Orchard Row-Detection in Drone Images

Timothy Simons
Department of Computer Science
University of Cape Town
May 2020

Abstract

This paper presents a review of the existing literature relevant to the classification of trees or shrubs in an orchard into distinct rows. The problem of orchard row-detection can be reduced to finding a configuration of nodes or points that accurately depict a discrete number of rows or lines. For this reason, there is a focus on the class of algorithms that solve combinatorial optimisation problems. These algorithms, and the applications of which, are evaluated for their applicability to the orchard row finding problem. Of the compared algorithms, PSO, ACO and their multi-objective variants prove to be the most eligible candidates.

1. Introduction

Agricultural management has become an increasingly difficult and cost-intensive task given the increasing size of modern-day farms. Presently, most of the orchard inspection is conducted on foot with workers walking between rows to examine each tree or shrub in an orchard. The manual inspection and constant supervision of trees is proving infeasible for larger farms and at least cost-ineffective for smaller ones. Recent advances in unmanned aerial vehicles (UAVs) and an increased accessibility to near real-time satellite images has enabled the automation of such processes.

Prior to orchard analysis or management, relevant data regarding the orchard must be acquired and processed. The part of the data acquisition process relevant to our project is that of row-detection. Once detected, this information could be used for different analyses, ranging from yield-estimation to the identification of missing trees. Our project will exclusively focus on the detection of rows, omitting the detection of the trees themselves. Object detection is a well-researched field and as such we are using an existing object detector to detect the trees or shrubs. The resulting detections can be seen below.

Figure 1 and Figure 2 represent a simple case and a difficult case respectively. This shows how the fidelity of detections can vary drastically from input to input. Our solution needs to accommodate for the worst case. One of the difficulties not illustrated in these two figures, is that of curved rows which also needs to be handled by our solution.

The detection of trees or shrubs produced by the object detector allows for a reduction of the problem description. Instead of finding rows from pixel data, we are now finding rows from a set of points, with each point being the center of a tree. This opens a new avenue of possible solutions, particularly those related to graph theory. When viewed in this way, we see that this is a combinatorial optimisation problem. That is to say there exists a set of discrete feasible solutions who's optimal cannot be found through exhaustive search. Furthermore, this problem can be posed as a multi-objective (MO) optimisation problem. Objectives can be chosen to encode any assumptions we make about how farmers plant their trees. For instance, if we assume spacing between trees to be smaller than spacing between rows, we can choose an objective to be the maximisation of the difference between inter-row and intra-row distances. Another possible objective might be the minimisation of the rate at which the angle of a given row changes. This is if we assume rows follow a constant curvature.

There exists *a priori* and *a posteriori* methods that solve MO problems. *A priori* methods require the knowledge of preference information before the solution process. When this preference is known, we can simply order objectives by level of importance or combine the objectives into a single objective using the weighted-sum approach. In the orchard-row detection problem, it can be argued that we do not know *a priori* preference.

Figure 1: An example of an orchard aerial image (left) accompanied by the corresponding tree detections (middle) and tree centers (right). This example depicts distinct rows which are neatly arranged into straight lines.

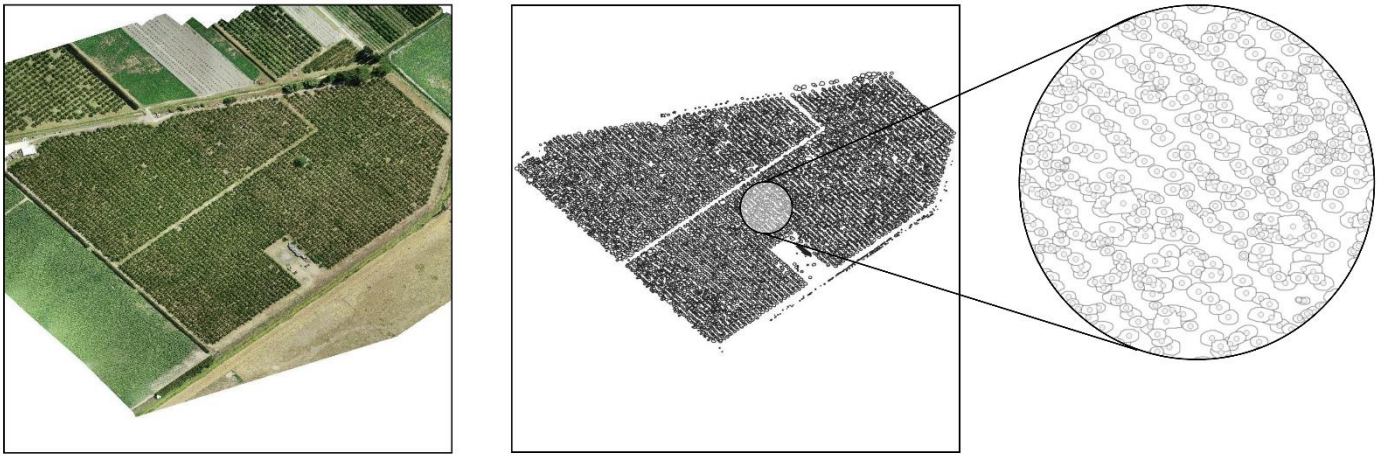

Figure 2: An example of a difficult case that contains erroneous and spurious detections with rows that aren't distinct or neatly arranged into straight lines.

We cannot know the combination of preferences that defines the optimal solution for a specific orchard prior to solving the problem. Different orchards may prefer one objective over another. For an example, one orchard may portray rows with a constant curvature and missing trees whilst another may have distinct rows with erratic curvature. In this case the first orchard will prefer the curvature objective whereas the second prefers the distance objective. Other orchards may prefer some combination of the two. We therefore require a set of possible solutions to choose from. That is, we require what is known as a Pareto optimal set where each combination of preferences isn't dominated by another. There is a large body of research into evolutionary algorithms that solve MO problems to produce this Pareto set. These algorithms will comprise most of the methods covered in this review along with a few others applicable to the orchard row finding problem.

2. Hough Transform

The Hough transform is a simple mathematical technique for finding groups of collinear points. This method was later extended and popularised by Duda and Hart (1972) who used the technique for edge-detection in image analysis [10]. The relevance of this technique comes from the fact that tree detections can be represented by a set of points. If we can find the points that are collinear with one another we have effectively found the trees that belong to the same row. The use of the Hough transform to detect crop rows has been explored in several papers in the context of both ground [1, 14, 20] and aerial images [2, 18, 23]. The application of the Hough transform is limited in that it can only detect straight crop rows. This drawback of the Hough transform was addressed by Ballard (1987) who described a method of using the Hough transform to detect arbitrary non-analytical shapes [3].

The method devised by Ballard uses the predefined boundary of a shape to detect instances of that shape in an image or picture. It is illustrated that these shape-detections are robust to rotations, scale changes and figure-ground reversals [3]. This method opens the possibility of detecting rows that depict some degree of curvature. Despite this generalisation, there is still a hindrance to the application of this method to the problem of orchard row-detection. That is, it is impossible to deduce the shape or curvature of a row prior to identifying the rows, and thus the requirement of a predefined boundary cannot be fulfilled. A possible workaround is to define a set of component curves that can be used to construct or identify more complex curves. This idea has been explored with specific application to the identification of plantation rows in aerial images of coffee crop fields [23].

The problem of identifying curved rows is a complex problem that is difficult to solve. Soares et al. (2018) takes advantage of the fact that a curved line can be approximated through a series of straight lines [23]. Each aerial image of a coffee crop field is segmented into a series of tiles such that the rows in each tile are approximately straight. This allows the use of the Hough transform in each of the tiles to detect the rows in that tile. The detected lines are then combined to form the series of curved lines that represent the rows of the orchard. Experimental results of this method aren't too promising. Detections of crop rows contain a high degree of both false positives and false negatives. This approach also relies on the assumption that rows aren't too heavily curved. Rows that depict a high degree of curvature can't use image tiling as there would be too few trees in each tile to apply the Hough transform.

3. Evolutionary Algorithms (EAs)

An evolutionary algorithm is a generic population-based metaheuristic optimisation algorithm. Generally, this class of algorithm employs some mechanism inspired by biological evolution. Candidate solutions act as the individuals of the population which are iteratively improved upon over subsequent generations. The quality of these solutions/individuals is determined by a fitness function.

3.1 Particle Swarm Optimisation (PSO)

Particle swarm optimisation optimises combinatorial problems through iteratively improving candidate solutions. Each candidate solution, called a particle, searches the solution space according to its position and velocity. Its movements are guided by the movement of other particles in the swarm where each particle aims to maintain an optimal distance between itself and its neighbours. Furthermore, the position and velocity of each particle seeks to match the best-known position and velocity of the swarm. In this way, particles remain diverse as they search the solution space whilst simultaneously converging on the best-known solution.

This simple idea has a promising correspondence when thought of in terms of orchard row-detection. Each particle's traversal can represent a row. These traversals can be guided by distance and angle metrics. This will help each particle remain on their respective row. Each particle will try remain equidistant from other particles close to it. This will also help ensure that each particle doesn't 'hop' onto another particle's row. The only modification of the PSO algorithm required is the removal of the particle's movement towards the best-known position. This behaviour assumes a single optimum position in the search space which isn't the case in the orchard row-detection problem. A promising characteristic of PSO is that each particle has its own momentum. This allows particles to ignore deviations/noise in each row as each particle would more readily travel along the points characterised by constant curvature. PSO is also considered an efficient method of combinatorial optimisation as the algorithm is composed of primitive mathematical operators with inexpensive computation and memory requirements [15].

3.2 Ant Colony Optimisation (ACO)

Ant Colony Optimisation is a probabilistic technique for solving hard combinatorial optimisation problems which can be reduced to finding optimal paths through graphs. ACO is loosely based on the behaviour of ants when they search for food. Ants secrete pheromones that serve as a signalling mechanism that allow them to collectively navigate their environment. Accumulated pheromones represent the ants' preference, typically based on quality of food, distance from the nest and amount of food available. The path with the highest level of trail pheromone is considered the optimal path. ACO builds on this idea by incorporating heuristics specific to the problem being solved. In this way, ACO is a metaheuristic that makes probabilistic decisions as a function of trail pheromones and domain-specific heuristics [9]. There exists some algorithmic variants and many applications thereof that bear relevance to the problem of orchard row classification. The first of which being edge detection [13, 17, 24].

Tian et al. (2008) applies an ACO algorithm to the problem of image edge detection [24]. Images typically contain pixel discontinuities or local changes in image brightness. These discontinuities can be organised into a set of curved line segments called edges. This approach aims to construct a pheromone matrix that represents the edge information of an image. This is similar to orchard row-detection in that we want to construct a pheromone matrix that -represents the rows of a given orchard. Once they've constructed the pheromone matrix for a given image, they use a thresholding method to determine which pixels represent an edge. If the deposited pheromone on a pixel is above a certain threshold, then it is an edge otherwise it is not. A similar approach could be applied to row-detection where tree centers with a pheromone value above a certain threshold are part of a row. An alternative to the thresholding method would be a local search that tries to extract the optimal paths from the pheromone matrix.

4. Evolutionary Multi-objective Optimisation (EMO)

As illustrated in the introduction, the row-detection problem requires a set of non-dominated solutions (Pareto front) from which to choose from. Evolutionary algorithms are a popular approach when dealing with *a posteriori* multi-objective optimisation. In this section, we'll focus on the applicability of a few well-known MO evolutionary algorithms.

4.1 Non-dominated Sorting Genetic Algorithm-II (NSGA-II)

Non-dominated Sorting Genetic Algorithm is a multi-objective genetic algorithm proposed by Deb (2002) [5]. NSGA-II contains the genetic operators present in most genetic algorithms (mutation, crossover and selection) apart from two important mechanisms, namely non-dominated sorting and crowd distancing. Non-dominated sorting sorts a population of solutions into a set of non-dominated fronts $(F_1, F_2, \ldots, F_n)$ where F_1 is the Pareto front and F_{i+1} is the subsequent non-dominated front. The best non-dominated fronts are chosen for crowd-distancing, followed by selection, crossover and mutation. Crowd distancing uses density estimation and a crowded-comparison operator to ensure a diverse spread of solutions [5]. Elitism is incorporated by comparing the current population to the previously found best non-dominated solutions.

Applicability of NSGA-II to the orchard row-detection problem depends on the ability to formulate a population that can undergo mutation, crossover and selection. An individual of a population could be represented as a configuration of rows that define an orchard. Unfortunately, there are many possible row configurations for a given orchard, thus resulting in a vast solution space. Typically, genetic algorithms, including NSGA-II, have a slow convergence time when faced with large or complex solution spaces. This does not rule out the applicability of NSGA-II altogether. A possible antidote would be to use NSGA-II to optimise a partial solution that has already been constructed using domain-specific heuristics.

4.2 Multi-Objective Particle Swarm Optimization (MOPSO)

As discussed earlier, PSO is a promising candidate for solving the orchard row-detection problem but needs a multi-objective counterpart. The first MO extension was proposed by Coello and Lechuga (2002) [4]. To tackle the problem of obtaining a set of non-dominated solutions, MOPSO uses a global repository in which particle trajectories are stored and updated every generation. As non-dominated trajectories are found, the global repository is updated by adding these non-dominated trajectories and discarding any trajectories that are dominated. This global set of trajectories is used to guide particles in subsequent generations. MOPSO was found to be highly competitive with state-of-the-art evolutionary MO techniques, namely NSGA-II and PAES [4]. It is also a highly efficient algorithm for the same reasons that were discussed for PSO.

4.3 Multi-objective Ant Colony Optimization (MOACO)

There are many MO variants of the well-known ACO algorithm, of which a few will be elaborated upon and compared. The first MOACO we'll look at is the multiple objective Ant-Q (MOAQ) algorithm proposed by Mariano and Morales (1999) which was applied to the design of water distribution networks [16]. MOAQ incorporates Q-learning into the standard ACO algorithm such that each ant represents an agent that learns some optimal policy. The environment through which the ants explore was modified from a set of edges to a set of state-action pairs. The value of each state-action pair is updated as per the Q-learning algorithm. In this algorithm, there exists a family of agents for each objective. Each family tries to optimise their objective with all families operating in the same environment (i.e. the same pheromone and heuristic matrices). MOAQ achieves a Pareto front by keeping record of all non-dominated solutions achieved over each iteration of the algorithm. MOAQ optimises each objective according to some pre-specified level of importance, similar to lexicographic ordering. This is a significant drawback of MOAQ, as it biases agents towards certain actions.

Another ACO variant called COMPETants was devised by Doerner et al. (2001) and was initially used to solve bi-objective transportation problems [6]. COMPETants use two ant colonies, each with their own pheromone and heuristic matrices. The ant colonies traverse their respective environment as in the normal ACO algorithm, with the exception of a few ants known as spies. These ants combine information from both environments to decide which edges to traverse. Another interesting characteristic of COMPETants is that the number of ants in each colony do not remain fixed but vary from generation to generation. The colony that constructs better solutions in the previous generation gets more ants in subsequent generations.

The last MOACO we'll look at is the Pareto ant colony optimisation (P-ACO) algorithm which was applied to the multi-objective portfolio selection problem [7]. P-ACO, just like COMPETants, uses several pheromone matrices, one for each objective. It is different in that ant colonies aren't confined to information from their own pheromone matrix. Each ant computes a set of weighted values for each of the matrices, which they then combine to determine which edge to traverse next. To construct the Pareto front, all non-dominated solutions are recorded in an external set, as in the previous MOACO algorithms.

A comparative study [11] established a taxonomy of the existing MOACO algorithms using the bi-criteria Travelling Salesman Problem (TSP). They found that MOAQ and COMPETants had the highest extent of non-dominated fronts but a bad covering over the central part of the Pareto fonts [11]. It was found that the non-dominated solutions of these algorithms converged towards one of either objectives. P-ACO, had the opposite behaviour, in that the

solutions had a good covering over the entire Pareto front but performed poorly. It is worth noting that, in P-ACO, ants combine information of pheromone matrices in such a way that gives automatic bias towards compromise solutions.

5. Discussion

In this paper we have covered a variety of methods that could be used to solve the orchard row finding problem. Each of these approaches vary drastically from one another and as such have differing advantages and disadvantages. To compare these methods, we have collated information with regards to several key metrics. The data is visualised below.

	Accuracy/ Pareto extent	Pareto Spread	Ease of implementation	Computational Complexity	Parameterised Complexity	Space Complexity
Hough Transform	POOR	N/A	AVG	GOOD	POOR	GOOD
PSO	AVG	N/A	GOOD	GOOD	GOOD	GOOD
ACO	AVG	N/A	GOOD	AVG	AVG	AVG
NSGA-II	GOOD	GOOD	POOR	POOR	GOOD	AVG
MOPSO	GOOD	GOOD	GOOD	GOOD	AVG	GOOD
MOAQ	GOOD	AVG	AVG	POOR	POOR	AVG
COMPETants	GOOD	POOR	GOOD	POOR	POOR	AVG
P-ACO	POOR	GOOD	GOOD	POOR	POOR	AVG

Table 1: Taxonomy of reviewed methods comparing eligibility as per several key metrics.

The first method under consideration is the Hough transform. Unfortunately, the Hough transform requires a predefined boundary of a shape or line to detect that shape or line in an image. Since orchard rows can take on arbitrarily shaped curves, this method is inapplicable. The approximation of arbitrary curves using component curves has been explored but with little success [23]. This inherent difficulty in the parameter space has led us to declare this method as having poor parameterised complexity.

PSO and ACO were both promising candidate solutions except for one common deficiency: they don't produce a non-dominated set of optimal solutions. That is, they don't solve multi-objective problems. The absence of this criterion means that neither of these algorithms can simultaneously find the optimal solution for two varying orchard inputs. This is because differing orchards may prefer different objectives.

The next class of algorithms is the class of multi-objective evolutionary algorithms. NSGA-II is generally the evolutionary algorithm against which all others are compared. As can be seen in Table 1, this is for good reason. NSGA-II outperforms all other MOACO algorithms in both Pareto extent and Pareto spread when applied to the benchmark task of bi-criteria TSP [11]. The greatest difficulty in applying NSGA-II is defining individuals or candidate solutions that can undergo all the genetic operators defined by NSGA-II. This algorithm is also notoriously slow, especially when individuals are ill-defined. This brings us to our two most promising approaches: MOPSO and MOACO. These two types of algorithms can be easily defined in terms of graph traversals, which greatly improves the ease of implementation. MOPSO may be favoured over the MOACO algorithms because of its small time and space complexity. MOAQ and COMPETants had a time complexity comparable to that of NSGA-II. When considering the main criterion of performance, both algorithms produce reliable results. It is worth noting that MOAQ and COMPETants outperformed other MOACO algorithms when applied to the bi-criterion TSP [11].

6. Conclusion

This review investigates the potential of a variety of algorithms for applicability to the orchard row finding problem. Previous works in this domain typically utilise common techniques in image processing and image analysis. Given the recent rise in the fidelity of object detection, we plan to use an existing object detector as the basis for row-detection rather than common image processing techniques. These object detections result in a reduction of the problem description to the traversal of edges and nodes in a graph. This opens the avenue of possible techniques to all methods relating to combinatorial optimisation and graph theory. There is, as far as we're aware, no other methods that employ this approach to the problem of finding rows. We have looked at a range of applicable algorithm with a focus on evolutionary MO. We have narrowed these methods down to the two most promising candidates: MOPSO and MOACO. This conclusion is made according to our prioritised criteria of Pareto extent and ease of implementation.

References

[1] Åstrand, B. & Baerveldt, A.J.J.M. 2005. A vision based row-following system for agricultural field machinery. 15(2):251-269.

[2] Bah, M.D., Hafiane, A. & Canals, R. Eds. 2017. Weeds detection in UAV imagery using SLIC and the hough transform. IEEE. 1-6.

[3] Ballard, D.H. 1987. Generalizing the Hough transform to detect arbitrary shapes. In Readings in computer vision. Elsevier. 714-725.

[4] Coello, C.C. & Lechuga, M.S. Eds. 2002. MOPSO: A proposal for multiple objective particle swarm optimization. IEEE. 1051-1056.

[5] Deb, K., Agrawal, S., Pratap, A. & Meyarivan, T. Eds. 2000. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. Springer. 849-858.

[6] Doerner, K., Hartl, R.F. & Reimann, M. 2001. Are COMPETants more competent for problem solving? The case of a multiple objective transportation problem.

[7] Doerner, K., Gutjahr, W.J., Hartl, R.F., Strauss, C. & Stummer, C.J.A.o.o.r. 2004. Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection. 131(1-4):79-99.

[8] Dorigo, M. & Di Caro, G. Eds. 1999. Ant colony optimization: a new meta-heuristic. IEEE. 1470-1477.

[9] Dorigo, M. & Stützle, T. 2019. Ant colony optimization: overview and recent advances. In Handbook of metaheuristics. Springer. 311-351.

[10] Duda, R.O. & Hart, P.E.J.C.o.t.A. 1972. Use of the Hough transformation to detect lines and curves in pictures. 15(1):11-15.

[11] García-Martínez, C., Cordón, O. & Herrera, F.J.E.J.o.O.R. 2007. A taxonomy and an empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP. 180(1):116-148.

[12] Iredi, S., Merkle, D. & Middendorf, M. Eds. 2001. Bi-criterion optimization with multi colony ant algorithms. Springer. 359-372.

[13] Jevtić, A., Quintanilla-Dominguez, J., Cortina-Januchs, M.G. & Andina, D. Eds. 2009. Edge detection using ant colony search algorithm and multiscale contrast enhancement. IEEE. 2193-2198.

[14] Jiang, G.-Q., Zhao, C.-J. & Si, Y.-S. Eds. 2010. A machine vision based crop rows detection for agricultural robots. IEEE. 114-118.

[15] Kennedy, J. & Eberhart, R. Eds. 1995. Particle swarm optimization. IEEE. 1942-1948.

[16] Mariano, C.E. & Morales, E.J.I.M.d.T.d.A., Technical Report HC-. 1999. A multiple objective ant-Q algorithm for the design of water distribution irrigation networks.

[17] Nezamabadi-Pour, H., Saryazdi, S. & Rashedi, E.J.S.C. 2006. Edge detection using ant algorithms. 10(7):623-628.

[18] Nolan, A., Park, S., Fuentes, S., Ryu, D. & Chung, H. Eds. 2015. Automated detection and segmentation of vine rows using high resolution UAS imagery in a commercial vineyard. 1406-1412.

[19] Roberge, V., Tarbouchi, M. & Labonté, G.J.I.T.o.I.I. 2012. Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning. 9(1):132-141.

[20] Rovira-Más, F., Zhang, Q., Reid, J. & Will, J.J.P.o.t.I.o.M.E., Part D: Journal of Automobile Engineering. 2005. Hough-transform-based vision algorithm for crop row detection of an automated agricultural vehicle. 219(8):999-1010.

[21] Shi, Y. & Eberhart, R.C. Eds. 1999. Empirical study of particle swarm optimization. IEEE. 1945-1950.
[22] Sklansky, J.J.I.T.o.c. 1978. On the Hough technique for curve detection. (10):923-926.
[23] Soares, G.A., Abdala, D.D. & Escarpinati, M. Eds. 2018. Plantation Rows Identification by Means of Image Tiling and Hough Transform. 453-459.
[24] Tian, J., Yu, W. & Xie, S. Eds. 2008. An ant colony optimization algorithm for image edge detection. IEEE. 751-756.
[25] Vidović, I., Scitovski, R.J.C. & agriculture, e.i. 2014. Center-based clustering for line detection and application to crop rows detection. 109:212-220.
[26] Yang, Q. & Yoo, S.-J.J.I.A. 2018. Optimal UAV path planning: Sensing data acquisition over IoT sensor networks using multi-objective bio-inspired algorithms. 6:13671-13684.