in y-θ space with 50,000 sampled poses, given a measurement ${}^{\{V\}}Z$ and a template T; brighter color represent higher probability.

If the vehicle can get other sources of localization or motion information (e.g., control input, wheel odometry, steering angle sensor, GNSS, and visual odometry), the possible poses can be sampled from a distribution that is informed by other sensors. In this work, visual odometry was used as an additional motion information source (Cvišic et al., 2017) because of its accuracy and the fact that it requires no additional sensor hardware than the already available stereo camera. Monte Carlo Localization with visual odometry is presented in Algorithm 3.

The algorithm also estimates the covariance matrix of the estimated lateral offset and yaw by sampling 1% of the particles with the highest weights before resampling and calculating their covariance around the best pose estimate. The intuition behind this approach is that if 1% of best candidate particles are concentrated around the best pose estimate, this best pose estimate is more likely to be accurate, and the solution uncertainty (variance) is small. If 1% of the best candidate particles are spread out, the quality of the best pose guess tends to be low.

Algorithm 3 MonteCarloLocalization

Input: measurement: ${}^{\{C\}}Z_t$, a set of particles: $\mathbf{X}_t$, odometry input: u_t, odometry noise: Σ_t, template $\mathbf{T}$, range

Output: $\hat{X}_{t+1}$, $\mathbf{X}_{t+1}$

for $k = 1 : n$ **do**

  $< X_t^k, w_t^k >= \mathbf{X}_t^k$

  $X_{t+1}^k =$ **SampleMotionModel**(u_t, Σ_t, X_t^k) {movement step}

  $w_{t+1}^k =$ **MeasurementModel**$({}^{\{C\}}Z_{t+1}, \mathbf{T}, X_{t+1}^k$, range$)$ {measurement step}

  $\mathbf{X}_{t+1}^k =< X_{t+1}^k, w_{t+1}^k >$

end for

$i = \underset{1 \le k \le n}{\operatorname{argmax}} w_{t+1}^k$

$\hat{X}_{t+1} = X_{t+1}^i$

$\Sigma_{t+1} =$ **CalculateCovarianceMatrix**$(\mathbf{X}_{t+1})$

$\mathbf{X}_{t+1} =$ **Resample**$(\mathbf{X}_{t+1})$ {resampling step}

return $\hat{X}_{t+1}, \mathbf{X}_{t+1}, \Sigma_{t+1}$

4 Experimental Design

The goals of the experiments were to evaluate the accuracy of the template-based localization method in different orchards and seasons (section 5.1); to evaluate the robustness of its accuracy against different template instances (section 5.2), and against mismatches between a template and traversed rows due to gaps in the tree rows (from missing or smaller trees) (section 5.3), and examine the localization accuracy as the sensor reaches the end of the row (section 5.4). Finally, the effect of the number of measurements used to build the template on the accuracy was investigated (section 5.5).

Experiments were conducted using a 3D camera in several rows, in a vineyard (L. *Vitis vinifera*) in spring, and in an apricot orchard (L. *Prunus armeniaca*), in different seasons. The metrics used to evaluate the accuracy of the lateral offset and heading with respect to the row centerline were the mean absolute error (MAE), the standard deviation (SD) of the absolute error, and the 95th percentile of the absolute error. Next, the experimental platform, the experimental design, and the ground truth generation process are presented in detail.

4.1 Experimental platform

The sensor used in this research was a low-cost ZED stereo camera (Stereolabs Inc, San Francisco, CA). The field of view of this sensor is 90° (H) x 60° (V) x 110° (D) max, and the baseline of the stereo camera pair is 120 mm. It can output point clouds produced by stereo triangulation at 30 frames per second with 1080P resolution, using an NVIDIA GPU; it also provides visual odometry at the same rate and supports communication via ROS (Robot Operating System). Our localization method can work with one single 3D camera without other sensors, which largely simplifies the overall system complexity and reduces the cost. If additional sensors are available, such as IMU (Inertial Measurement Unit) and wheel odometry, they can also be integrated with the template-based measurement model, and provide more informed sampling in the Monte Carlo localization framework.

A locally developed mobile robot was used as a mobile platform for data collection (Figure 9). The ZED stereo camera was mounted in the front center of the robot, facing forward. Two RTK-GNSS receivers provided ground truth for the position and heading in the vineyard. Ground truth in the orchard was measured using a different approach (see Section 3) because GNSS signals or RTK corrections were not available due to the foliage of large trees.

Figure 9 The mobile robot that was used as our experimental platform.

In all experiments, the robot traveled at a speed of 1 m/s. While traveling inside each row (vineyard or orchard), the robot was controlled remotely and steered to travel on a sinusoidally-shaped path, in order to sample the widest possible (collision-free) ranges for offset and yaw deviations from the centerline.

4.2 Experiments in a vineyard

Localization experiments were performed in a vineyard at Davis, California, during the spring season, 2019 (Figure 10a). Ten random rows were traversed with the robot platform. The average row spacing was 3 meters, and vines were planted at 1.8 meters apart in average. The UTM (Universal Traverse Mercator) coordinates of the endpoints of ten vine rows were measured with an RTK GPS receiver; each row was approximately 90 meters long. Ground truth for the position and orientation of each row's centerline was computed from the measured row endpoints.

4.3 Experiments in a tree orchard

Localization experiments were also performed inside two rows of apricots trees, at Davis, California during 2018 and 2019. Experiments included traversal of the rows in the winter when trees were dormant and had no foliage (Figure 10b); in the summer, with dense foliage (Figure 10c), and in the spring with sparse foliage (Figure 10d). The average row spacing was 5 meters, and trees were planted at 2.5 meters apart in average; each row was approximately 50 meters long.

a) 

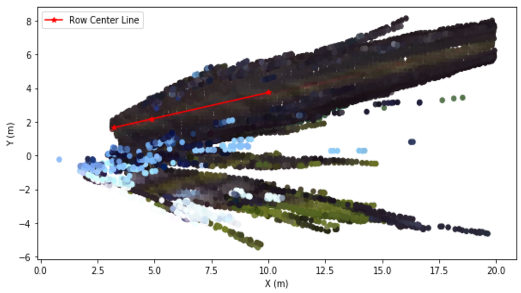
b) 

c)  d) 

Figure 10 a) Vineyard in spring; apricot tree orchard in b) winter, c) summer, and d) spring.

Inside the tree rows, the RTK-GNSS did not provide reliable localization because of tall canopies and foliage. As an alternative way to generate the ground truth for the sensor's offset and yaw with respect to the row's centerline, a physical centerline was used. A visually prominent colored rope was placed – and stretched - along the center of the orchard row (Figure 11a), and an image processing pipeline was developed to detect and localize it. The pipeline included three steps; 1) detect the rope as a line in the camera's image space (Figure 11b); 2) project the detected line back onto the physical ground plane in the camera frame, using a well-calibrated camera projection matrix (Figure 11c); 3) compute the lateral offset and heading angle of the camera relative to the rope-defined centerline. The template-based localization algorithm does not use any color information, so this colored rope did not affect the algorithm's performance. We pre-evaluated the localization accuracy of this rope-based method in a field where RTK-GNNS was available. The y-offset difference was 0 ± 0.012(SD) m, and the heading difference was 0 ± 0.00043(SD) rad between the rope-based method and RTK-GNSS.

a)  b) 



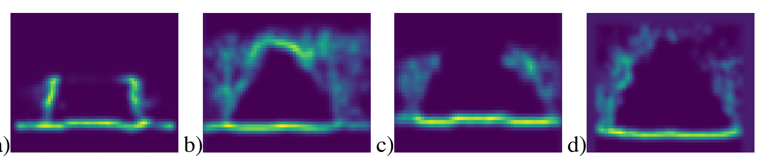
c)

Figure 11 a) The figure shows the red rope in camera's image view; b) the blue line is the extracted line in the image space; c) the red line is the extracted line projected onto the ground plane (top-down view of the corresponding point cloud).

5 Experimental Results

5.1 Localization accuracy

We call a specific orchard at a specific season as an operational scenario. An orchard row template is valid for a whole operational scenario. We built the template using 100 continuous measurements (~7 seconds) with ground truth for each operational scenario. Figure 12 shows the middle slices of the 3D templates generated for our experiments.



a) b) c) d)

Figure 12 Visualization of the middle slice of templates generated in a) vineyard in spring and apricot tree orchard in b) winter, c) summer, and d) spring.

For all experiments, we ran our algorithm offline with and without visual odometry information. Without visual odometry, our particle filter sampled poses from a uniform distribution in the $\{T\}$ frame ($y \in \mathrm{U}[-0.8, 0.8]$ m, $\theta \in \mathrm{U}[-0.6, 0.6]$ rads). With visual odometry, the Monte-Carlo sampling procedure in algorithm 3 was used. An example of the final localization output in a vineyard row with visual odometry is shown in Figure 13.

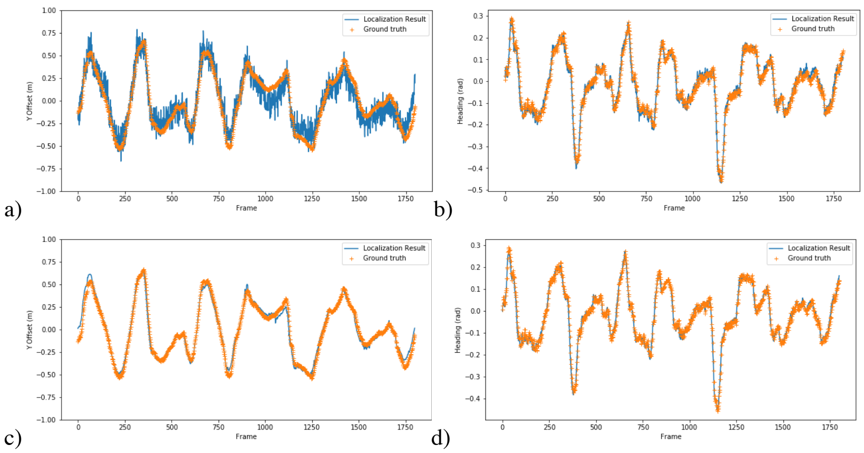


Figure 13 The localization results of lateral offset (a) and the heading (b) with uniform sampling are overlaid with ground truth. The localization results of lateral offset (c) and the heading (d) with visual odometry informed sampling are overlaid with ground truth.

The overall localization results for all operational scenarios and also for each run are reported in Table 1.

Operational Scenario	Visual Odometry Informed Sampling								Uniform Sampling							
	Y error (meter)					Yaw error (rad)			Y error (meter)					Yaw error (rad)		
	MAE†	SD†	95%†	MAE / Row spacing†	95% / Row spacing	MAE	SD	95%	MAE	SD	95%	MAE / Row spacing	95% / Row spacing	MAE	SD	95%
Vineyard	0.03	0.02	0.07	1.0%	2.5%	0.01	0.01	0.04	0.10	0.07	0.24	3.2%	8.0%	0.02	0.01	0.04
Apricots Winter	0.09	0.08	0.21	1.8%	4.3%	0.03	0.03	0.07	0.17	0.17	0.47	3.3%	9.4%	0.02	0.03	0.04
Apricots Spring	0.07	0.06	0.20	1.5%	4.1%	0.02	0.01	0.04	0.18	0.14	0.46	3.5%	9.2%	0.02	0.04	0.04
Apricots Summer	0.09	0.06	0.20	1.8%	4.0%	0.02	0.02	0.05	0.18	0.14	0.46	3.6%	9.2%	0.02	0.02	0.05

† MAE: Mean Absolute Error; SD: Standard Deviation; 95%: 95th Percentile;

† † The average row spacing is 3 meters for the vineyard and 5 meters the apricots orchard

Table 1 Localization results with and without visual odometry, for each scenario. The average row spacing is 3 meters the vineyard and 5 meters for the apricots orchard.

In all the operational scenarios, our method localized the vehicle with heading MAE below 0.03 rad and lateral MAE below 5% of the row spacing, without visual odometry. When visual odometry was used for informed sampling, the lateral MAE dropped below 2% of the row spacing. The results were consistent across different orchards and seasons.

5.2 Localization robustness against template instance

An important assumption in the proposed method is that a template built from a set of data from one or more rows can be used for localization in all rows – in the same orchard - without significant loss in localization accuracy. To evaluate the validity of this assumption, a template instance that was generated using 100 consecutive measurements while moving inside row k (for each $k = 0, 1, \ldots, 9$) was used to localize the robot – with and without visual odometry informed sampling - in evaluation runs inside all the ten rows in the vineyard block. The localization MAEs - with and without visual odometry - are given in Figure 14a and Figure 14b, respectively.

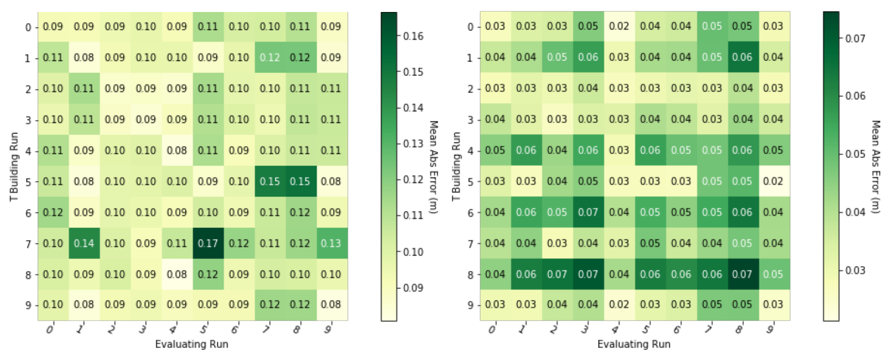


Figure 14 Localization errors (MAE) when a template generated using data from row k was used to localize the robot in an evaluation run in row j ($0 \leq k, j \leq 9$); lighter colors correspond to better accuracy. a) Left: Monte-Carlo localization with uniform sampling. b) Right: Monte-Carlo localization with informed sampling from visual odometry.

The elements (k, k) on the main diagonals corresponded to localization errors when the template that was built with data from row k was used for localization inside the same row k. When using uniform sampling, the maximum MAE was 0.11 m, whereas the off-diagonal MAE was 0.17 m. However, when using visual odometry informed sampling, the maximum MAE was 0.07 m for both the on and off-diagonal elements. Also, some templates resulted in overall slightly better results than others (e.g., in the right matrix, row #2 is much lighter-colored than row #8), Table 2 presents detailed localization errors when all vineyard rows were traversed using a template based on measurements from row #2.

Run ID	Visual Odometry Informed Sampling								Uniform Sampling							
	Y error (meter)					Yaw error (rad)			Y error (meter)					Yaw error (rad)		
	MAE†	SD†	95%†	MAE / Row spacing†	95% / Row spacing	MAE	SD	95%	MAE	SD	95%	MAE / Row spacing	95% / Row spacing	MAE	SD	95%
0	0.03	0.02	0.07	0.9%	2.5%	0.01	0.01	0.04	0.10	0.07	0.24	3.2%	7.9%	0.02	0.01	0.04
1	0.03	0.02	0.08	1.1%	2.6%	0.02	0.01	0.04	0.11	0.08	0.26	3.5%	8.8%	0.02	0.01	0.05
2†	0.03	0.03	0.08	1.0%	2.7%	0.02	0.01	0.04	0.08	0.07	0.21	2.7%	6.9%	0.02	0.01	0.04
3	0.05	0.03	0.11	1.6%	3.8%	0.02	0.02	0.05	0.08	0.06	0.20	2.7%	6.7%	0.02	0.02	0.05
4	0.03	0.03	0.08	1.0%	2.7%	0.02	0.02	0.05	0.08	0.07	0.21	2.8%	7.1%	0.02	0.02	0.05
5	0.03	0.02	0.07	1.1%	2.5%	0.01	0.01	0.03	0.11	0.09	0.28	3.6%	9.2%	0.01	0.01	0.03
6	0.03	0.03	0.08	1.2%	2.8%	0.01	0.01	0.03	0.10	0.08	0.25	3.2%	8.3%	0.02	0.01	0.04
7	0.04	0.03	0.10	1.4%	3.3%	0.02	0.02	0.05	0.10	0.08	0.24	3.3%	8.1%	0.02	0.01	0.05
8	0.04	0.03	0.11	1.4%	3.6%	0.02	0.02	0.05	0.10	0.08	0.25	3.5%	8.4%	0.02	0.01	0.05
9	0.03	0.02	0.06	0.8%	2.0%	0.01	0.01	0.03	0.10	0.08	0.25	3.3%	8.4%	0.02	0.01	0.04

† MAE: Mean Absolute Error; SD: Standard Deviation; 95%: 95 Percentile;
† The average row spacing is 3 meters for the vineyard.
† Template is built using data from run id 2

Table 2 The table shows accuracy results when a template that was built from measurements in row #2 of the vineyard is used for localization in all ten rows of the vineyard.

Overall, the above results suggested that a template developed from one row could be used for localization in other rows without significant loss of accuracy.

5.3 Localization robustness against gaps in rows

The template-based localization method is based on the assumption that when the 3D sensor is on the center line and aligned to it, the spatial distribution of the point cloud sensed anywhere along an orchard row matches the point cloud distribution of the template. However, in commercial orchards, it is very common that one or more trees are missing (e.g., due to disease) or are much smaller (because of replanting), thus creating "gaps" along the tree lines. Such gaps represent extreme cases/outliers of variability inside a row. Examples of missing trees from our data can be seen in Figure 15a. In the left image, one tree is missing on the left of the row, and in the right image, one tree is missing on both sides of the row. Figure 15b shows the top-down views of the corresponding 3D point clouds (sliced at 1.5 m height), transformed into the template frame $\{T\}$.

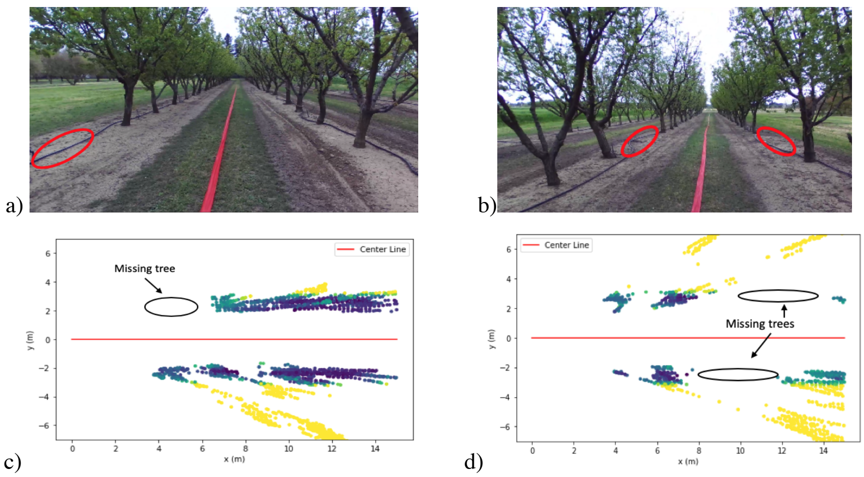


Figure 15 a, b) Camera views of orchard rows with gaps (missing trees) (red ellipses). c, b) Top-down view of the point cloud (excluding treetop and ground); black ellipses are gaps.

To evaluate the robustness of the template method in the presence of gaps, sets of points in the measurements were artificially removed, to simulate such gaps. A length of 1 m was used as a "unit length" for gaps in the point cloud data; this length is referred to as a "unit-tree." Smaller trees could result in one unit missing, whereas larger missing trees could result in more than one consecutive missing units. Since the 3D sensor used in this work had a range of 20 m, each side of the measurement was split into 20 units, as shown in Figure 16a, where each color corresponds to one unit-tree. Then, *n* unit-trees were randomly removed from the measurement, and the localization error was evaluated using the remaining measurement points (e.g., green points in Figure 16b).

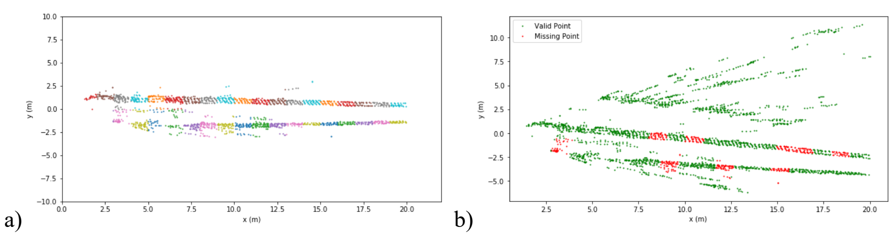
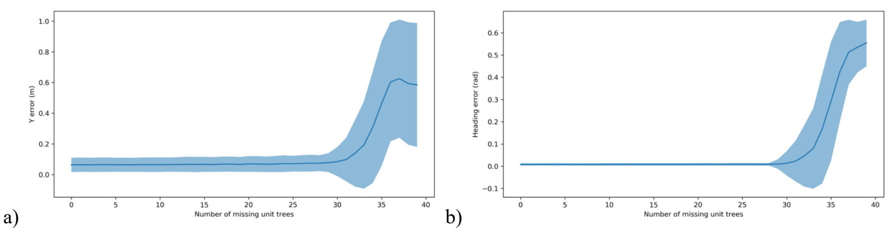


Figure 16 a) This figure shows unit-trees in a measurement; each color represents a unit-tree. b) A point-cloud example when four random unit-trees are removed from each side of the row; green points represent the remaining points.

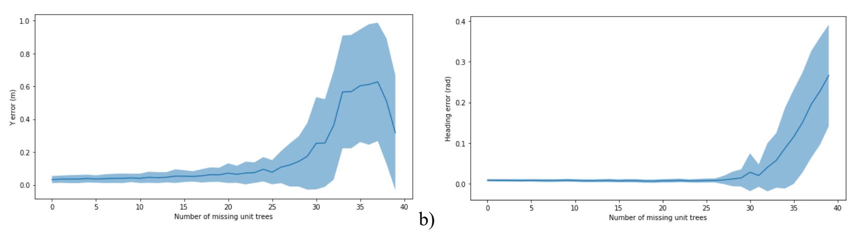
There are C(40, n) different combinations for removing n unit trees from 40 unit trees. Given the very large number of possible combinations, 100 were sampled randomly to evaluate the performance of the approach, at a given number of gap units n, as n was increased from 0 to 40. The lateral offset and heading error results for the vineyard data are shown in Figure 17, and for the apricot orchard data, in Figure 18.



a) b)

Figure 17 a) Offset error curve vs. the number of missing unit-trees in the vineyard. b) Heading error curve as a function of the number of missing unit-trees in the vineyard.

Commented [SGV1]: Explain shaded regions around curves.



a) b)

Figure 18 a) Offset error curve vs. the number of missing unit-trees in apricot orchard. b) Heading error curve as a function of the number of missing unit trees in the apricot orchard.

The localization accuracy remained almost constant until a certain threshold-number of unit trees were removed from the measurements. After inserting more gaps than this threshold, the errors grew very fast. In the particular vineyard, the threshold was approximately 30 unit trees (out of 40), i.e., the proposed localization approach performed robustly as long as no more than 75% of the measurement points were missing. For the apricot orchard, this threshold was approximately 25 (out of 40), i.e., the algorithm was robust as long as no more than 62.5% of the measurements were missing. The difference in algorithm robustness between the vineyard and the apricot orchard could be attributed to the fact that apricot trees were spaced farther apart from each other than grapevines. In real orchards and vineyards, the percentage of missing trees is very small. Hence, the proposed localization method is not expected to encounter such situations, and its robustness against gaps in tree rows seems adequate.

Along with the errors, we also extracted the standard deviation of the offset and heading from the covariance matrix returned by our localization algorithm in the vineyard case (Figures 18 a, b, respectively). The standard deviations represent the *confidence* of the localizer. Our localization algorithm started reporting high standard deviation in the lateral offset and the heading at the same time when the localization errors grow. The magnitude of the standard deviation is highly correlated with the actual error, as shown in Figure 18 c, d). These results showed that our algorithm could correctly report the uncertainties in lateral offset and yaw. The reported uncertainties could be used by other robot software modules, such as Bayesian filters or fail-safe modules, which can act accordingly when the uncertainly becomes large.

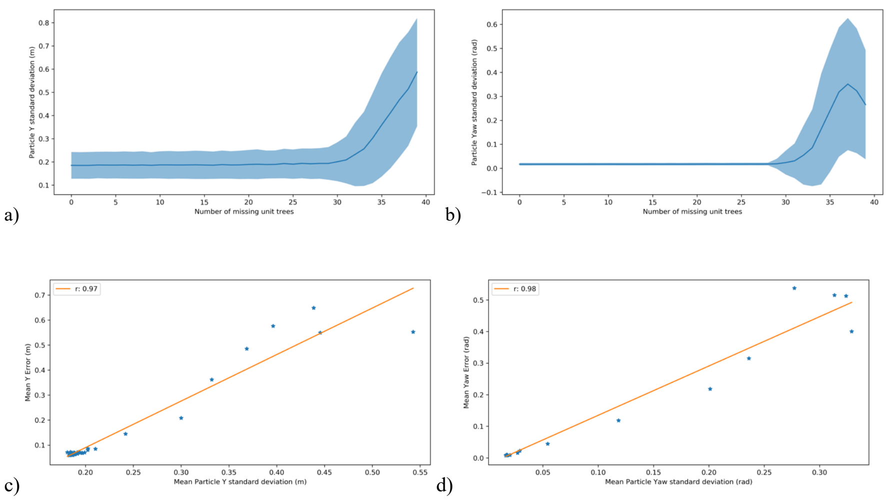


Figure 19 a) Lateral offset standard deviation curve vs. the number of missing unit trees in the vineyard. b) Heading standard deviation curve vs. the number of missing unit trees in the vineyard.

5.4 Localization accuracy near row boundaries

The template-based method is designed for in-row navigation, but it is important to understand the method's behavior when the vehicle approaches a row's end, in order to integrate this method in the future into an full orchard navigation system. A simulation experiment was designed to analyze the localization accuracy when the vehicle approaches the end of the row that it is traversing. The measurement points that were further than *d* meters away in the template frame's x-axis were removed. The remaining points represent the row end in *d* meters from the vehicle. An example of the simulated measurement is shown in Figure 20, and the localization accuracy as the vehicle "approaches the row end" is shown in Figure 21.

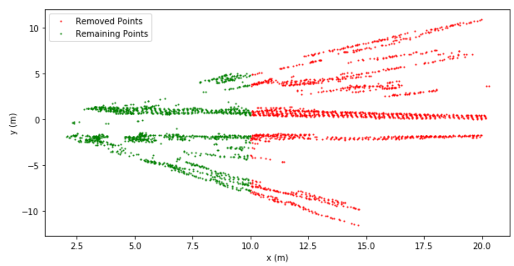


Figure 20 This figure shows points farther from the sensor than d meters are removed to simulate row exiting. The red points are removed points, and the green points are remaining points.

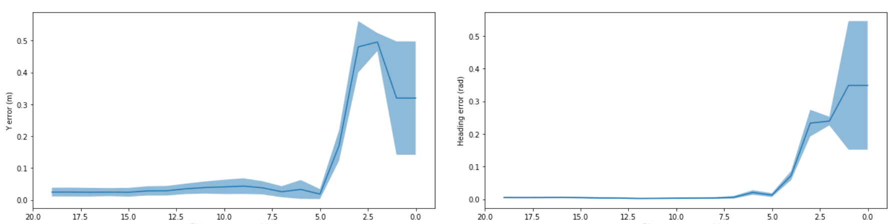


Figure 21 These figures show the localization accuracy as a function of the robot distance to the end of the row. The left figure shows the lateral offset error (Y), and the right figure shows the heading error.

The row-exiting results indicate that the localization accuracy did not change much until the vehicle was 5 m away from the row end. The remaining observed measurement points when the vehicle was 5 m away are shown in Figure 22 a). The overall row-exiting results indicated that the method could work well for an in-row vehicle until 5 m away from the row-end. The specific number may change under different situations; however, we can see that the safety margin is significant. Also, when the vehicle gets too close to the row end, the localization algorithm reports high localization uncertainty.

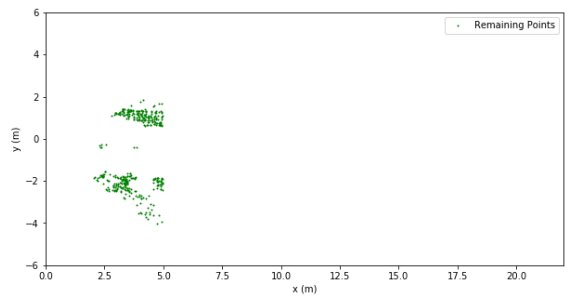


Figure 22 The figure shows remaining measurement points after 75% of points were removed to simulate the robot is 5 m away from the row end.

5.5 Template data requirements

All the experimental results were generated using templates built by 100 consecutive point-cloud measurements, which corresponds to approximately 7 seconds or 7 meters of data, as the sensor moved inside a row. Given that a template should capture the "expected" or "typical" structure of the orchard row (by storing spatial occupancy probabilities), one important question is, how many measurements are needed to build a good template. To explore this question, seven templates were built using 1, 5, 10, 20, 100, 200, and 300 point-clouds from consecutive frames of data in the vineyard. Then, the localization results - when each one of the seven templates was used - were evaluated on all vineyard data (Table 3).

Number of Measurements	**With visual odometry**								**Without visual odometry**							
	Y error (meter)					Yaw error (rad)			Y error (meter)					Yaw error (rad)		
	MAE†	SD†	95%†	MAE / Row spacing†	95% / Row spacing	MAE	SD	95%	MAE	SD	95%	MAE / Row spacing	95% / Row spacing	MAE	SD	95%
1	0.03	0.03	0.08	1.00%	2.80%	0.01	0.01	0.04	0.11	0.09	0.29	3.80%	9.50%	0.02	0.01	0.04
5	0.03	0.03	0.08	1.00%	2.80%	0.01	0.01	0.04	0.11	0.09	0.28	3.70%	9.30%	0.02	0.01	0.04
10	0.03	0.02	0.07	1.00%	2.40%	0.01	0.01	0.04	0.11	0.09	0.28	3.70%	9.30%	0.02	0.01	0.04
20	0.03	0.02	0.07	1.00%	2.20%	0.01	0.01	0.04	0.11	0.08	0.26	3.60%	8.70%	0.02	0.01	0.04
100	0.03	0.02	0.07	1.00%	2.50%	0.01	0.01	0.04	0.1	0.07	0.24	3.20%	8.00%	0.02	0.01	0.04
200	0.03	0.02	0.07	0.90%	2.30%	0.01	0.01	0.04	0.1	0.08	0.25	3.30%	8.20%	0.02	0.01	0.04
300	0.03	0.02	0.07	1.10%	2.50%	0.01	0.01	0.04	0.1	0.08	0.25	3.40%	8.30%	0.02	0.01	0.04

† MAE: Mean Absolute Error; SD: Standard Deviation; 95%: 95 Percentile;

† The average row spacing is 3 meters for this vineyard.

Table 3 Localization results in the vineyard, using templates built with different number of measurements

The results in Table 3 show that even a small number of measurements could build a good template. The mean absolute error did not change much, and the 95% percentile of the Y error decreased slightly as the number of measurements increased. After a certain number of measurements, the improvement saturates. This number was 100 in our vineyard experiments.

6 Conclusions and discussion

In this work, we proposed a generic 3D sensor-based method for robot localization with respect to orchard row centerlines. Instead of relying on assumptions about the presence of features and their spatial distributions with respect to row centerlines, our method discovers orchard-specific structure in a data-driven way, encodes it as an orchard row-sensing template, and utilizes the full 3D measurement information to determine the vehicle pose. We also proposed a way to estimate the confidence of the template-based localization estimate. Experiments were performed in a vineyard, and in an orchard in different seasons. Results showed that the method was quite accurate; lateral mean absolute error (MAE) was less than 3.6% of the row width, and heading MAE was less than 1.72°. Localization was also robust with respect to gaps in the tree rows, and the choice of row and number of measurements to build the template.

One limitation of this method – and actually of all in-row localization methods - is that the error grows significantly as the robot comes close to the end of the row. This happens because the set of sensed 3D points that belong to the current row becomes smaller, and the points beyond the end of the row can have arbitrary spatial distributions, which do not match the distribution of the points inside the row. In our approach, this limitation could be overcome to some extent by using concurrently a front-looking and a rear-looking sensor – with their corresponding template localization threads - and fusing their localization outputs based on the estimated confidence from each one.

The second limitation is that, in order to build the template, our method needs a set of initial measurements inside the row, with known ground truth sensor poses with respect to the centerline, in the absence of GNSS signals. In our experiments, GNSS was used in the vineyard for evaluation purposes, and a colored rope was used in the orchard, as an easily detectable physical centerline. Obviously, these methods are not practical for real-world deployment. One possibility, could be to setup a small number of permanent, easy-to-sense artificial landmarks in certain locations inside a few orchard rows, and use them to compute the sensor's true pose with respect to the centerline. This could be automated, thus making it easy to adapt/update row-templates when tree geometries/appearance changes because of events such as season changes, pruning or thinning.

Overall, the experimental results indicate that the proposed localization method is accurate and robust, and by re-building the template, the method can adapt to different orchards and dynamic changes in orchard appearance. Thus, the proposed method presents a generic approach to localization inside orchard rows, and in principle, inside any agricultural and non-agricultural structures that exhibit Characteristics #1 and #2; such environments include row-crops, greenhouse tunnels, corridors, aisles, etc.

References

Asaei, H., Jafari, A. and Loghavi, M., 2019. Site-specific orchard sprayer equipped with machine vision for chemical usage management. Computers and Electronics in Agriculture, (162), 431-439.

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. Biosystems Engineering, 96(2), 139-149.

Bell, J., MacDonald, B. A., & Ahn, H. S. (2016, October). Row following in pergola structured orchards. In 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 640-645).

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 Robotics & Automation Magazine, 22(1), 54-63.

Blok, P. M., van Boheemen, K., van Evert, F. K., IJsselmuiden, J., & Kim, G. H. (2019). Robot navigation in orchards with localization based on Particle filter and Kalman filter. Computers and electronics in agriculture, 157, 261-269.

Bolles, R. C., & Fischler, M. A. (1981, August). A RANSAC-Based Approach to Model Fitting and Its Application to Finding Cylinders in Range Data. In IJCAI (Vol. 1981, pp. 637-643).

Botterill, T., Paulin, S., Green, R., Williams, S., Lin, J., Saxton, V., Mills, S., Chen, X. and Corbett-Davies, S., 2017. A robot system for pruning grape vines. Journal of Field Robotics, 34(6), 1100-1122.

Charlton, D., Taylor, J. E., Vougioukas, S., & Rutledge, Z. (2019). Innovations for a Shrinking Agricultural Workforce. Choices, 34(2), 1-8.

Cvišic, I., Cesic, J., Markovic, I., & Petrovic, I. (2017). Soft-slam: Computationally efficient stereo visual slam for autonomous UAVs. Journal of field robotics.

Durand-Petiteville, A., Le Flecher, E., Cadenat, V., Sentenac, T., & Vougioukas, S. (2018). Tree Detection With Low-Cost Three-Dimensional Sensors for Autonomous Navigation in Orchards. IEEE Robotics and Automation Letters, 3(4), 3876-3883.

Elfes, A. (1989). Using occupancy grids for mobile robot perception and navigation. Computer, 22(6), 46-57.

Hamner, B., Bergerman, M., & Singh, S. (2011). Autonomous orchard vehicles for specialty crops production. In 2011 Louisville, Kentucky, August 7-10, 2011 (p. 1). American Society of Agricultural and Biological Engineers.

He, B., Liu, G., Ji, Y., Si, Y., & Gao, R. (2010, October). Auto recognition of navigation path for harvest robot based on machine vision. In International Conference on Computer and Computing Technologies in Agriculture (pp. 138-148). Springer, Berlin, Heidelberg.

Lyu, H. K., Park, C. H., Han, D. H., Kwak, S., & Choi, B. (2018). Orchard Free Space and Center Line Estimation Using Naive Bayesian Classifier for Unmanned Ground Self-Driving Vehicle. Symmetry, 10(9), 355.

Marden, S., & Whitty, M. (2014). GNSS-free localisation and navigation of an unmanned ground vehicle for yield forecasting in a vineyard. In Recent Advances in Agricultural Robotics, International workshop collocated with the 13th International Conference on Intelligent Autonomous Systems (IAS-13).

Radcliffe, J., Cox, J., & Bulanon, D. M. (2018). Machine vision for orchard navigation. Computers in Industry, 98, 165-171.

Rusu, R. B., & Cousins, S. (2011, May). 3d is here: Point cloud library (PCL). In 2011 IEEE international conference on robotics and automation (pp. 1-4). IEEE.

Shalev, O. and Degani, A., 2020. Canopy-Based Monte Carlo Localization in Orchards Using Top-View Imagery. IEEE Robotics and Automation Letters, 5(2), pp.2403-2410.

Sharifi, M., & Chen, X. (2015, February). A novel vision based row guidance approach for navigation of agricultural mobile robots in orchards. In 2015 6th International Conference on Automation, Robotics and Applications (ICARA) (pp. 251-255). IEEE.

Subramanian, V., Burks, T. F., & Arroyo, A. A. (2006). Development of machine vision and laser radar based autonomous vehicle guidance systems for citrus grove navigation. Computers and electronics in agriculture, 53(2), 130-143.

Thrun, S., Burgard, W., & Fox, D. (2005). Probabilistic robotics. MIT press.

Torres-Sospedra, J., & Nebot, P. (2011). A new approach to visual-based sensory system for navigation into orange groves. Sensors, 11(4), 4086-4103.

Vougioukas, S. G. (2019). Agricultural Robotics. Annual Review of Control, Robotics, and Autonomous Systems, 2, 365-392.

Williams, H., Ting, C., Nejati, M., Jones, M.H., Penhall, N., Lim, J., Seabright, M., Bell, J., Ahn, H.S., Scarfe, A. and Duke, M., 2020. Improvements to and large-scale evaluation of a robotic kiwifruit harvester. Journal of Field Robotics, 37(2), 187-201.

Zhang, J., Chambers, A., Maeta, S., Bergerman, M., & Singh, S. (2013, November). 3d perception for accurate row following: Methodology and results. In 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems(pp. 5306-5313). IEEE.

Zhang, Q. (editor), 2017. Automation in Tree Fruit Production: Principles and Practice. ISBN: 9781780648507. CABI.

Zude-Sasse, M., Fountas, S., Gemtos, T.A., and Abu-Khalaf, N., 2016. Applications of precision agriculture in horticultural crops.