IEEE Robotics and Automation Letters (RAL) paper presented at the
2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Canopy-Based Monte Carlo Localization in Orchards Using Top-View Imagery

Omer Shalev and Amir Degani, *Member, IEEE*

***Abstract*—Localization of ground mobile robots in orchards is a complex problem which is yet to be fully addressed. The typical localization approaches are not adjusted to the characteristics of the orchard environment, especially the homogeneous scenery. To alleviate these difficulties, we propose to use top-view images of the orchard acquired in real-time. The top-view observation of the orchard provides a unique signature of every tree formed by the shape of its canopy. This practically changes the homogeneity premise in orchards and paves the way for addressing the kidnapped robot problem. Using computer vision techniques, we build a virtual canopies laser scan around the ground robot which is generated from low-altitude top-view video streams. We apply Monte Carlo Localization on this virtual scan to localize the robot against a high-altitude top-view snapshot image which is used as a map. The suggested approach is examined in numerous offline experiments conducted on data acquired in real orchards and is compared against a typical simulated approach which relies on ground-level trunk observations. The canopy-based approach demonstrated better performance in all measures, including convergence to centimeter-level accuracy.**

I. Introduction

Mobile autonomous robots are becoming common in agriculture and are used for a variety of purposes. In orchards, there are various tasks that an autonomous robot can perform, such as sensing plant stress, sensing pests, yield monitoring, or selective spraying. One commonality to these tasks is the need for localization. While for some tasks it is enough to have a rough location estimate, for others it is crucial to have centimeter-level accuracy to allow precise sensing or precise manipulation capabilities, i.e., precision agriculture. As examples, accurate localization of an Unmanned Ground Vehicle (UGV) allows to execute accurate and selective pesticide spraying based on the tree's status and history.

Orchard environments present unique challenges for localization and mapping. First, the terrain roughness leads to significant vehicle skid. Additionally, the large dimensions of the orchard lead to large accumulated error in the estimated pose and therefore, wheel odometry [1] is not a reliable solution for UGV localization. Another common approach for localization in outdoor environments, which is extensively used in open field crops [2], is the use of Global Positioning System (GPS). In orchards, GPS has several drawbacks which impair its applicability. First, GPS depends on satellite coverage, which deteriorates at the ground level due to occlusions from the tree canopies [3], [4]. Second, the use of GPS alone does not allow to estimate the robot's orientation and requires an additional sensor such as an Inertial Measurements Unit (IMU) or a compass. Third, the nominal GPS meter-level accuracy is insufficient for precision agriculture tasks. Differential GPS (D-GPS) or Real-Time Kinematic (RTK) GPS offer nominal centimeter-level accuracy but require a constant base station in the field and are rather expensive. In addition, D-GPS and RTK-GPS suffer from similar occlusion problems in orchards as mentioned above, and hence their accuracy deteriorates in such environments. Finally, orchards are also problematic for visual odometry [5] since leaves and trunks lack unique visual signatures, and hence no natural landmarks allow loop closure of graph-based visual Simultaneous Localization and Mapping (SLAM) algorithms [6].

A backbone of many previous works related to GPS-denied localization in orchards is the Extended Kalman Filter (EKF) that is used to fuse information from different information channels such as wheel odometry, IMU readings, Light Detection and Ranging (LiDAR) scans and camera streams. A typical approach leverages the orchard's geometric characteristics and uses a compact map representation of tree lines. Two-dimensional LiDAR, mounted on a UGV, provides wide horizontal scans of the tree trunks level and of artificial landmarks (e.g., reflective tapes) that are used in some of the works, e.g., [7]. Computer vision techniques such as Hough transform [8] can perceive the tree lines which are used as measurements for the EKF that localizes the robot on the map built a priori [7], [9], [10]. This approach is satisfactory for simple tasks that only involve patrolling and require rough location estimate. One major drawback of this approach is the fact it uses the robot's starting point as a global reference. For the kidnapped robot problem, i.e., estimating the robot's pose when the initial pose is unknown (for example, following a hardware reset), this approach is unsuitable. The performance of a few typical localization approaches in orchards is demonstrated in the first part of the supplementary video.

Other works that focus on orchard mapping vastly rely on a three-dimensional point cloud acquired by a LiDAR or stereo camera. Several among these works tackle the mapping problem by detecting artificial landmarks that are used for loop closure [11] while others detect and segment trees using various techniques [12]–[14]. All these mapping methods

O. Shalev is with the Technion Autonomous Systems Program, Technion – Israel Institute of Technology, Haifa 3200003 ISRAEL (e-mail: omershalva@gmail.com).

A. Degani is with the Faculty of Civil and Environmental Engineering and with the Technion Autonomous Systems Program, Technion – Israel Institute of Technology, Haifa 3200003 ISRAEL (e-mail: adegani@technion.ac.il).

require expensive perception sensors and strong computation power. GPS, or even D-GPS, is also required in these works for localization during the mapping stage. Moreover, the tree segmentation algorithms use certain heuristics about tree shape and hence are typically environment dependent.

Monte Carlo Localization (MCL) [15] and its commonly used variant, Adaptive Monte Carlo Localization (AMCL) [16], are probabilistic localization approaches that use a particle filter to track the pose of a robot against a known map based on sensor readings. The big advantage of these approaches is their ability to operate without prior knowledge of the robot's initial state, that is, they address the kidnapped robot problem. Being based on a particle filter, MCL and AMCL are likely to converge in environments where the heterogeneity is large. As perceived by most sensing means mounted on a UGV, the heterogeneity of an orchard is low due to the similar trunk sizes and spacing and thus, orchards are traditionally conceived unsuitable for MCL and AMCL.

In this study we suggest a different AMCL-based approach for UGV localization in orchards. We manage to achieve global localization, without the knowledge of the initial state, against a map that we produce in a preliminary stage. The suggested approach is GPS independent and refrains from using artificial landmarks which might be expensive for setup and maintenance.

II. METHODOLOGY

The key concept of this study is the use of top-view images of the orchard as an auxiliary sensing aid to the ground vehicle. From the top-view images we extract the canopy shapes which form a heterogeneous representation of the orchard. With this representation, AMCL can provide global localization with centimeter-level accuracy. The suggested solution requires two top-view observations as input: (1) one large Field of View (FoV) snapshot image which captures the entire orchard plot, (2) a video stream in which the ground vehicle is continuously tracked. The large FoV image is used as a map and can be captured once by an Unmanned Aerial Vehicle (UAV) hovering at a high-altitude (60-80 m). The continuous video stream can be captured at a lower altitude (20-30 m) by either a UAV (potentially tethered to the ground and powered by higher capacity batteries) or a downward looking camera mounted to the UGV on top of a mast. Figure 1 illustrates the flow of the entire solution. Next, we describe the solution's components.

A. Canopies Extraction

In order to build the heterogeneous representation of the orchard, we apply computer vision techniques to extract the canopies from the top view images (Algorithm 1). We first convert the image to Hue Saturation Value (HSV) representation which allows us to easily extract the green canopy areas and create a binary green mask. From this mask, the contours are extracted using the Border Following algorithm suggested by Suzuki and Abe [17]. To remove outliers, we calculate the interior area of every contour and those below a threshold are removed. Eventually we create a binary mask by filling each of the contours. The details of this process is presented in our technical report [18].

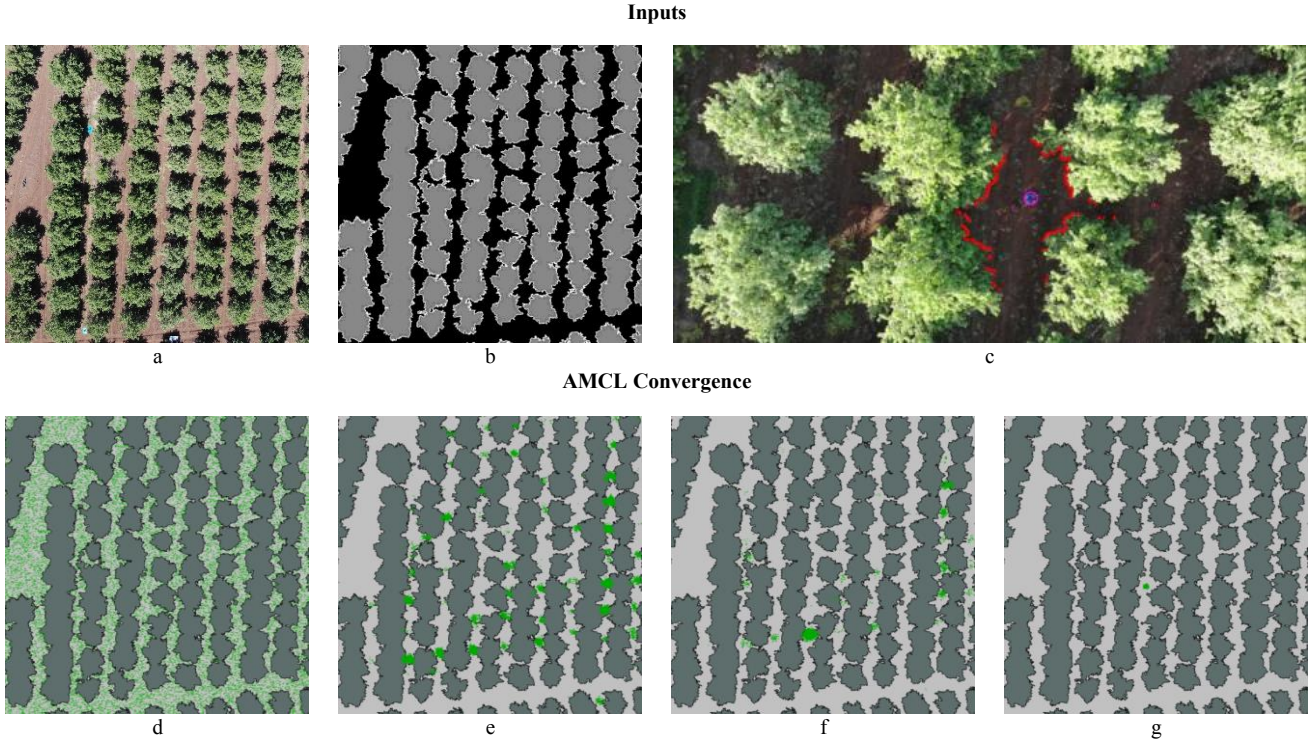

Figure 1: Canopy-based localization flow. (a) one high-altitude image; (b) the extracted canopy map; (c) one frame of the low-altitude video, with visualization of the virtual canopies scan (red) around the ground vehicle (purple); (d) initial uniform distribution of AMCL pose beliefs; (e-f) towards convergence of pose beliefs; (g) convergence of pose beliefs.

Algorithm 1: Extract-Canopies-Mask

Input: image [RGB matrix]
Output: canopies_mask [binary matrix]
Constants:
- UPPER_HSV
- LOWER_HSV
- AREA_THRESHOLD

```
image := Convert-To-HSV(image)
green_mask := LOWER_HSV ≤ image ≤ UPPER_HSV
contours_list := Border-Following(green_mask)
for contour in contours_list:
    if Area(contour) < AREA_THRESHOLD:
        remove contour
canopies_mask := Fill-Contours-Interior(contours_list)
```

Algorithm 2: Generate-Virtual-Canopies-Scan

Input:
- canopies_mask [binary matrix]
- origin [pixel coordinates]
- pixels_to_meters [float number]

Output: scan [array of float numbers]
Constants:
- NUMBER_OF_SAMPLES
- MIN_RADIUS
- MAX_RADIUS
- SEARCH_DELTA

```
scan[0:NUMBER_OF_SAMPLES] := {NULL}
for sample_index := 0 to NUMBER_OF_SAMPLES:
    angle := 360° / sample_index
    for radius := MIN_RADIUS to MAX_RADIUS
                  with SEARCH_DELTA step size:
        pixel := origin + (radius × Cos(angle),
                           radius × Sin(angle))
        if canopies_mask[pixel] = 1:
            scan[sample_index] := radius × pixels_to_meters
            break
```

B. *Virtual Canopies Scan*

Common localization solutions use two-dimensional LiDAR scans as their main sensorial input. These scans are used for understanding local and incremental translation and rotation changes (e.g., ICP [19]), but also for global localization by matching the scans against a prior map, and by that addressing the kidnapped robot problem (e.g., AMCL [16]). Motivated by the extensive use of LiDAR scans for localization and mapping of ground vehicles, we decided to generate a "virtual canopies scan" which depicts the shape of the canopy contours. As a LiDAR scan, the virtual canopies scan measures distances from an origin point where a robot is located to its surroundings. The surroundings in this case are the canopies. The virtual scan is calculated using computer vision techniques from the top-view image. Algorithm 2 describes the virtual scan generation procedure. The pixels-to-meters ratio is calculated as the quotient of measured distances at the ground level and their matching distances in pixels on the image. The canopies scans form a heterogenous representation of the orchard and, as already stated, this heterogeneity is vital for scan-based algorithms. These scans are also advantageous for being a compact representation that is also compatible with existing localization and mapping software libraries. The second part of the supplementary video ("Virtual Canopies Scan Demo") illustrates the tracking of a UGV and the extraction of virtual canopies scans in a real video acquired from a UAV at 20 m altitude.

C. *Canopy-Based AMCL*

MCL relies on two major inputs: map and scans. As stated above, a canopy map is generated from one high-altitude image of the orchard using the canopies extraction procedure. The top-view image and the resulting map are illustrated in Figure 1a-b. In addition, the continuous top-view video stream records the UGV in the orchard from low-altitude. For each frame of the stream, the vehicle is segmented from the image and its location is used as the origin point for generating the canopies scan. Figure 1c illustrates one virtual canopies scan on one frame of the low-altitude video. Assuming that the orchard ground is nearly planar, the vehicle's state is the two-dimensional vector (x, y), relative to the map origin. MCL operates on the canopies scans and estimates the vehicle's two-dimensional position against the canopy map.

We chose to use ROS's [20] implementation of AMCL [21] whose adaptive resampling typically results in lower computational complexity and faster convergence. This implementation estimates the pose, (x, y, θ), of the ground robot. To avoid unnecessary complexity and potential divergence following the estimation of θ, we modified the source code such that its initial particles are bounded in a narrow angular range. AMCL's motion update stage relies on an odometry model which accounts for noises and drifts. We chose the most generic omnidirectional model which can describe any robot as a point mass that can translate in all directions and consists of only five parameters. Figure 1d-g illustrates the convergence of AMCL particles on the canopy map over the time.

III. EXPERIMENTS

In order to evaluate our approach and demonstrate its robustness, we conducted numerous offline experiments using data collected in real almond (*Prunus dulcis*) orchards in Kibbutz Lavi in northern Israel. In order to strengthen the reliability of our findings, we collected multiple images at different hours of the day and at different seasons: April and November. In April, the data was collected in one orchard plot (denoted "plot A") throughout the day; in November, we collected data at the same original plot and in an additional plot (denoted "plot B"). In total, for the offline experiments we used eight different images (Figure 2).

For capturing the top-view images, we used the camera of a DJI Mavic Air [22]. The offline experiments were executed on a desktop PC with i7-6700 CPU and 16 GB RAM running Ubuntu 16.04 and ROS Kinetic. Computer vision code was implemented in Python using OpenCV [23] and NumPy [24]. Localization and mapping algorithms mainly relied on ROS libraries. Automation scripts were developed in Python and allowed dozens of hours of continuous experiments execution. Entire code used in this work can be found in our repository: https://github.com/CearLab/orchards_top_view_aided_navigation_experiments


a: April, 15:08, plot A


b: April, 15:53, plot A


c: April, 16:55, plot A


d: April, 19:04, plot A


e: November, 10:09, plot A


f: November, 10:10, plot A


g: November, 11:07, plot B

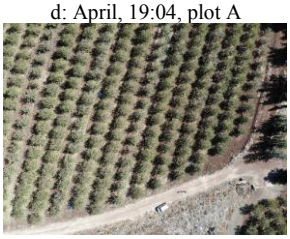
h: November, 11:07, plot B

Figure 2: Images at two different seasons, different hours of the day and from two different plots.


a: narrow passage


b: wide passage


c: S-shaped patrol


d: fork-shaped patrol


e: series of tasks

Figure 3: Virtual UGV trajectories (in yellow) on two among the orchard images. Upper row: plot A. Lower row: plot B.

A. Experimental Methodology

In our experiments, we use virtual UGV trajectories rather than tracking a real vehicle. The virtual trajectories are advantageous for providing the ground truth position of the virtual vehicle which will serve us for evaluation purposes. In addition, their use allows to generate and run numerous experiments offline. Each experiment uses one high-altitude image as a map and simulates the low-altitude video stream by cropping small windows of another high-altitude image along the virtual trajectory. The use of two different images acquired at different times of the day simulates the actual suggested mode of work. Figure 3 illustrates the different trajectories that we examined on two among the top-view images. Trajectories a and b are basic patterns along one passage, c and d are S-shaped and fork-shaped patrol patterns, and e combines a series of tasks at various points in the orchard. As an odometry source for the AMCL algorithm, we use the incremental translation between adjacent points of the trajectory.

In order to better evaluate our approach, we aim to not only measure our performance but also compare it against a reference. As mentioned above, two-dimensional LiDARs are typically mounted on the UGV such that they acquire a horizontal laser scan at the trunks level. To mimic the typical LiDAR mode of operation, we simulate trunks images and apply the same AMCL-based approach elaborated above for the canopies. The trunk positions are approximated at the center of mass of each canopy. Their dimensions are derived from real values of trunk circumference measured in the field. Since trunks are not perfect circles, we place ellipses centered at the approximated trunk positions with random deviation of ±5 cm in their axis lengths, derived from the measurement in the field, and with random orientation angle. Figure 4 illustrates the trunk map aligned with the canopy map as well as the canopies' and trunks' AMCL instances.

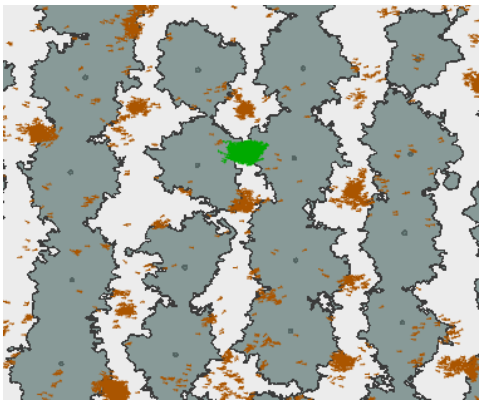

Figure 4: Aligned maps and AMCL particles: canopies (green) vs trunks (brown).

The AMCL trunks instance has mitigated conditions compared to real-world LiDAR scans. First, the angular range of the trunks scans is 360° whereas typical real-world LiDARs provide lower ranges. Second, we ignore the effect of terrain bumps and provide a continuous smooth scan which constantly perceives the surrounding trunks as if the scanner remains horizontal. Moreover, while the canopies scans suffer from discontinuities while the UGV passes beneath a canopy, the trunks scans are simulated continuously and are thus valid throughout the entire trajectory. The results show that the canopies instances demonstrate better performance despite the above-mentioned advantages which the trunks instances have over them.

B. *Synthesized Noise*

To further validate the suggested canopy-based AMCL approach, we conducted a series of experiments with various types of synthesized noise. We selected four among the experiments without noise (denoted "basic experiments") in which the performance of the AMCL trunks instance was competitive with the canopies instance and reran each of them multiple times with the various noises.

The first type of noise that we synthesized is the measurement noise. We simply added to the virtual scans an Independent and Identically Distributed (IID) gaussian noise with zero mean $N(0,\sigma^2)$. In reality, this type of noise can account for inaccuracies in the canopy extraction procedure but also for slight variations in canopy shape, e.g., due to windy weather conditions. We modified the σ parameter in the range $[0.1,0.5]$ meters.

The second and third type of synthesized noise are related to the odometry source which is used by AMCL as incremental positioning. As mentioned above, in the basic experiments without synthesized noise, the odometry was derived from the incremental positioning on the known virtual trajectory. In reality, we do not have this ground truth information and therefore we wanted to simulate realistic odometry noises on top of the above-mentioned incremental positioning. Two sources of noise account for odometry errors: drift and slippage [25]. We modeled the odometry noises as two independent gaussian random processes in both x and y axes, distributed $N(\mu_x,\sigma_x^2)$ and $N(\mu_y,\sigma_y^2)$, respectively, such that the standard deviations, σ_x,σ_y, reflect the slippage. The mean values, μ_x,μ_y, reflect the drift. For symmetry reasons, we decided to examine the addition of odometry noise only in the x-axis. We first examined the slippage effect in a set of experiments in which we modified σ_x in the range $[0.01,0.05]$. Then, in order to examine the effect of odometry drifts in addition to slippage, we set $\sigma_x = 0.01$ and modified μ_x in the range $[0.001,0.005]$. The odometry noise can also account for segmentation errors of the UGV from the top-view image.

C. *Evaluation Metrics*

For each of the two AMCL instances, we extract two output temporal signals. Most trivially, we are interested in the temporal pose error, calculated as the Euclidean distance between the UGV's pose (estimated by AMCL) and the ground truth pose, i.e., the pose on the virtual trajectory. The pose error is served as a measure of accuracy. An additional output of AMCL is a pose covariance matrix. We calculate the Frobenius norm of this matrix at every timestamp and that is used as a measure of certainty of AMCL. Since AMCL is stochastic by its nature, we repeat every experiment ten times. At every timestamp we calculate the standard deviation of the ten repetitions, which reflects the degree of nondeterminism.

In order to compare between trunks and canopies, we define five quantitative measures of an experiment which aggregate the results of all ten repetitions. As our interest is in centimeter-level tracking accuracy, we define a one-meter error band for the pose errors. A repetition of an experiment is said to converge if the pose error curve enters the one-meter error band (Figure 5a).

Convergence rate is defined as the portion of experiment repetitions which converged among the ten repetitions:

$$\text{CONVERGENCE RATE} = \frac{\#\text{ converged repetitions}}{10} \quad (1)$$

The second measure relates to the settling time of the pose error. We define settling time of a single repetition $t_s^{(i)}$ as the first time the pose error curve enters the error band (Figure 5a). Even though AMCL pose error might exceed the one-meter band following its first entrance, this measure reflects AMCL's speed of convergence. To aggregate all repetitions, we calculate the average settling time of the converged repetitions:

$$\text{SETTLING TIME} = \frac{\sum_{i\in\text{converged repetitions}} t_s^{(i)}}{\#\text{ converged repetitions}}\ [\text{sec}] \quad (2)$$

The third measure expresses the portion of time in which the pose error is within the error band, i.e., the portion of time with centimeter-level localization accuracy. For a single repetition, this quotient is calculated as $t_{EB}^{(i)} = \frac{1}{T}\sum_j \Delta t_{EB,j}$ (Figure 5a). This measure practically quantifies the relative amount of time with centimeter-level accuracy. Again, we aggregate all repetition results by averaging:

$$\text{IN-BAND RATE} = \sum_i t_{EB}^{(i)}/10 \quad (3)$$

The fourth measure of an experiment quantifies AMCL's uncertainty. For each repetition, we calculate the mean value

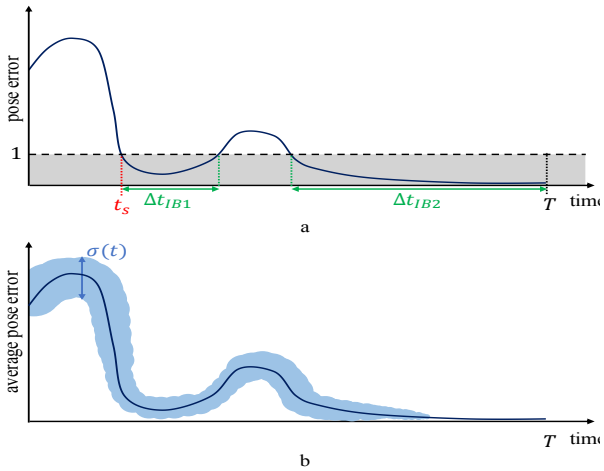

Figure 5: Figurative pose error graphs. (a) pose error of one experiment repetition; (b) average pose error of all repetitions of one experiment and their standard deviation.

of covariance norms over the time $E[\|Cov\|]^{(i)}$. Lower uncertainty values indicate better confidence of the algorithm in its estimation. Again, we average over all repetitions:

$$\text{UNCERTAINTY} = \frac{\sum_i E[\|Cov\|]^{(i)}}{10} \ [\text{m}^2] \qquad (4)$$

The fifth measure quantifies the variance between the ten repetitions of an experiment and reflects the degree of determinism. Low variance implies high degree of determinism and better robustness. At each timestamp, we calculate the variance of pose errors over the ten repetitions $\sigma(t)$ (Figure 5b). As a total variance we take the mean value of $\sigma(t)$ over the time:

$$\text{VARIANCE} = E[\sigma(t)] \ [\text{m}] \qquad (5)$$

IV. Results

The third part of the supplementary video ("Canopy-Based AMCL Demo") visualizes the two AMCL instances in one repetition of an experiment. In this experiment, Figure 2a is used as a map and Figure 2d is used to generate the simulated low-altitude video stream along an S-shaped patrol trajectory.

Table 6 depicts the aggregated results of the three series of experiments using both April and November data on the two plots. The colored bars visualize the cells' magnitude. For each combination of two images that are used as map and tracking sources, the five virtual trajectories illustrated in Figure 3 are generated using trajectory waypoints. As mentioned, each experiment consisted of ten repetitions. The measures in the table show clear advantage for the canopies instances, despite the above mentioned mitigated conditions of the trunks. In terms of convergence rate, in-band rate and settling time, the canopies instance almost always outperforms the trunks instance. The uncertainty of the canopies is lower in most cases compared with the trunks, implying stronger confidence of the algorithm in its estimated pose. Another interesting observation is the variance which is by magnitudes lower for the canopies instances in most cases, indicating stronger robustness for the canopies. The canopies instances fail in the "narrow passage" trajectories in certain cases. This is explained by the lower number of valid scans in these trajectories (around 70% only).

Table 7 concentrates the results of all experiments with synthesized measurement noise ($\sigma = 0$ is also listed as a reference). Despite certain deterioration in the canopies instances as σ increases, the measures show reasonable performance in most cases. In the trunks instances, on the other hand, the degree of performance deterioration is more significant, indicating lower robustness to this type of noise.

The results of experiments with slippage noise are presented in Table 8 (with $\sigma_x = 0$ as reference). The results of experiments with drift noise and constant slippage (with the reference experiments of $\sigma_x = 0.01$, $\mu_x = 0$) are presented in Table 9. In both sets of experiments, like in the case of measurement noise, the canopies instances demonstrated stability whereas the trunks instances demonstrated gradual degradation in all measures as noise parameter increases.

Table 6: Aggregated AMCL results (basic experiments)

series	map image	tracking image	trajectory	convergence rate		in-band rate		settling time		uncertainty		variance	
				canopies	*trunks*	*canopies*	*trunks*	*canopies*	*trunks*	*canopies*	*trunks*	*canopies*	*trunks*
April, plot A	Fig. 2a	Fig. 2b	narrow passage	1.0	0.9	0.40	0.33	35.78	37.70	194.96	227.56	4.16	9.74
			wide passage	1.0	0.9	0.56	0.40	26.37	32.89	126.47	255.80	0.57	13.38
			S-shaped patrol	1.0	0.9	0.89	0.76	17.71	28.24	33.27	76.41	0.32	8.33
			fork-shaped patrol	1.0	0.6	0.90	0.40	21.83	56.62	34.34	118.04	0.40	13.53
			series of tasks	1.0	0.4	0.75	0.30	50.15	54.47	76.71	107.85	1.43	14.20
	Fig. 2a	Fig. 2c	narrow passage	1.0	0.6	0.26	0.25	44.61	35.32	261.29	161.30	2.18	11.63
			wide passage	1.0	0.8	0.60	0.22	23.68	42.96	141.32	295.56	1.88	12.09
			S-shaped patrol	0.9	0.5	0.53	0.34	65.77	60.82	67.43	119.89	5.49	12.70
			fork-shaped patrol	1.0	0.7	0.86	0.44	30.24	62.52	50.36	95.08	0.21	11.60
			series of tasks	1.0	0.3	0.56	0.22	84.47	61.83	127.88	142.21	4.24	14.73
	Fig. 2a	Fig. 2d	narrow passage	1.0	0.7	0.60	0.21	24.02	41.78	146.25	234.86	1.70	9.64
			wide passage	1.0	0.7	0.65	0.15	20.92	47.03	133.61	252.06	1.05	11.31
			S-shaped patrol	1.0	0.8	0.89	0.57	16.67	54.63	34.96	110.64	0.17	10.32
			fork-shaped patrol	1.0	0.7	0.49	0.35	35.70	51.47	27.81	125.61	0.20	16.26
			series of tasks	1.0	0.4	0.85	0.14	29.92	141.58	40.68	107.17	0.72	13.17
	Fig. 2b	Fig. 2c	narrow passage	0.7	0.7	0.34	0.40	35.18	29.04	242.85	163.81	6.23	6.52
			wide passage	1.0	0.7	0.66	0.25	23.04	44.07	145.92	290.83	3.56	14.22
			S-shaped patrol	1.0	0.1	0.79	0.07	26.95	71.97	67.40	151.12	0.76	8.34
			fork-shaped patrol	1.0	1.0	0.90	0.74	24.32	55.79	65.46	118.30	0.79	7.78
			series of tasks	1.0	0.3	0.86	0.20	25.28	76.64	62.32	119.74	0.39	15.89
	Fig. 2b	Fig. 2d	narrow passage	0.1	0.9	0.06	0.54	29.91	26.18	264.86	151.15	6.38	7.45
			wide passage	1.0	0.6	0.77	0.18	16.10	48.81	90.65	219.03	0.76	11.13
			S-shaped patrol	1.0	1.0	0.91	0.70	15.16	61.95	45.22	93.16	0.29	2.55
			fork-shaped patrol	1.0	0.7	0.77	0.42	17.92	38.06	43.68	150.65	0.51	15.96
			series of tasks	1.0	0.6	0.85	0.31	16.34	129.38	50.09	125.47	0.78	8.56
	Fig. 2c	Fig. 2d	narrow passage	1.0	1.0	0.72	0.60	16.27	17.92	110.01	157.75	1.26	5.29
			wide passage	1.0	0.9	0.67	0.20	19.42	44.89	126.08	324.63	0.76	7.95
			S-shaped patrol	1.0	0.8	0.85	0.51	14.00	57.95	34.09	82.54	0.17	6.01
			fork-shaped patrol	1.0	0.5	0.61	0.21	19.33	80.93	32.67	143.40	0.41	12.23
			series of tasks	1.0	0.0	0.78	0.00	16.78		32.21	91.07	0.28	5.33
November, plot A	Fig. 2e	Fig. 2f	narrow passage	0.0	0.1	0.00	0.02		22.87	239.25	230.77	3.04	3.52
			wide passage	1.0	0.0	0.24	0.00	40.42		169.78	315.10	1.54	11.79
			S-shaped patrol	1.0	0.9	0.41	0.60	38.15	52.79	53.17	102.26	1.55	10.46
			fork-shaped patrol	1.0	1.0	0.79	0.63	20.51	24.54	60.54	96.75	1.33	3.26
			series of tasks	1.0	0.1	0.61	0.07	45.34	56.70	98.15	113.96	0.81	9.48
November, plot B	Fig. 2g	Fig. 2h	narrow passage	0.0	0.3	0.00	0.01		8.66	209.22	559.86	3.44	19.43
			wide passage	1.0	0.0	0.10	0.00	60.02		111.48	420.39	4.19	9.59
			S-shaped patrol	1.0	0.6	0.11	0.17	52.07	51.95	83.60	203.74	1.25	12.27
			fork-shaped patrol	1.0	0.9	0.86	0.03	28.90	34.18	33.81	131.55	0.81	12.11
			series of tasks	0.7	0.8	0.01	0.16	91.26	60.18	110.78	128.13	5.35	9.35

Table 7: Aggregated AMCL results (measurement noise)

map image	tracking image	trajectory	sigma	convergence rate		in-band rate		settling time		uncertainty		variance	
				canopies	*trunks*	*canopies*	*trunks*	*canopies*	*trunks*	*canopies*	*trunks*	*canopies*	*trunks*
Fig. 2a	Fig. 2b	wide passage	0	1.0	0.9	0.56	0.40	26.37	32.89	126.47	255.80	0.57	13.38
			0.1	1.0	0.9	0.52	0.21	28.40	45.15	133.73	282.88	0.98	11.50
			0.2	1.0	1.0	0.47	0.09	31.50	53.45	138.97	323.07	0.72	10.57
			0.3	1.0	0.3	0.28	0.02	42.34	55.43	173.56	356.03	2.54	11.41
			0.4	1.0	0.0	0.10	0.00	53.21		262.17	505.10	5.69	16.57
			0.5	0.2	0.0	0.00	0.00	57.98		324.72	548.19	6.61	10.97
Fig. 2b	Fig. 2c	wide passage	0	1.0	0.7	0.66	0.25	23.04	44.07	145.92	290.83	3.56	14.22
			0.1	1.0	1.0	0.52	0.23	32.87	52.17	202.91	324.08	2.65	7.55
			0.2	1.0	0.8	0.60	0.17	27.21	48.54	185.67	356.21	2.45	13.16
			0.3	1.0	0.1	0.52	0.00	32.75	67.40	202.65	388.53	1.53	14.91
			0.4	1.0	0.0	0.41	0.00	40.46		204.00	469.36	1.37	11.90
			0.5	0.2	0.0	0.01	0.00	63.61		305.57	627.40	4.53	13.27
Fig. 2c	Fig. 2d	narrow passage	0	1.0	1.0	0.72	0.60	16.27	17.92	110.01	157.75	1.26	5.29
			0.1	1.0	0.8	0.68	0.36	18.55	32.30	129.10	190.70	1.93	8.50
			0.2	1.0	0.9	0.66	0.39	20.10	33.08	137.49	212.73	1.92	7.51
			0.3	1.0	0.6	0.57	0.17	25.26	41.97	169.41	252.58	2.15	5.77
			0.4	0.4	0.0	0.01	0.00	56.58		302.58	518.33	5.14	8.73
			0.5	1.0	0.1	0.06	0.00	54.65	7.18	298.23	378.89	4.21	7.97
Fig. 2c	Fig. 2d	wide passage	0	1.0	0.9	0.67	0.20	19.42	44.89	126.08	324.63	0.76	7.95
			0.1	1.0	0.5	0.65	0.07	20.23	50.19	132.65	402.45	1.13	9.89
			0.2	1.0	0.3	0.59	0.02	24.08	54.47	149.88	346.26	0.81	10.04
			0.3	1.0	0.0	0.47	0.00	31.11		172.63	457.90	1.96	14.06
			0.4	0.3	0.0	0.03	0.00	51.18		302.06	608.44	5.73	12.75
			0.5	1.0	0.0	0.13	0.00	50.62		285.19	467.69	6.61	11.67

Table 8: Aggregated AMCL results (slippage noise)

map image	tracking image	trajectory	sigma x	convergence rate		in-band rate		settling time		uncertainty		variance	
				canopies	*trunks*	*canopies*	*trunks*	*canopies*	*trunks*	*canopies*	*trunks*	*canopies*	*trunks*
Fig. 2a	Fig. 2b	wide passage	0	1.0	0.9	0.56	0.40	26.37	32.89	126.47	255.80	0.57	13.38
			0.01	1.0	0.8	0.53	0.29	28.00	38.00	130.11	273.64	0.89	14.56
			0.02	1.0	0.9	0.54	0.27	27.35	41.22	127.40	303.14	1.12	13.68
			0.03	1.0	0.9	0.50	0.12	29.45	50.83	138.75	336.02	2.20	15.75
			0.04	0.9	0.4	0.41	0.04	32.13	52.55	163.88	277.75	6.04	15.02
			0.05	0.8	0.1	0.32	0.01	35.82	51.36	193.89	380.97	10.13	16.41
Fig. 2b	Fig. 2c	wide passage	0	1.0	0.7	0.66	0.25	23.04	44.07	145.92	290.83	3.56	14.22
			0.01	1.0	1.0	0.54	0.28	31.32	48.71	195.98	323.62	3.23	8.87
			0.02	1.0	1.0	0.65	0.33	23.89	45.56	157.79	309.15	3.57	14.72
			0.03	1.0	0.7	0.64	0.18	24.99	44.76	161.88	318.91	3.91	13.78
			0.04	1.0	0.2	0.61	0.04	26.49	55.76	173.11	359.25	3.76	15.08
			0.05	0.9	0.2	0.18	0.00	54.27	16.76	261.12	579.08	7.26	11.05
Fig. 2c	Fig. 2d	narrow passage	0	1.0	1.0	0.72	0.60	16.27	17.92	110.01	157.75	1.26	5.29
			0.01	1.0	1.0	0.70	0.51	17.70	26.58	123.39	161.95	2.06	4.13
			0.02	1.0	0.9	0.71	0.49	17.01	26.54	114.06	170.61	1.43	7.04
			0.03	1.0	0.9	0.68	0.35	18.56	35.30	126.29	173.27	2.79	7.87
			0.04	1.0	0.3	0.38	0.09	35.92	41.03	205.05	299.53	5.16	5.31
			0.05	1.0	0.3	0.56	0.07	25.26	43.41	172.61	225.24	3.64	5.22
Fig. 2c	Fig. 2d	wide passage	0	1.0	0.9	0.67	0.20	19.42	44.89	126.08	324.63	0.76	7.95
			0.01	1.0	1.0	0.67	0.24	19.36	43.99	126.72	295.08	1.07	6.52
			0.02	1.0	1.0	0.65	0.23	20.20	44.76	127.28	325.53	1.30	7.93
			0.03	1.0	0.3	0.62	0.05	22.31	47.85	144.27	395.16	1.84	12.84
			0.04	0.8	0.0	0.27	0.00	38.40		217.72	477.81	4.28	13.88
			0.05	1.0	0.0	0.54	0.00	26.66		156.75	385.84	3.13	14.45

Table 9: Aggregated AMCL results (drift noise with constant slippage)

map image	tracking image	trajectory	mu x	convergence rate		in-band rate		settling time		uncertainty		variance	
				canopies	*trunks*	*canopies*	*trunks*	*canopies*	*trunks*	*canopies*	*trunks*	*canopies*	*trunks*
Fig. 2a	Fig. 2b	wide passage	0	1.0	0.8	0.53	0.29	28.00	38.00	130.11	273.64	0.89	14.56
			0.001	1.0	0.8	0.52	0.25	28.22	40.76	127.61	260.95	1.39	11.46
			0.002	1.0	1.0	0.50	0.24	29.82	44.73	136.98	321.40	1.29	12.42
			0.003	1.0	0.2	0.49	0.02	30.25	51.93	137.38	315.96	1.62	7.62
			0.004	1.0	0.1	0.46	0.00	32.19	8.33	144.10	345.75	1.78	16.80
			0.005	0.9	0.0	0.37	0.00	35.20		181.78	531.28	6.78	14.90
Fig. 2b	Fig. 2c	wide passage	0	1.0	1.0	0.54	0.28	31.32	48.71	195.98	323.62	3.23	8.87
			0.001	1.0	0.7	0.65	0.17	23.83	51.58	157.33	360.01	2.94	13.29
			0.002	0.8	0.3	0.32	0.06	40.62	53.52	235.50	455.70	8.09	14.92
			0.003	1.0	0.7	0.66	0.14	23.58	54.42	149.26	342.46	3.54	13.42
			0.004	1.0	0.0	0.64	0.00	24.57		165.21	441.38	3.19	15.84
			0.005	1.0	0.0	0.64	0.00	24.88		163.19	491.48	4.30	13.90
Fig. 2c	Fig. 2d	narrow passage	0	1.0	1.0	0.70	0.51	17.70	26.58	123.39	161.95	2.06	4.13
			0.001	1.0	1.0	0.72	0.44	16.21	32.46	110.06	170.80	1.23	3.04
			0.002	1.0	1.0	0.72	0.51	16.22	28.66	113.79	205.36	1.89	5.86
			0.003	1.0	1.0	0.71	0.44	16.68	32.58	115.24	198.35	1.73	5.60
			0.004	1.0	0.6	0.72	0.16	16.67	39.75	116.25	221.57	1.08	5.45
			0.005	1.0	0.1	0.71	0.00	16.93	6.09	117.39	305.04	1.22	9.42
Fig. 2c	Fig. 2d	wide passage	0	1.0	1.0	0.67	0.24	19.36	43.99	126.72	295.08	1.07	6.52
			0.001	1.0	0.5	0.66	0.10	20.02	46.42	129.54	347.25	1.09	11.95
			0.002	1.0	0.8	0.65	0.16	20.53	46.54	135.22	328.60	0.86	12.81
			0.003	1.0	0.5	0.62	0.07	21.88	49.53	140.73	349.03	1.34	10.79
			0.004	1.0	0.0	0.60	0.00	23.50		146.05	440.98	1.44	10.36
			0.005	1.0	0.0	0.48	0.00	30.24		179.16	454.74	3.08	11.52

V. Conclusions and Future Work

Our canopy-based localization approach presented in this work is novel in the sense of using the MCL algorithm in orchards, which are large-scale environments that are characterized by highly homogeneous scenery. MCL requires a prior map of the robot's environment. While map acquisition is typically a complex and expensive task, we easily generate a map from one high-altitude image of the orchard by applying simple computer vision operations. We use a low-altitude video stream to generate virtual canopies scans in order to localize the robot on the map.

The results listed above demonstrate better performance of the canopies instances compared with the reference trunks instances in almost every aspect. Furthermore, we were able to demonstrate centimeter-level localization accuracy for the canopies instances in the majority of the experiments. This capability is vital for applications of precision agriculture in orchards. The low uncertainty and variance demonstrated in the canopies instances are also indicative of the solution's quality. The robustness of the solution was further strengthened in the experiments involving synthesized noise. In addition, the results demonstrate robustness to lighting conditions, e.g., in the third set of experiments in Table 6.

The main contribution of this work over other approaches is the addressing of the kidnapped robot problem and demonstrating centimeter-level accuracy. Another advantage of our approach stems from its simplicity. If compared against full graph-based SLAM algorithms, our approach is much more efficient in terms of computational complexity. If compared against EKF-based localization, our approach requires a much simpler model parametrization (only five parameters).

The major weakness of the canopy-based approach is visual occlusion when the UGV passes beneath the canopies and therefore scans are not available. This can be alleviated using a hybrid approach in which the UGV's state is updated based on its odometry while not visible to the UAV.

The successful results of canopy-based AMCL inspires the use of AMCL for UAV localization in future work. The formulation of this problem is slightly different as it requires estimating not only the UAV's horizontal position (x, y) but also its heading θ and its altitude z. However, we trust that the canopy signatures can help address this problem as well. Another follow up work is integrating the canopies AMCL output with the UGV's standard EKF in order to check if the additional stream of information improves the EKF's performance. In addition, we intend to continue implementing the approach suggested in this paper on different types of orchards and groves. The top-view based approach suggested in this work also has potential in other rapidly changing environments (e.g., fast growing vegetation).

References

[1] R. Siegwart and I. R. Nourbakhsh, "Mobile Robot Kinematics," in *Introduction to Autonomous Mobile Robots*, MIT press, 2011, pp. 47–82.

[2] "GPS.gov: Agricultural Applications." [Online]. Available: https://www.gps.gov/applications/agriculture/. [Accessed: 20-Jan-2018].

[3] J. Libby and G. Kantor, "Accurate GPS-free positioning of utility vehicles for specialty agriculture," in *ASABE Annual International Meeting, Pittsburgh, PA*, 2010, vol. 0300, no. 10, p. 1.

[4] J. P. Underwood, G. Jagbrant, J. Nieto, and S. Sukkarieh, "Tree centric localisation in almond orchards," in *XXIX International Horticultural Congress on Horticulture: Sustaining Lives, Livelihoods and Landscapes*, 2014, no. 1130, pp. 619–624.

[5] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004, vol. 1, pp. 652–659.

[6] M. Labbé and F. Michaud, "Long-term online multi-session graph-based SPLAM with memory management," *Auton. Robots*, vol. 42, no. 6, pp. 1133–1150, Nov. 2018.

[7] J. Libby and G. Kantor, "Deployment of a point and line feature localization system for an outdoor agriculture vehicle," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 1565–1570.

[8] P. V. C. Hough, "Machine analysis of bubble chamber pictures," *2nd Int. Conf. High-Energy Accel. Instrum.*, vol. 73, pp. 554–558, 1959.

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

[10] M. Bergerman *et al.*, "Robot farmers: Autonomous orchard vehicles help tree fruit production," *IEEE Robot. Autom. Mag.*, vol. 22, no. 1, pp. 54–63, Mar. 2015.

[11] J. Zhang, S. Maeta, M. Bergerman, and S. Singh, "Mapping Orchards for Autonomous Navigation," in *ASABE Annual International Meeting*, 2014.

[12] M. Nielsen, D. C. Slaughter, C. Gliever, and S. Upadhyaya, "Orchard and Tree Mapping and Description Using Stereo Vision and Lidar," in *International Conference of Agricultural Engineering*, 2012.

[13] G. Jagbrant, J. P. Underwood, J. Nieto, and S. Sukkarieh, "LiDAR Based Localisation in Almond Orchards," in *The 9th Conference on Field and Service Robotics*, 2013.

[14] N. Shalal, T. Low, C. McCarthy, and N. Hancock, "Orchard mapping and mobile robot localisation using on-board camera and laser scanner data fusion - Part B: Mapping and localisation," *Comput. Electron. Agric.*, vol. 119, pp. 267–278, Nov. 2015.

[15] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte Carlo localization for mobile robots," in *1999 IEEE International Conference on Robotics and Automation*, 1999, vol. 2, pp. 1322–1328.

[16] D. Fox, "KLD-Sampling: Adaptive Particle Filters," in *Advances in neural information processing systems*, 2002, pp. 713–720.

[17] S. Suzuki and K. Abe, "Topological Structural Analysis of Digitalized Binary Images by Border Following," *Comput. Vision, Graph. Image Process.*, vol. 30, no. 1, pp. 32–46, 1985.

[18] O. Shalev, "Robot Navigation in Orchards Using Top-View Imagery." [Online]. Available: https://cear.net.technion.ac.il/files/2019/12/Thesis-Omer-Shalev-Submission-Final.pdf. [Accessed: 07-Dec-2019].

[19] A. Censi, "An ICP variant using a point-to-line metric," in *2008 IEEE International Conference on Robotics and Automation*, 2008, pp. 19–25.

[20] "ROS.org | Powering the world's robots." [Online]. Available: http://www.ros.org/. [Accessed: 15-Feb-2019].

[21] "amcl - ROS Wiki." [Online]. Available: http://wiki.ros.org/amcl. [Accessed: 05-Feb-2018].

[22] "DJI Mavic Air – Foldable 4K Drone – DJI." [Online]. Available: https://www.dji.com/mavic-air. [Accessed: 06-Apr-2019].

[23] "OpenCV." [Online]. Available: https://opencv.org/. [Accessed: 07-Jun-2019].

[24] "NumPy — NumPy." [Online]. Available: https://www.numpy.org/. [Accessed: 07-Jun-2019].

[25] S. Thrun, W. Burgard, and D. Fox, "Odometry Motion Model," in *Probabilistic Robotics*, MIT press, 2005, pp. 107–113.