UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE

CS/IT Honours Final Paper 2021

Title: **Orchard Structure Classification utilizing Hausdorff Distance**

Author: Chelsea Van Coller

Project Abbreviation: ORCHSC

Supervisor(s): Patrick Marais

Category	*Min*	*Max*	Chosen
Requirement Analysis and Design	*0*	*20*	0
Theoretical Analysis	*0*	*25*	0
Experiment Design and Execution	*0*	*20*	5
System Development and Implementation	*0*	*20*	20
Results, Findings and Conclusions	*10*	*20*	20
Aim Formulation and Background Work	*10*	*15*	15
Quality of Paper Writing and Presentation	*10*		10
Quality of Deliverables	*10*		10
Overall General Project Evaluation (*this section allowed only with motivation letter from supervisor*)	*0*	*10*	
Total marks		**80**	**80**

Orchard Structure Classification utilizing Hausdorff Distance

Chelsea Van Coller
VCLCHE001
University of Cape Town
VCLCHE001@myuct.ac.za

ABSTRACT

As the human population continues to rise at an exponential rate the need for food security has become a pertinent issue. Farmers are looking to improve their crop yield, whilst maintaining the same land. Precision Agriculture has become a relevant market that allows farmers to easily maintain their crops. This paper studies identifying different planting patterns within aerial images of orchards. These patterns affect yield, accessibility, cross pollination and irrigation. The Hausdorff Distance is applied to the abstracted data from the aerial images to detect these patterns through a matching algorithm. The matching algorithm matches the orchard data against point sets containing the pattern to be matched. Three modifications of the Hausdorff distance are applied: exact, partial and average. Our experiments reveal that the average Hausdorff distance provides the best results, when matching pattern, as it correctly handles noisy orchard data.

CCS Concepts

• Optimization algorithms • Computational Geometry

KEYWORDS

Precision Agriculture, Hausdorff Distance, Pattern Detection

1 INTRODUCTION

Food security is a large problem faced globally with many factors influencing it such as high population growth, global warming and poor agricultural systems [1]. As the human population continues to rise at an exponential rate the need for food security has become a pertinent issue[2]. Agriculturalists have to look to alternative methods to produce larger amounts of food whilst facing land constraints. This has given rise to the concept of precision agriculture. This is a practise that assists farmers in making management decisions through abstracting data from technology such as satellites, drones, robots and sensors. The data collected via the technology allows farmers to monitor their crops more closely and efficiently.These findings can be used to evaluate yield, soil condition, plant health, fertilizer and pesticide effect and irrigation.

This study serves to analyze different planting patterns found within orchards. Farmers organize their orchards into different planting patterns to assist with the many aspects of farm management, such as efficient irrigation and cross-pollination, accessibility for farm equipment and for ease of harvesting. These structures affect the system management recommendations produced through precision agriculture, and assist in providing a strong preliminary for detecting individual trees, as well as identifying locations where trees are missing.

There have been many studies conducted over the years to determine the most optimal structure and density for plants per unit area. There are many published studies, yielding somewhat inconsistent results, which makes it challenging to determine one specific pattern that is the best at yield production, holding all other factors constant[4][5][6] . Using precision farming to detect these patterns in orchards and accurately determine the most suitable structures based on the real data. The study focuses on identifying and classifying various planting patterns and recovering their parameters. This is achieved by using aerial imagery produced by unmanned aerial vehicles, provided by Aerobotics, a South African based precision agriculture company. The trees within the images are identified as polygons, which are then converted to a point set. The problem has been recast to a point pattern matching problem under rigid motion.

The proposed solution uses a matching technique, whereby each planting pattern to be matched has an associated point set that is matched against the data. An optimisation algorithm that utilizes the Hausdorff distance to measure the difference between the point sets, is implemented to establish whether the data matches the pattern. A number of planting patterns must be identified within the orchards, namely the square, rectangle, quincunx, double row and hexagonal pattern. For the double row pattern there are three parameters to be recovered: Intra-pair spacing, inter-pair spacing and row spacing.

This paper is laid out as follows: Section 2 introduces related works surrounding planting patterns and the Hausdorff distance.Section 3 discusses the design and implementation of the experiment. Section 4 displays the results and findings are deduced. Section 5 concludes the paper and discusses avenues for future work.

2 RELATED WORK

This section discusses the layout of planting patterns and then discusses the Hausdorff distance and the improved Hausdorff, along with improving the technique.

2.1 Planting patterns

Farmers have used different planting patterns for years in an attempt to produce better yields for their crops and maximize the number of trees per hectare. This paper only looks at 5 common planting patterns, square, rectangle, double row, quincunx and hexagonal. The square pattern is a commonly used pattern that allows for intercropping and bidirectional cultivation[18]. A quincunx pattern utilizes the extra space within the square pattern and places a plant in the middle of the square. The additional tree/filler tree is generally a short lived tree that is replaced after the main trees are bearing, as it creates additional competition [18]. This pattern can be followed when the permanent trees are more than 10m apart. The hexagonal planting pattern is where trees are planted in each corner of an equilateral triangle. Different planting distances between the trees are used for different types of trees, as certain trees have roots and branches that spread over a larger area. Thus the distance between patterns will differ for different orchards.

2.2 The Hausdorff distance

The Hausdorff distance's objective is to find the difference between two points sets, by measuring how far the sets are from one another. The metric is fairly simple, given two point sets $A = \{x_1, x_2, \ldots, x_n\}$and $B = \{y_1, y_2, \ldots, y_m\}$ the Hausdorff distance is defined as

$$(1)\ H(A,b) = max(h(A,B), h(BA))$$

where

$$(2)\ h(A,B) = max_{x \epsilon A}(min_{y \epsilon B}||x - y||)$$
$$(3)\ h(B,A) = max_{x \epsilon B}(min_{y \epsilon A}||y - x||)$$

Where $||\ \ ||$acts as the normaliser. It is assumed in this paper that this is the Euclidean norm. (1) is known as the undirected Hausdorff distance and (2) and (3) are known as the directed Hausdorff [12]. To summarize, each point within the point set A determines the closest point in point set B and vice versa. Of those points, the maximum distance is found in point set A and the maximum is taken of each h(x, y) function. If the maximum is zero, then the points match perfectly. **Figure 1** illustrates the concept, showing how the final Hausdorff distance will be calculated by the largest distance from the closest point.

This is the basis for solving many computer vision related problems such as pattern recognition [7, 8], shape matching [9] and measure of similarity[10] [11].

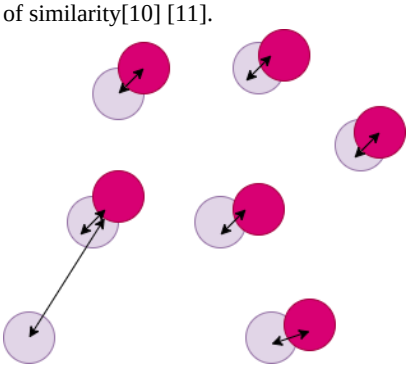

Figure 1: The Hausdorff distance between two point sets

2.3 The improved Hausdorff Distance

There are a number of problems relating to this approach namely the stringent nature for penalising outliers and the time complexity. Huttenlocher et al. [12] presents an extension of the original algorithm that is extended to rigid motion. It accounts for and is tolerant of dissimilarity between the images, whereas previous work would reject dissimilar images. This is achieved by taking the Kth ranked point rather than the max in (2) and (3).

Where (3) $h_K(B,A) = K^{th}_{x \epsilon B} min_{y \epsilon A}|\ \ |y - x|\ \ |$

The Kth ranked value indicates the Kth ranked values as opposed to the maximum. It allows for a portion of the model and image to be matched thus allowing for scenes containing many objects and objects partially obscured from view [12].

However, this adds additional complexity, thus computation takes twice as long [12]. The average Hausdorff distance can also be used as an alternative to creating a more robust detection [13]. Where (2)

$h_{mean}(A,B) = \frac{1}{N}\sum_{x \epsilon A}\ \ (min_{y \epsilon B}||x - y||)$

The average Hausdorff is particularly useful in the medical field when segmenting tasks that contain particularly complex boundaries and small vessels [13].

2.4 Improving efficiency

The brute-force algorithm to compute Hausdorff distance requires O(n x m) time, as finding the nearest point for each point is a costly process [11]. Alt et al. [14] proposes a faster solution that can calculate the Hausdorff-distance in time $O(n\ log\ n)$, where n is the number of vertices. It is based on using a Voronoi diagram to quickly identify the location of the shape to be matched. Chew et. al [15] improve upon this algorithm by implementing a parallel algorithm using the technique of parametric searching that greatly speeds up the original runtime.

Kharbach [16] proposes using a genetic algorithm in order to improve upon this search speed, which yielded poorer results opposed to the linear search, but may produce better results in a larger sample size. Papidas [17] proposes a solution using aggregate nearest neighbor queries in spatial databases. This results in an efficient processing system as well as allowing for techniques to be easily integrated for solving related problems.

2.5 Improving the optimisation function

As the Hausdorff distance is required to be calculated for every possible translation within the data, this results in a large runtime. Huttenlocher[12] attempts to solve this by implementing different pruning techniques to speed up the process. These techniques are ruling out circles within the data, early scan termination for points that exceed the threshold. Skipping forward is the next technique that relies on large spaces within the data that can be skipped to reduce the number of calculations. Danilove et al.[22] propose an optimised solution for convex polygons that uses super operative memory and the exclusion of certain polygons and vertex pairs. Wang et al. [23] propose an optimisation technique that uses a segmentation approach and a search path.

There is a large amount of research into the Hausdorff distance. Most research uses the metric to find items within an image, however there is sufficient work utilizing point sets. Currently there is no work surrounding identifying planting patterns within crops, however this approach is likely to have some merit within pattern detection.

3 DESIGN AND IMPLEMENTATION

The design and implementation discusses the data set used and the data structures used to contain the data. The preprocessing implementation is discussed in depth along with the Hausdorff distance and matching algorithm implementation. The experiment set up is discussed and evaluation for the experiment declared.

3.1 Dataset

Aerobotics provided 7 datasets of extracted data from aerial images of different orchards. The details of each orchard can be seen in **Table 1**. Each dataset consists of 2 geojson files and 2 tiff files.

The predominant dataset used in this study is raw geojson files of the orchard. Each tree is indicated via a polygon, the geojson file contains all the coordinates of the polygon, alongside the confidence level of each polygon. These are used to determine the certainty of the tree's presence within the dataset. The second geojson file contains the boundary data for each polygon

3.2 Data Structures

The predominant data structure used within this study is the Point structure produced by the Shapely Library. These points contain the x and y coordinates of their location. The data structure to contain these points is a Multipoint set. The multipoint set can be easily manipulated by affine transformations and different metrics, however it cannot contain other attributes. To accommodate for this

a Pattern Node object which stores the Point, best matching shape and confidence of the correct pattern estimation. This confidence figure is produced by the Hausdorff distance metric. The lowest Hausdorff distance indicated the closest match thus indicated the correct shape.

There are a number of metrics used within the study. These are:
Euclidean Distance: Returns the distance of the line between two points.
Convex hull: Returns the smallest convex containing all the points in the object
Minimum rotated rectangle: Returns the smallest rectangle containing all the points. It is not constrained by the axises thus allowing for a rotated rectangle.

3.3 Preprocessing

The data needs to undergo preprocessing in order to be effective in measuring the Hausdorff distance. The abstraction, transformation and dataset creation are discussed.

Abstracting data

In order to manipulate the trees, they are abstracted into a point set. Firstly, we calculate the threshold confidence level for which trees will be used within the data set. This ensures that trees with very low confidence intervals will be excluded, as we want to ensure that all trees included will be within one standard deviation of the mean confidence interval. The centroid of each polygon is abstracted to create a point set. This process can be seen in **Figure 2**. The centroids are likely to be slightly distorted, as the trees may have grown in different directions, resulting in different sized polygons. This means that not all the data will be in a straight line. The point set is stored in a Multipoint data structure that allows for affine transformations to be applied.

Transforming the data

Scaling the data correctly is a crucial part of the Hausdorff distance as it is important to apply match point sets that are of the same scale, otherwise the metric will not be able to correctly identify patterns. The ideal scaling is to have each point column one unit apart. In order to do this, the convex hull of the multipoint data set acts as the area. The density is then calculated by dividing the number of trees within the data set by the area. The multipoint set is then scaled by this density metric. In order to correctly align the data within the axis the minimum rotated rectangle is taken of the multipoint set. The bottom left point of the rectangle is translated to the origin and it is rotated down to the x-axis. This is an initial attempt to account for rotation.

Model Dataset creation

The next important set is creating the model set that will be matched against the data. Simple point sets consisting of the 3 main patterns to be matched are the square, quincunx, hexagonal and double row planting pattern. These patterns are created and stored with multipoint sets. The patterns need to be standardized to create a universally accepted scaling method. This is achieved by having each shape have a column width of 1 unit. The square pattern is created by creating 4 points that are all 1 unit apart from the adjacent point. This is then extended to fill the model size, which is indicated by the window variable. This variable is discussed in the pattern matching section. The quincunx is an extension of the square pattern that simply places a point in the centre of the square pattern. The double row pattern is created by implementing the square pattern for one row, then skipping a column of one unit, then implementing another row of the square pattern. This is continued by creating continuous rows. The rectangle pattern is implemented by appending points within the rectangular shape with the column width of 1 unit and the row height of 0.5 units. The triangular shape deviates most from the other shape as it requires the triangle to be equilateral, thus the column width remains at one unit however the height of the triangle becomes 0.866. These shapes can all be seen in **Figure 3**.

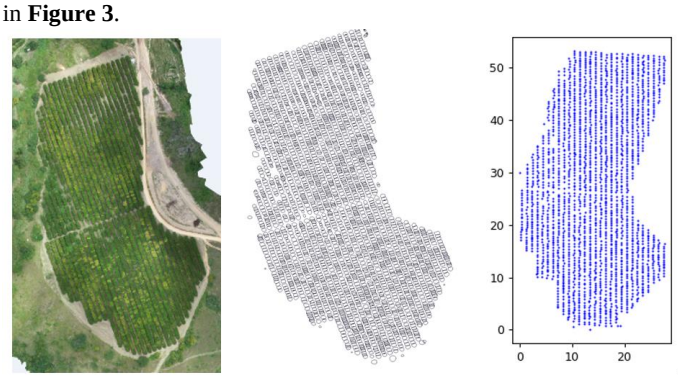

Figure 2: Centroid abstraction process

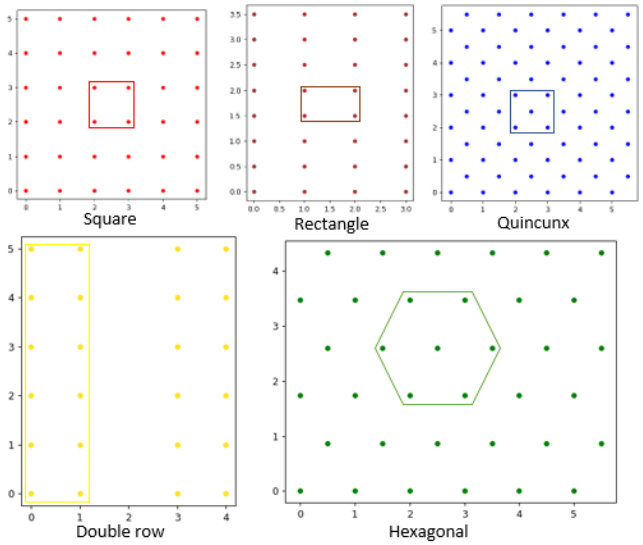

Figure 3: The different model datasets produced

3.4 Hausdorff Distance

Three implementations of the Hausdorff Distance were tested to establish the best approach to this method. These approaches are namely the directed Hausdorff distance, the average Hausdorff and the partial Hausdorff. Each implementation shares similar features that reduce inconsistencies and improve time.

Reducing inconsistencies and improving runtime:

As explained in related work, the Hausdorff distance is a computationally expensive calculation of $O(n^2)$, as the nearest neighbour is calculated for all points within the model and data set. As this metric needs to be called multiple times within the experiment it was important to significantly reduce this complexity. This is achieved through using a CKD-tree. This tree structure is a binary tree that stores points in a multi dimensional space. When searching for the nearest point, it recursively divides the space thus reducing the search space by half each iteration[32]. It can effectively perform the nearest neighbour search with a query time of $O(log(n))$ [19]. This greatly reduces runtime.
Usually when implementing the Hausdorff distance, the model would be much smaller than the data set being matched. However, as we are finding patterns within the data, the model data set needs

to be the same size as the data set. In order to account for the case where the data is rotated, the model is created slightly larger than data. However, in the Hausdorff distance this will account for a larger metric in order to encompass the additional points. These surrounding model points will skew the data. To accommodate for this a convex hull is drawn around the data to indicate the points that may be matched. If the point is within the convex hull the closest distance will be considered as the maximum value.

Modified Exact Hausdorff Distance

The modified exact Hausdorff will be used as the basis of comparison for all other methods. It is constructed using the basis behind the Directed Hausdorff distance, however it includes the convex hull to minimize the distance.

Algorithm Modified Exact Hausdorff Distance

Input: Point Set Model, Point Set Data
Output: Hausdorff distance from Model to Data

CKD Tree C_D ←Initialize KDTree with Data
Distance array D_M← distances set for Model
Convex Hull H← convex hull(Data)
for each Point m in model **do**
  if H contains m
    d_M← max{DistanceToNearestNeighbor(m, C_D)}
CKD Tree C_M ←Initialize KDTree with Model
Distance array D_D← distances set for Datal
for each Point d in data **do**
    d_D← max{DistanceToNearestNeighbor(d, C_M)}
    for each point in distances set:
h_d← max(d_M, d_D)
return h_d

The Average Hausdorff Distance

This is an approximation of the Hausdorff distance which allows for outliers not to be heavily penalised. Instead of returning the maximum, it will return the average of the closest point for each directed Hausdorff distance (2) and (3). The undirected Hausdorff distance will then return the maximum of the two.

Algorithm Modified Average Hausdorff Distance

Input: Point Set Model, Point Set Data
Output: Hausdorff distance from Model to Data

CKD Tree C_D ←Initialize KDTree with Data
Distance array D_M← distances set for Model
Convex Hull H← convex hull(Data)
for each Point m in model **do**
  if H contains m
    d_M← average{DistanceToNearestNeighbor(m, C_D)}
CKD Tree C_M ←Initialize KDTree with Model
Distance array D_D← distances set for Datal
for each Point d in data **do**
    d_D← average{DistanceToNearestNeighbor(d, C_M)}
    for each point in distances set:
h_d← max(d_M, d_D)
return h_d

The Kth Partial Distance

The Kth partial distance takes the Kth ranked distance as opposed to the largest one. In order to achieve this, a fraction is specified $0 \leq f_1 \leq 1$, in this case $\frac{1}{5}$is used. The Kth ranked value is then given by $\lfloor f_1 q \rfloor$where q is the maximum distance. Of the Kth ranked values the minimum is computed thus returning the directed Hausdorff Distance, the directed Hausdorff distance is then computed using the maximum.

Algorithm Modified Partial Hausdorff Distance

Input: Point Set Model, Point Set Data
Output: Hausdorff distance from Model to Data

CKD Tree C_D ←Initialize KDTree with Data
Distance array D_M← distances set for Model
Convex Hull H← convex hull(Data)
K_D ←floor(0.25*length(Data))
K_M ←floor(0.25*length(Model))
Array A_M ← initialize
for each point m in model in K_M rank
  if H contains m
    d_M← max{DistanceToNearestNeighbor(m, C_D)}
    append d_M to A_M
  increment K_M
M_M← min(A_M)
CKD Tree C_M ←Initialize KDTree with Model
Array A_D ← initialize
for each point din data in K_D rank
  if H contains d
    d_D← max{DistanceToNearestNeighbor(d, C_M)}
    append d_D to A_D
  increment K_D
M_M← min(A_M)
h_d← max(M_M, M_D)
return h_d

3.5 Matching Algorithm

The algorithm matches all the different pattern point sets to the data. The pattern that is selected as the best match will produce the lowest Hausdorff distance. However, when matching the model and the data, they are unlikely to align, as they may be at different orientations. As a result poorer Hausdorff distances would be produced. To accommodate for this, affine transformations are applied to the model to provide the correct match, which is achieved through a number of rotations and translations. There are two main considerations when matching the data and the model. Although the Hausdorff distance is an effective metric, it is a fairly expensive metric, thus one needs to reduce the number of calls to make the matching process efficient. Another consideration is that it will not be able to distinguish between different patterns within an orchard using one Hausdorff distance calculation.

Matching multiple patterns

To match different patterns within the data, a window variable is set to determine the size of the data that will be sampled, i.e. a window of size ten will sample data in a 10 by 10 region within the data set. A window is used as the points within the multipoint cannot be

accessed in a specific area through indexing. The point set within the specific area is accessed through a minimum rotated rectangle. The model of size 2 + the window size will be matched to the data in this area. Although additional points add computational complexity, it is necessary to create a larger model for the case of rotation, as the point set must stay within the bounds of the image. It also allows for better matching as morphed data can be segmented.

Approach to improving efficiency

In most papers seen on the topic [12][15][29][35], the model being matched is a specific shape or image, thus it needs to be translated incrementally through the entire dataset to find the best match. However in this circumstance the model is formed of multiple uniform points, this indicates that it is not necessary to translate the model through every point as it will be matched to the same point. Thus when matching the model to the data, it only needs to be translated within the entire area of one completed pattern. Within this experiment, all models are translated within a 1 by 1 square of increments of size 0.25. The only exception is the double row pattern, which is required to be translated within a rectangle of width 2 and length 1. This is due to the nature of the pattern containing missing rows. These increments will be iterated through the area, returning the Hausdorff distance for each match. The lowest Hausdorff distance indicates the correctly translated model. This approach allows for a large reduction in complexity.

Huttenlocher [12] and Chew et. al[15] suggest pruning parameters that will assist in speeding up translation, however these techniques are unlikely to work in this circumstance as the dataset is fairly dense.The next transformation to be accounted for is rotation. The pattern is unlikely to be correctly orientated, thus for each incremental translation, the correct rotation needs to be produced. Each pattern is rotated in increments of 15°. To improve efficiency, not all patterns are rotated the full 360°. The square pattern rotated by 90° will produce the same result, thus reducing the number of rotations. The quincunx pattern follows the same understanding. The double row pattern needs to be rotated 180° as it has unequal sides. The rectangle pattern will also follow this. However the triangular pattern needs to be rotated the full 360°, as every angle needs to be compensated for. For each small increment, all the rotations need to be performed, which contribute significantly to the computational complexity of the algorithm.

Algorithm Matching Algorithm

Input: Point Set Data, Window Size, Implement Rotations bool
Output: Final matching pattern

Square, Rectangle, Double Row, Quincunx, Hexagonal ← Initialize patterns with window length
for each pattern:
  for window size within data set:
    Multipoint M_w ←detected points within window
    MatchingModel B ←initialize best match
    Number Increments and rotations I_a, R_y ←allocated to each shape
    for y in I_a
      for x in I_a
        for z in R_y
          distance ← Hausdorff Distance(M_w)
          if B > distance:
            B ←MatchingModel(distance, M_w)
            rotate M_w
        translate(x)
      translate(y)
    return best pattern rotation
return best pattern within window

Speeding up rotations

As rotations contribute a large amount to the complexity of the matching algorithm, another algorithm is introduced that approximates rotation within each window. This is achieved by sorting the data set of points into ascending values of X, then iterating through the first 20 points within the data set. Then these points are used to fit the least squares regression line. The angle at which this line is positioned acts as the angle for the set to be rotated by. This significantly reduces the number of rotations.

3.6 Road finding algorithm

Detecting a round or boundary is not possibly using the Hausdorff Distance approach. One could try matching a pattern that contains only points and an empty space, thus representing the road, however this is unlikely to work as there are too many variables to consider. Such as the size of the road, whether it curves, etc. Hinton [20] discusses how it is a very challenging problem to solve. They note that after 30 years of research there is still no road detection published or on the market. Of the attempts to solve this problem, the solutions involve using neural networks that are trained on millions of datasets [20][21]. Unfortunately due to the nature of the problem, this was not achieved in this paper.

3.7 Software Implementation

This solution is implemented in Python, as there are many useful libraries for computer vision. The predominant library used is Shapely, as it contains effective data structures for manipulating points. It allows for affine transformations to be easily applied. Scipy Spatial is used to implement the KDTree for identifying nearest neighbours. Matplotlib is used to plot results and derive findings from the implementation.

3.8 Experimental Setup

All experiments were run on a laptop with 8GB random access memory. A Intel(R) Core(TM) i5-4200U processor was used. A solid state drive was used, along with a 64 bit operating system.
Windows 10 is run on the operating system and all experiments were run through the IDE Pycharm.

3.9 Evaluation

The main method for evaluating the patterns identified within the data is visually detected. Each dataset will be run in the experiment and assessed by 4 categories. These categories are:

- Was the correct pattern detected?
- The frequency of undetected patterns
- The frequency of incorrectly identified patterns
- The Hausdorff distance correctly aligned the model and the data

These will be assessed via a scoring system from 1 to 5. 5 being the best score and 1 being the worst score. Two tests will be run, one including the rotation iterations and the other preprocessing the data to the correct orientation. Thus, the times can be compared easily.

4 RESULTS AND DISCUSSION

Each pattern has been assigned a unique colour, to differentiate between patterns. The square pattern is represented by red, the quincunx is blue, the double row is yellow, the hexagonal green, the rectangle is brown and no match is indicated by black.

4.1. Pattern Observations

Firstly when studying the patterns there are a few points of failure for this approach.When comparing the square and the quincunx pattern, it is noted that they are very similar.

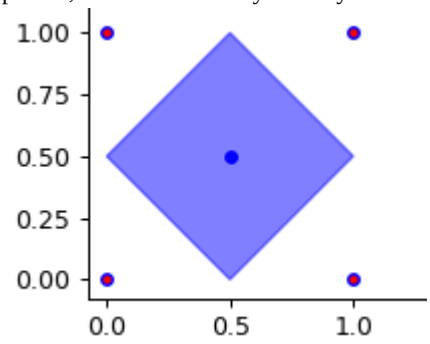

Figure 4: Square and quincunx pattern comparison

Observing **Figure 4**, the red points represent the square shape and the blue, the quincunx shape. Represented here is a polygon displaying the area in which (exclusive of the boundary) a point would generate a lower Hausdorff distance for the quincunx pattern. By this token, any outliers contained within this area in a square pattern would detect a quincunx pattern.

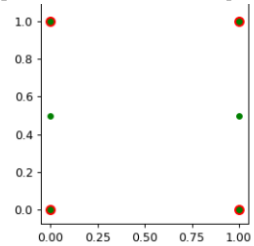

Figure 5: Rectangle and square pattern comparison

Next, compare the rectangle pattern and the square pattern seen in **Figure 5**. The red points represent the square pattern and green points represent the rectangle pattern. Green is here as opposed to brown to show the contrast between the two patterns. and the One can see that they are very similar, however the rectangle will have a higher density of points.

4.2 Scaling the model

The idea behind scaling the model in accordance to the density. This is a fairly good approximation for scaling, however it does not account for each scale perfectly. As different orchards may contain a higher density of trees or larger distances between trees. The matching algorithm requires the rows to be one unit apart and any deviation in this results in the identification of incorrect patterns. In order to account for this the data needs to be manually scaled for each orchard, thus assuring that the patterns can be correctly matched.

4.3 Observations on template containing every pattern

To compare the base results a generic template is created containing each pattern to test the results. As each template is correctly orientated, the rotation was not implemented within this experiment to assess the base runtime for each algorithm. It is noted that each pattern can correctly be identified in solitude, however when two patterns are contained within the same window the Hausdorff distance cannot correctly compensate for this. It will attempt to match the entire pattern, which results in irregular matches. **Figure 6** shows the mixed pattern produced by the modified exact Hausdorff with a window size of 4. It identifies the correct pattern within the main body of the pattern, however it incorrectly identifies the pattern on the boundary of the two images.Extending this to real data could be problematic as real data contains much more inconsistency, however real world orchards are unlikely to have so many variable planting patterns within one boundary.

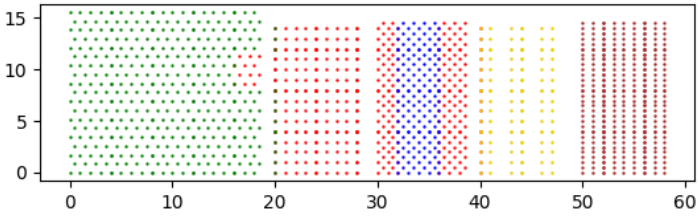

Figure 6: Implementation of Modified Exact Hausdorff Distance over quilted pattern

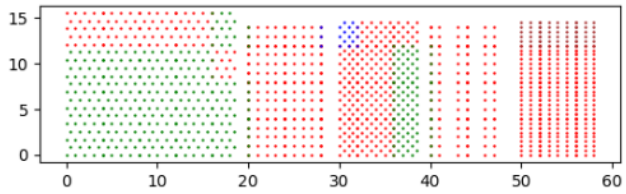

Figure 7: Implementation of the partial Hausdorff distance

The partial Hausdorff distance fails, when identifying multiple patterns within one data set, as it finds partial matches. Therefore, it is able to find a square pattern in almost every iteration, seen in **Figure 7**. There are multiple matches however the square is the first match, thus it is displayed repeatedly within the image.

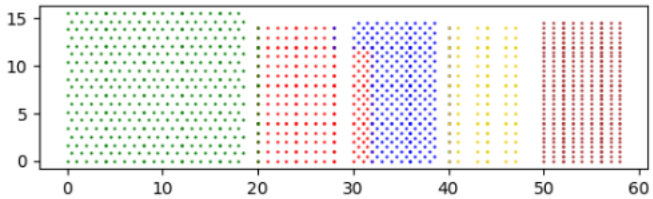

Figure 8: Implementation of the average Hausdorff Distance

The average Hausdorff distance produces the best results, seen in **Figure 8.** There are still areas in which it matches the incorrect pattern, however this is to be expected with the window approach. Reducing the window size is a possible solution, however this will inhibit the pattern from matching the double line pattern as it requires at least 4 columns to be correctly matched.

4.4 Rotation Observations

When finding the correct orientation for the model under the iterative method, the average Hausdorff distance performs the best. The reason for this being that it can account for outlying data better than the other two methods. **Figure 9** represents an incorrect rotation using the modified exact Hausdorff distance. This is as a result of the outlying points at the bottom of the figure. The partial Hausdorff distance also encounters these problems, as it may find a perfect match in one portion of the data set and disregard the unmatched pattern in another.

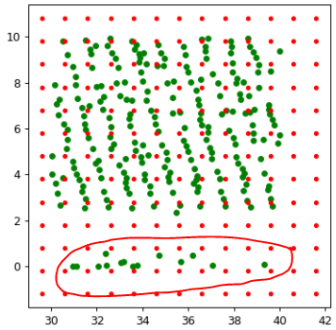

Figure 9: Incorrect orientation using the Exact Hausdorff distance

The least squares of regression provides fairly poor results when detecting a rotated pattern within the data. This is as a result of the erratic data and the window capturing odd proportions of data. It captures vertical data easily, thus rotations reducing rotations, however the angle detected for rotated sets is often incorrectly captured. An alternative approach to this could be using the Hough Transform, as it is known for detecting straight lines within planting patterns [24].

5.5 Window size observations

A window size of 10 has been used as the optimal window in the extensive experiments. This is chosen as a smaller metric does not allow for enough of the pattern to be matched, especially in the case of the partial Hausdorff. A larger pattern increases the area in which an error can be made. It also computes with a longer run time as the CKD tree is much larger.

4.6 Implementation of the Modified Exact Hausodorff

The modified exact Hausdorff produces fairly poor results, as there are many outliers within the orchard data. As seen in **Figure 10**, it produces a quilt -like pattern as it matches different patterns within the data. This can be attributed to the large amounts of white space seen within the unidentified patterns.

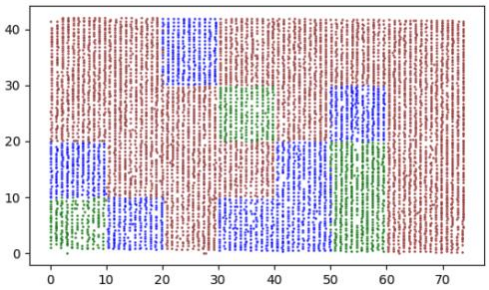

Figure 10: Modified Exact Hausdorff Distance on orchard 43581

However, it still manages to identify the predominant rectangular pattern within the data. The modified exact Hausdorff performs better than anticipated on orchard 36507, seen in **Figure 11.** This can be attributed to the neat nature of the orchard. The square pattern can be detected in a large portion of the data and the predominant pattern is the rectangle. Although this pattern is classified as a rectangular pattern, there may be some merit in identifying the square pattern. As the data experiences outliers within the rows it is likely that there are a number of square patterns.

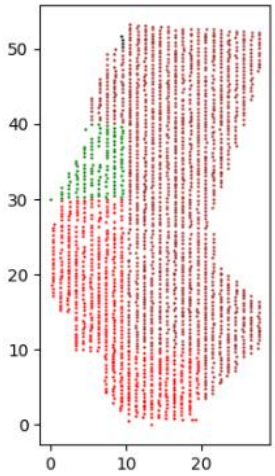

Figure 11:Modified Exact Hausdorff Distance on orchard 36507

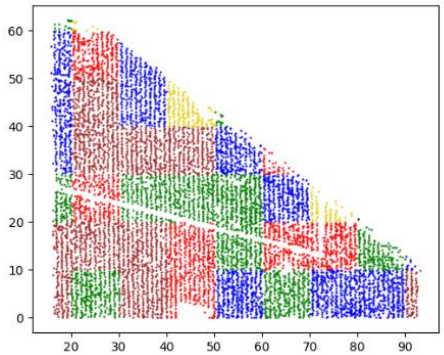

Figure 12: Modified Exact Hausdorff Distance on orchard 35516

Figure 12 demonstrates how the exact Hausdorff cannot effectively identify patterns within very noisy data. Multiple patterns are incorrectly detected. This is a result of the penalizing nature of the metric. The modified exact Hausdorff can perform well within clean data, however for the nature of this study it is a poor metric.

4.7 Implementation of the Partial Hausdorff

The partial hausdorff distance provides a good alternative to the exact Hausdorff distance, however the results produced are somewhat similar. This metric also produces a quilt-like appearance, however it is not depicted within the figures, but for each incorrect pattern detected within **Figures 17, 18** and **19** there is a correct match for the rectangular shape. However, due to the nature of the partial match it is able to detect multiple matches within the data set.

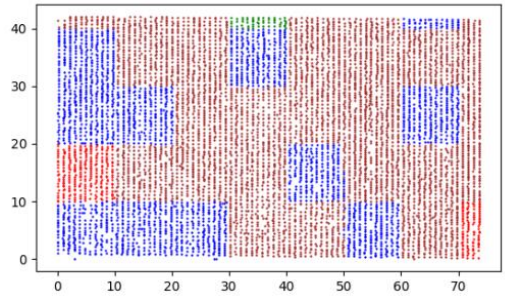

Figure 13: Partial Hausdorff Distance on orchard 43581

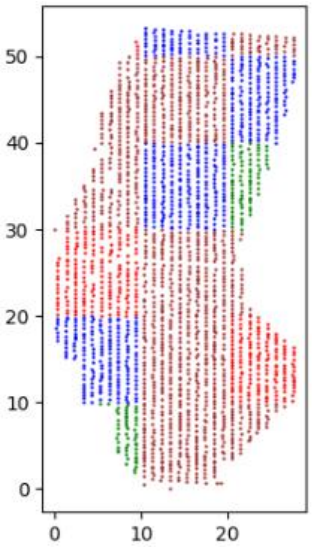

Figure 13 shows how the partial Hausdorff detects similar inconsistencies within the data as the exact Hausdorff. However, it contains more inconsistencies. **Figure 14** also contains more inconsistencies than the exact Hausdorff distance. This trend continues in **Figure 15.**

In order to distinguish between patterns using this approach, it may be viable to implement a weighted partial distance that takes the most common Hausdorff as opposed to the minimum. This would accommodate for outliers, whilst still detecting the predominant pattern.

Figure 14:Partial Hausdorff Distance on orchard 36507

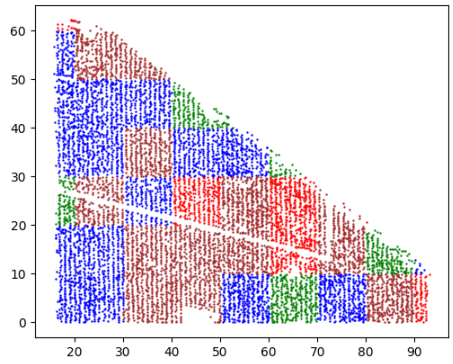

Figure 15: Modified Exact Hausdorff Distance on orchard 35516

4.8 Implementation of the Average Hausdorff

The average Hausdorff performs the best in comparison to the exact and partial Hausdorff distance. It is able to detect the rectangular pattern within all the datasets and it does not encounter multiple pattern matches to the point set. **Figure 16** displays a perfect match for the rectangular pattern within the data. As it approximates the data, it allows for missing trees and distorted rows to be accounted for.

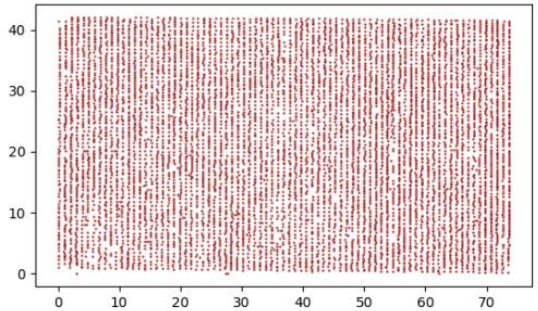

Figure 16: Modified Exact Hausdorff Distance on orchard 43581

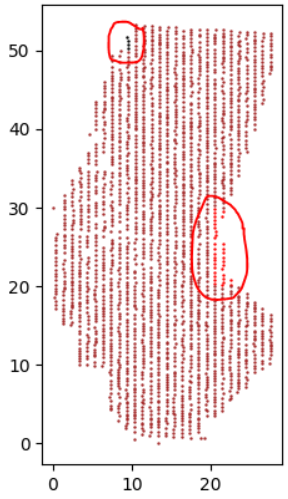

Figure 17 displays a near perfect match, however there are two points of failure within the image. The first point identifies the square pattern as opposed to the rectangle, in this case, it has correctly identified the pattern within the window, however this pattern does not correctly correspond to the rest of the data. The next point of failure is 3 unidentified points at the top of the image. This is owed to the failure of the matching algorithm as 3 points cannot be effectively matched to any pattern.

Figure 17: Modified Exact Hausdorff Distance on orchard 36507

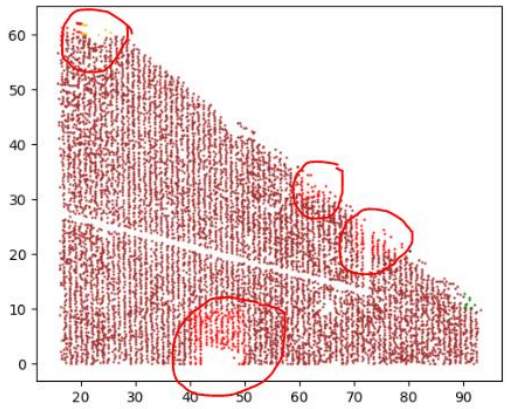

Figure 18: Modified Exact Hausdorff Distance on orchard 35516

Figure 18 demonstrates the multiple points of failure on the edges of the data, this is another result of the window containing too small a dataset. In order to combat this problem, the average Hausdorff distance should only be called if the window contains a sufficient number of points. If it does not the window should be shifted by half an increment to encompass enough points.

Method	Orchard number	Correct pattern Detected	No incorrectly identified patterns	Correct Orientation	Time with rotations (sec)	Time without rotations (sec)
Exact	43581	2	1	5	10138	1683
Partial	43581	2	1	5	12209	2044
Average	43581	5	5	5	10187	1689
Exact	36507	2	2	2	3738	623
Partial	36507	1	1	5	4222	704
Average	36507	4	4	5	4188	682
Exact	35516	1	1	1	9942	1657
Partial	35516	1	1	1	13002	2154

Average	**35516**	**4**	**3**	**5**	**11734**	**1952**

Figure 19: Table summarizing the experiment results from all the tests

Time complexity

The major problem with this approach is the extreme time complexity associated with computing the Hausdorff Distance for many interactions. The larger the dataset the longer the experiment takes, this can be clearly seen in **Figure 19**. Rotations are also shown to increase the run time by 500%. Thus implementing a working algorithm to estimate the correct rotation is crucial for this experiment. Smaller data sets produce faster results, however for a data set containing 2561 trees an implementation including rotations takes just over an hour.

5 CONCLUSIONS AND FUTURE WORK

The average Hausdorff distance proves to be the best metric in comparison to the exact and partial Hausdorff distance. It performs well when aggregating points within noisy datasets and is able to detect the planting pattern within the provided data. The modified exact Hausdorff penalizes outlying points too heavily, thus distorting the pattern shapes detected. The partial Hausdorff distance provides too lenient an approach as it detects multiple patterns within one area due to the outliers within the data Thus it is not a suitable solution for the problem. There are some merits to the average Hausdorff approach as it can provide a good approximation for the patterns contained within a data set. However, as most of the data supplied contains rectangular patterns, it cannot be pronounced a robust method until tested on more data containing the patterns to be detected. As Aerobotics was unable to provide this data during this timeframe, more tests on different data would need to be conducted in the future. The matching algorithm needs to account more accurately for different window sizes by implementing a better approach that accounts for more trees within the window. Reducing runtime is the most important aspect of improving this implementation. This can be achieved through finding an effective pruning process. The least squares regression line attempted to solve this problem through orientating the data thus not using the Hausdorff distance to find the correct orientation. This proved unsuccessful, however further implementations could assess using the Hough Transform as a method to identify lines and then correctly orientate the rows.

Acknowledgements

I would like to thank my supervisor Patrick Marais for his constant guidance through the process and for the fantastic learning experience. I would also like to thank my two group members, Piero Puccin and Jonathon Everatt for their input. Lastly, I would like to thank Aerobotics for supplying the data and for their added guidance.

References

[1] UNITED NATIONS. Sustainable Development Goals. Retrieved May 25, 2021 from https://www.un.org/sustainabledevelopment/hunger/

[2] ZHANG, Q. 2016. Precision Agriculture Technology for Crop Farming. Taylor & Francis, Washington.

[3] JAGANNATH, M. K. 1978. A MODIFIED MODEL FOR PLANT COMPETITION UNDER VARYING PLANT GEOMETRY. Current Science, 47, 8 (April 1978), 251-254. DOI: https://www.jstor.org/stable/24081256

[4] RAZA, M.A., BIN KHALID, M.H., ZHANG, X. et al. 2019. Effect of planting patterns on yield, nutrient accumulation and distribution in maize and soybean under relay intercropping systems. Sci Rep 9, 4947 (2019). DOI: https://doi.org/10.1038/s41598-019-41364-1

[5] PANT. M. 1979. Dependence of Plant Yield on Density and Planting Pattern. *Annals of Botany*. 44, 4 (October 1979), 513-516. DOI: https://www.jstor.org/stable/42756640

[6] PANDEY, P., PRASHANT,S., BALZTER, H., AND BHATTACHARYA, B. 2020. Hyperspectral Remote Sensing: Theory and Applications. Elsevier.1(2020), 121-146. DOI: https://doi.org/10.1016/B978-0-08-102894-0.00009-7

[7] NI, B., HE, F., PAN, Y., YUAN, Z. 2016 Using shapes correlation for active contour segmentation of uterine fibroid ultrasound images in computer-aided therapy. Appl Math Ser B. (2016); 31(1): 37-52.

[8] SUDHA, N. Robust Hausdorff distance measure for face recognition. Pattern Recogn. (2007); 40(2): 431-442.

[9] ALT, H., BEHRENDS, B., BLOMER, J. Approximate matching of polygonal shapes. Ann Math Artif Intel. (1995); 13(3-4): 251-265.

[10] CARDONE, A., GUPTA, SK., DESHMUKH, A., KARNIK, M. 2006. Machining feature-based similarity assessment algorithms for prismatic machined parts. Comput Aided Design. (2006); 38(9): 954-972.

[11] ZHANG, D., HE, F. et al. 2017. 'An Efficient Approach to Directly Compute the Exact Hausdorff Distance for 3D Point Sets'. (1 Jan. 2017) : 261 – 277.

[12] D. HUTTENLOCHER, D., KILANDERMAN, S., AND RUCKLIDGE, W. 1993. Comparing Images Using the Hausdorff Distance. In IEEE Transactions on Pattern Analysis & Machine Intelligence, 15(9), pp. 850-863. DOI: 10.1109/34.232073

[13] AYDIN, O.U., TAHA, A.A., HILBERT, A. et al. 2021. On the usage of average Hausdorff distance for segmentation performance assessment: hidden error when used for ranking. Eur Radiol Exp 5, 4 (2021). https://doi.org/10.1186/s41747-020-00200-2

[14] ALT H, Behrends B, Blömer J. Approximate matching of polygonal shapes. Ann Math Artif Intel. 1995; 13(3-4): 251-265.

[15] CHEW, L., GOODRICH, M., HUTTENLOCHER, D., KEDEM, K., KLEINBERG, J., KRAVETS, D. 1997.Geometric pattern matching under Euclidean motion. Computational Geometry. 7(1-2), pp. 113-124. DOI: https://doi.org/10.1016/0925-7721(95)00047-X.

[16] KHARBACH, A., MERDANI, A., BELLACH, B., RAHMOUN, R., ELOMARI, M. 2017. An optimized algorithm for 2D images comparing based on Hausdorff Distance. Proceedings of the International Conference on Industrial Engineering and Operations Management Rabat, Morocco, (April 11-13, 2017), pp. 1329-1337.

[17] PAPADIAS, D., TAO, Y., MOURATIDIS, K., HUI, CK. Aggregate nearest neighbor queries in spatial databases. Acm T Database Syst. 2005; 30(2): 529-576.

[18] Dr. N. Kumar. 1997. Introduction to Horticulture. Rajalakshmi Publications, 28/5 – 693, Vepamoodu Junction, Nagercoil. Pp: 15.47- 15.50.

[19] Yeggoli, Narasimhulu & Venkaiah, Vadlamudi & Pasunuri, Raghunadh & Suthar, Ashok. (2021). CKD-Tree: An Improved KD-Tree Construction Algorithm. https://www.researchgate.net/publication/350089982_CKD-Tree_An_Improved_KD-Tree_Construction_Algorithm

[20] HINTON, G., HNIH, V. 2010. Learning to Detect Roads in High-Resolution Aerial Images. Learning to Detect Roads in High-Resolution Aerial Images. Computer Vision -- ECCV 2010. (2010). Pp. 210-223.

[21] ALSHAIKHLI, T., LUI, W., MARUYAMA, Y. 2019. Automated Method of Road Extraction from Aerial Images Using a Deep Convolutional Neural Network. Applied Sciences. (2019); 9(22):4825. https://doi.org/10.3390/app9224825

[22] Danilov, Dmitry & Lakhtin, Alexey. (2018). OPTIMIZATION OF THE ALGORITHM FOR DETERMINING THE HAUSDORFF DISTANCE FOR CONVEX POLYGONS. Ural mathematical journal. 4. 14-23. 10.15826/umj.2018.1.002.

[23] WANG, H., LI, G., PAN, J., SHANG, F. 2013. An Improved Algorithm of Improved Computation Efficiency on LTS Hausdorff Distance. Applied Mechanics and Materials (Volumes 347-350). (August 2013). Pp. 3217-3221. https://doi.org/10.4028/www.scientific.net/AMM.347-350.3217

[24] NIXON, M., AGUADO, A. 2020. High-level feature extraction: fixed shape matching. Feature Extraction and Image Processing for Computer Vision (Fourth Edition). (2020). Pp. 223-290. https://doi.org/10.1016/B978-0-12-814976-8.00005-1.

Appendix

Table 1

Orchard Number	Average Confidence Level	Standard Deviation	Confidence Used	Description	Shapes Detected
35516	0.2960	0.2142	0.154	Rectangular rotated rows	Rectangle
36507	0.8552	0.1497	0.7055	Rectangular rows	Rectangle
43581	0.5444	0.2698	0.2746	Rectangular rows	Rectangle

I