



(51) International Patent Classification:

G05D 1/00 (2006.01) *G01C 21/00* (2006.01)
G05D 3/00 (2006.01)

(21) International Application Number:

PCT/AU2014/000829

(22) International Filing Date:

22 August 2014 (22.08.2014)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

2013903183 22 August 2013 (22.08.2013) AU

(71) Applicant: THE UNIVERSITY OF SYDNEY [AU/AU];
Parramatta Road, Sydney, New South Wales 2006 (AU).(72) Inventors: UNDERWOOD, James Patrick; 9/111 McE-
voy Street, Alexandria, New South Wales 2015 (AU). NI-
ETO, Juan Ignacio; 1/16-17 New Beach Road, Darling
Point, New South Wales 2027 (AU). JAGBRANT, Gus-
tav Lars Henrik; Mariannelunds Gard 1, S-598 97 Marian-
nelund (SE). SUKKARIEH, Salah; 48 Bowns Road, Kog-
arah (AU).(74) Agent: SPRUSON & FERGUSON; GPO Box 3898,
Sydney, New South Wales 2001 (AU).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— with international search report (Art. 21(3))

(54) Title: LOCALISATION SYSTEM AND METHOD

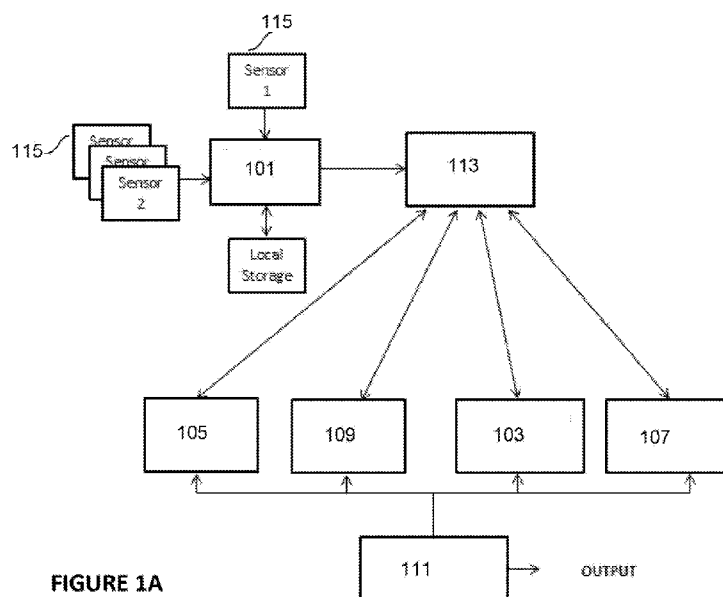


FIGURE 1A

(57) Abstract: A method or system of determining a location relative to the position of one or more natural elements within an environment, the method comprising the steps of: measuring a set of characteristics associated with the natural elements, wherein the set of characteristics comprises at least one characteristic associated with the natural elements; creating a plurality of discrete data sets from the measured set of characteristics; associating the discrete data sets with individual natural elements; sequencing data within the discrete data sets to create a current data sequence; and determining a location relative to the position of one or more natural elements within the environment based on a comparison of the current data sequence with a stored data sequence.

LOCALISATION SYSTEM AND METHOD

Technical Field

[0001] The present invention relates to a localisation system and method. In particular, the present invention relates to a method of determining a location relative to the position of one or more natural elements within an environment, a method of determining a set of characteristics associated with natural elements in an environment for determining a location relative to the position of a plurality of natural elements, and related systems.

Background

[0002] The population of the earth has been increasing and is expected to reach a staggering 9.6 billion by year 2050. Additionally, this increase in population is not evenly distributed, but is concentrated in the cities, leading to an unprecedented urbanisation of the world. As a direct consequence, a proportionally smaller rural population must sustain a much larger urban population. In order for such a change to be possible, it is necessary to increase the efficiency of the agricultural sector. To that end, systems are required that are capable of autonomous orchard surveillance, including mapping, classification and detection.

[0003] One key problem to any autonomous robotic system is a localisation problem of how a robot determines its position. One common approach in outdoors environments is to use Differential GPS (DGPS). In orchards however, this approach is generally only suitable for larger and taller robots, as the dense vegetation can attenuate the DGPS signal of smaller and shorter robots. In order to enable autonomous localisation for smaller robots, it is preferably to use a GPS independent localisation system.

[0004] The present invention aims to overcome, or at least alleviate, some or all of the afore-mentioned problems.

[0005] Further objects and advantages of the invention will be brought out in the following portions of the specification, wherein the detailed description is for the purpose of fully disclosing the preferred embodiment of the invention without placing limitations thereon.

[0006] The background discussion (including any potential prior art) is not to be taken as an admission of the common general knowledge in the art in any country. Any references discussed state the assertions of the author of those references and not the assertions of the applicant of this application. As such, the applicant reserves the right to challenge the accuracy and relevance of the references discussed.

Summary

[0007] Disclosed are arrangements which seek to address the above problems by determining a location relative to the position of one or more natural elements within an environment, or determining a set of characteristics associated with the natural elements in the environment for determining a location relative to their position.

[0008] According to a first or second aspect of the present disclosure, there is provided a method or system of determining a location relative to the position of one or more natural elements within an environment, the method comprising the steps of: measuring a set of characteristics associated with the natural elements, wherein the set of characteristics comprises at least one characteristic associated with the natural elements; creating a plurality of discrete data sets from the measured set of characteristics; associating the discrete data sets with individual natural elements; sequencing data within the discrete data sets to create a current data sequence; and determining a location relative to the position of one or more natural elements within the environment based on a comparison of the current data sequence with a stored data sequence.

[0009] The system may include a data capture module, a segmentation module, a sequencing module, a characterisation module and a localisation module adapted to perform the method.

[0010] The stored data sequence may be based on a previous characteristic data set created by measuring a second set of characteristics associated with the natural elements, the method further comprising the steps of associating data within the previous characteristic data set with the specific individual natural elements; and sequencing the associated data within the previous characteristic data set.

[0011] Data within the discrete data sets being associated with individual natural elements may be data associated with a first characteristic within the first set of characteristics. The data being sequenced to create the current data sequence may be data associated with a second characteristic within the first set of characteristics, and the first characteristic and second characteristic may be different.

[0012] Data within the discrete data sets being associated with individual natural elements may be data associated with a first characteristic within the first set of characteristics. The data being sequenced to create the current data sequence may be data associated with a second characteristic within the first set of characteristics, and the first characteristic and second characteristic may be the same characteristic.

[0013] The step of associating the discrete data sets with individual natural elements may segment the characteristic data set into discrete portions, and associate each discrete portion with one of the individual natural elements.

[0014] The step of segmenting the characteristic data set into discrete portions may segment the characteristic data set using a Hidden Semi-Markov Model.

[0015] The step of segmenting the characteristic data set into discrete portions may take discrete measurements of the first set of characteristics at predefined intervals.

[0016] The predefined intervals may be time intervals or distance intervals.

[0017] The step of sequencing data within the discrete data sets to create a current data sequence may group the discrete data sets into a defined sequence length.

[0018] The step of sequencing data within the discrete data sets may obtain a plurality of descriptor values, wherein each descriptor value is based on a plurality of single data points or a single data point, where each single data point is selected from one of a plurality of discrete data sets that are associated with one individual natural element, and position the obtained descriptor values in a defined sequence.

[0019] The descriptor value may be calculated from the single data points by selecting one of the single data points or calculating a new value based on the single data points.

[0020] The step of sequencing data within the discrete data sets may obtain a plurality of descriptor values, wherein each descriptor value is based on a plurality of data points, where each data point in the plurality of data points is selected from one of a plurality of discrete data sets that are associated with one individual natural element, and position the obtained descriptor values in a defined sequence.

[0021] The step of determining a location may use particle filters on the current data sequence.

[0022] The step of determining a location may use a HMM algorithm.

[0023] The HMM algorithm may use a Viterbi algorithm, Forwards algorithm or Forwards-Backwards algorithm.

[0024] The set of characteristics may be one or more of height, volume, density, colour, and temperature characteristics of natural elements in the environment.

[0025] Further measurements may be performed before, during or after determining a location, and those further measurements may be associated with a natural element based on the determined location.

[0026] A potential error may be determined based on the comparison of the current data sequence and stored data sequence. The potential error may be created from discrete data sets not being associated with the correct individual natural element.

[0027] According to a third or fourth aspect of the present disclosure, there is provided a method or system of determining a set of characteristics associated with natural elements in an environment for determining a location relative to the position of one or more natural elements within the environment. The method or system may measure a set of characteristics associated with the natural elements. The set of characteristics may have at least one characteristic associated with the natural elements. A plurality of discrete data sets may be created from the measured set of characteristics. The discrete data sets may be associated with individual natural elements. The data within the discrete data sets may be sequenced to create a data sequence. The data sequence may be stored in a manner suitable for determining a location within the environment.

[0028] Data within the discrete data sets being associated with individual natural elements may be data associated with a first characteristic within the first set of characteristics. The data being sequenced to create the current data sequence may be data associated with a second characteristic within the first set of characteristics. The first characteristic and second characteristic may be different.

[0029] Data within the discrete data sets being associated with individual natural elements may be data associated with a first characteristic within the first set of characteristics. The data being sequenced to create the current data sequence may be data associated with a second characteristic within the first set of characteristics. The first characteristic and second characteristic may be the same characteristic.

[0030] The step of associating the discrete data sets with individual natural elements may segment the characteristic data set into discrete portions, and associate each discrete portion with one of the individual natural elements.

[0031] The step of segmenting the characteristic data set into discrete portions may segment the characteristic data set using a Hidden Semi-Markov Model.

[0032] The step of segmenting the characteristic data set into discrete portions may take discrete measurements of the first set of characteristics at predefined intervals. The predefined intervals may be time intervals or distance intervals.

[0033] The step of sequencing data within the discrete data sets to create a current data sequence may group the discrete data sets into a defined sequence length.

[0034] The step of sequencing data within the discrete data sets may obtain a plurality of descriptor values. Each descriptor value may be based on a plurality of single data points, or a single data point. Each single data point may be selected from one of a plurality of discrete data sets that are associated with one individual natural element. The obtained descriptor values may be positioned in a defined sequence.

[0035] The descriptor value may be calculated from the single data points by selecting one of the single data points or calculating a new value based on the single data points.

[0036] The step of sequencing data within the discrete data sets may obtain a plurality of descriptor values. Each descriptor value may be based on a plurality of data points. Each data point in the plurality of data points may be selected from one of a plurality of discrete data sets that are associated with one individual natural element. The obtained descriptor values may be positioned in a defined sequence.

[0037] A location value may be determined based on a GPS signal. The location value may be stored with the current data sequence.

[0038] The system may include a data capture module, a segmentation module, a sequencing module, a characterisation module and a localisation module adapted to perform the method.

Brief Description of the Drawings

[0039] At least one embodiment of the present invention will now be described with reference to the drawings and appendices, in which:

[0040] Fig. 1A shows a schematic block diagram of a localisation system according to the present disclosure;

[0041] Figs. 1B and 1C form a schematic block diagram of a general purpose computer system upon which arrangements described can be practiced;

[0042] Fig. 2 shows a representation of a point cloud according to the present disclosure;

[0043] Fig. 3 shows a representation of height, density and volume values in data slices according to the present disclosure;

[0044] Fig. 4 shows a representation of height data values in data slices according to the present disclosure;

[0045] Fig. 5 shows a further representation of height data values in data slices according to the present disclosure;

[0046] Fig. 6 shows an image of equipment used to collect data in an orchard according to the present disclosure;

[0047] Fig. 7 shows a 3D point cloud according to the present disclosure;

[0048] Fig. 8 shows a state transition diagram of a Hidden Markov Model used in a method and system according to the present disclosure;

[0049] Fig. 9 shows a comparison of the implicit duration distribution of a HMM with the explicit duration distribution of a HSMM.

[0050] Fig. 10 shows a state transition diagram of a model of an orchard according to the present disclosure;

[0051] Figs. 11A and 11B show measured height data points over distance for orchard rows over 80 meters, and 80 meters to 160 meters according to the present disclosure;

[0052] Figs. 12A and 12B show measured density data points over distance for orchard rows over 80 meters, and 80 meters to 160 meters according to the present disclosure;

[0053] Figs. 13A and 13B show measured volume data points over distance for orchard rows over 80 meters, and 80 meters to 160 meters according to the present disclosure;

[0054] Fig. 14 shows measured volume differences over distance for an orchard row over 80 meters according to the present disclosure;

[0055] Fig. 15 shows measured difference of moving averages for an orchard row over 80 meters according to the present disclosure;

[0056] Fig. 16A and 16B show hand-tuned observation and border distributions according to the present disclosure;

[0057] Fig. 17 shows a state duration distribution for a tree and gap according to the present disclosure;

[0058] Fig. 18 shows a state duration distribution for a border according to the present disclosure;

[0059] Figs. 19A, 19B and 19C show histograms of volume measurements given the state of a tree, gap or border according to the present disclosure;

[0060] Fig. 20 shows learnt observation distributions according to the present disclosure;

[0061] Fig. 21 shows a histogram of the duration for a tree state according to the present disclosure;

[0062] Fig. 22A shows a data point image before pre-processing for ground removal according to the present disclosure;

[0063] Fig. 22B shows a data point image after pre-processing for ground removal according to the present disclosure;

[0064] Fig. 23 shows a comparison of volume measurements before and after removing the ground according to the present disclosure;

[0065] Fig. 24 shows hand tuned observation distributions according to the present disclosure;

[0066] Fig. 25 shows a state transition diagram according to the present disclosure;

[0067] Figs. 26A and 26B show hand tuned observation distributions according to the present disclosure;

[0068] Fig. 27 shows an image depicting a section of large trees according to the present disclosure;

[0069] Fig. 28 shows an image depicting a section with two small trees surrounded by large trees according to the present disclosure;

[0070] Fig. 29 shows an image depicting a section of a medium sized tree surrounded by large trees according to the present disclosure;

[0071] Fig. 30 shows an extract of the resulting segmentation using an ordinary Hidden Markov Model according to the present disclosure;

[0072] Fig. 31 shows an extract of volume measurements and state changes using an ordinary Hidden Markov Model according to the present disclosure;

[0073] Fig. 32 shows an extract of the resulting segmentation using 3 state Hidden Markov Model according to the present disclosure;

[0074] Fig. 33 shows an extract of the resulting segmentation when the gap between trees is labelled as part of the tress according to the present disclosure;

[0075] Fig. 34 shows a segmentation failure when the 4 state model is used with learnt volume and height observation distributions according to the present disclosure;

[0076] Fig. 35 shows an overview of the resulting segmentation using a 4 state model according to the present disclosure;

[0077] Fig. 36 shows an overview of the resulting segmentation using a 4 state model according to the present disclosure;

[0078] Fig. 37 shows two medium sized trees segmented using the 4 state model according to the present disclosure;

[0079] Fig. 38 shows two small trees segmented using the 4 state model according to the present disclosure;

[0080] Fig. 39 shows an empty portion before entering the row segmented using the 4 state model according to the present disclosure;

[0081] Fig. 40 shows a medium sized tree being labelled as a small tree according to the present disclosure;

[0082] Fig. 41 shows one of three border errors when segmenting the longer dataset using learnt volume distributions

[0083] Fig. 42 shows an excerpt of a data set collected by a robot moving in a sinusoidal motion according to the present disclosure;

[0084] Fig. 43 shows estimated volumes of tress with robot moving in a straight line according to the present disclosure;

[0085] Fig. 44 shows volume values calculated according to the present disclosure;

[0086] Fig. 45 shows height values calculated according to the present disclosure;

[0087] Fig. 46 shows volume signature values calculated according to the present disclosure;

[0088] Fig. 47 shows height signature values calculated according to the present disclosure;

[0089] Fig. 48 shows a descriptor difference distribution according to the present disclosure;

[0090] Fig. 49 shows a descriptor difference distribution according to the present disclosure;

[0091] Fig. 50 shows a cumulative observation distribution function according to the present disclosure;

[0092] Fig. 51 shows a histogram of tree differences according to the present disclosure;

[0093] Fig. 52 shows a histogram of tree duration differences according to the present disclosure;

[0094] Fig. 53 shows a histogram of differences between real measurements of the same tree according to the present disclosure;

[0095] Fig. 54 shows a histogram of differences between simulated measurements of the same tree according to the present disclosure;

[0096] Fig. 55 shows a ratio of correct matches and segmentation errors according to the present disclosure;

[0097] Fig. 56 shows a ratio of correct matches and segmentation errors according to the present disclosure;

[0098] Fig. 57 shows a ratio of correct matches and segmentation errors according to the present disclosure;

[0099] Fig. 58 shows a ratio of correct matches and segmentation errors according to the present disclosure;

[00100] Fig. 59 shows a ratio of correct matches and segmentation errors according to the present disclosure;

[00101] Fig. 60 shows a ratio of correct matches and segmentation errors according to the present disclosure;

[00102] Fig. 61 shows a ratio of correct matches and segmentation errors according to the present disclosure;

[00103] Fig. 62 shows a ratio of correct matches and segmentation errors according to the present disclosure;

[00104] Fig. 63 shows a ratio of correct matches and segmentation errors according to the present disclosure;

[00105] Fig. 64 shows a ratio of correct matches and segmentation errors according to the present disclosure;

[00106] Fig. 65 shows a ratio of correct matches and segmentation errors according to the present disclosure;

[00107] Fig. 66 shows a ratio of correct matches and segmentation errors according to the present disclosure;

[00108] Fig. 67 shows a self-similarity distribution of a volume descriptor in the form of a histogram of differences between measurements of the same tree according to the present disclosure;

[00109] Fig. 68 shows a self-similarity distribution of a height descriptor in the form a histogram of differences between measurements of the same tree according to the present disclosure;

[00110] Fig. 69 shows a self-similarity distribution of a volume signature descriptor in the form a histogram of differences between measurements of the same tree according to the present disclosure;

[00111] Fig. 70 shows a self-similarity distribution of a height signature descriptor in the form a histogram of differences between measurements of the same tree according to the present disclosure;

[00112] Fig. 71 shows a similarity distribution to other trees for a volume descriptor in the form of a histogram of differences between measurements of different trees according to the present disclosure;

[00113] Fig. 72 shows a similarity distribution to other trees for a height descriptor in the form of a histogram of differences between measurements of different trees according to the present disclosure;

[00114] Fig. 73 shows a similarity distribution to other trees for a volume signature descriptor in the form of a histogram of differences between measurements of different trees according to the present disclosure;

[00115] Fig. 74 shows a similarity distribution to other trees for a height signature descriptor in the form of a histogram of differences between measurements of different trees according to the present disclosure;

Description of Embodiments

[00116] Embodiments of the present invention are described herein with reference to a system adapted or arranged to perform the methods described.

[00117] Autonomous location methods and systems are described herein.

[00118] A first task examines how to segment a 3D point cloud of an orchard, collected by a 2D Laser Scanner, into trees; allowing a surveying robot to designate estimated attributes such as health and yield to individual trees.

[00119] A second task examines how the geometric characteristics of the individual trees can be described in order to enable localisation.

[00120] A third task presents a robust localisation method, based on the previously introduced segmentation and characterisation methods. Finally, a method and system for detecting segmentation errors when performing localisation is described, allowing erroneous attribute estimates to be discarded.

[00121] The presented evaluation is based on 3D point data created from DGPS location estimates of the robot's position. Similar results can also be obtained by using odometry data from the robot or any other suitable method.

[00122] Alternatively, analysis for orchards where trees are in a trellis or wall like structure is also envisaged. In this example, characteristic measurements can be taken such as "linearity" of 3D points, which corresponds to trunks of the trees, and colour characteristics obtained from cameras, as described herein. That is, where the captured data points are arranged in a linear manner, the system may interpret this as representing a trunk like structure.

[00123] It is acknowledged that the terms "comprise", "comprises" and "comprising" may, under varying jurisdictions, be attributed with either an exclusive or an inclusive meaning. For the purpose of this specification, and unless otherwise noted, these terms are intended to have an inclusive meaning - i.e. they will be taken to mean an inclusion of the listed components that the use directly references, but optionally also the inclusion of other non-specified components or elements. It will be understood that this intended meaning also similarly applies to the terms mentioned when used to define steps in a method or process.

[00124] It will be understood that, when describing various integers, such as modules, components, elements etc., any integer may be constituted by a single integer or multiple integers.

[00125] It will be understood that the embodiments of the present invention described herein are by way of example only, and that various changes and modifications may be made without departing from the scope of invention.

[00126] In summary, the system may include at least a processor, one or more memory devices or an interface for connection to one or more memory devices, input and output interfaces for connection to external devices or sensors in order to enable the system to receive and operate upon instructions from one or more users or external systems, a data bus for internal and external communications between the various components, and a suitable power supply. Further, the system may include one or more communication devices (wired or wireless) for communicating with external and internal devices, and one or more input/output devices, such as a display, pointing device, keyboard or printing device.

[00127] The processor or various modules described are arranged to perform the steps of a program stored as program instructions within a memory device. The program instructions enable the various methods of performing the invention as described herein to be performed. The program instructions may be developed or implemented using any suitable software programming language and toolkit, such as, for example, a C-based language. Further, the program instructions may be stored in any suitable manner such that they can be transferred to the memory device or read by the processor or modules, such as, for example, being stored on a computer readable medium. The computer readable medium may be any suitable medium, such as, for example, solid state memory, magnetic tape, a compact disc (CD-ROM or CD-R/W), memory card, flash memory, optical disc, magnetic disc or any other suitable computer readable medium.

[00128] It will be understood that the system herein described includes one or more elements or modules that are arranged to perform the various functions and methods. The following portion of the description

is aimed at providing the reader with an example of a conceptual view of how various modules and/or engines that make up the elements of the system may be interconnected to enable the functions to be implemented. Further, the following portion of the description explains in system related detail how the steps of the herein described method may be performed. The conceptual diagrams are provided to indicate to the reader how the various data elements are processed at different stages by the various different modules and/or engines.

[00129] It will be understood that the modules and/or engines described may be implemented and provided with instructions using any suitable form of technology. For example, the modules or engines may be implemented or created using any suitable software code written in any suitable language, where the code is then compiled to produce an executable program that may be run on any suitable computing system. Alternatively, or in conjunction with the executable program, the modules or engines may be implemented using any suitable mixture of hardware, firmware and software. For example, portions of the modules may be implemented using an application specific integrated circuit (ASIC), a system-on-a-chip (SoC), field programmable gate arrays (FPGA) or any other suitable adaptable or programmable processing device.

[00130] According to this embodiment, data is captured in a natural environment by a robotic system. However, it will be understood that as an alternative, embodiments may use a self-contained non-mobile system that is designed to be attached to any existing vehicle, such as a tractor or quad bike for example. Further, as another alternative, the data capture system could be moved or towed by hand.

[00131] As shown in figure 1A, the system includes a data capture module 101, a characterisation module 103, a segmentation module 105, a sequencing module 107, a localisation module 109, and a control module 111 adapted to control the operation of the characterisation module, segmentation module, sequencing module and localisation module. The system also includes a data store 113, database or data storage module arranged to store data and enable data to be retrieved therefrom. Further, any of the modules may include their own local data storage facility to enable data to be stored and retrieved locally.

[00132] Multiple sensors 115 are also part of the system and may include one or more of a laser scanning and detection system, a GPS device, an odometry device, a colour or thermal camera device or any other sensing device that may be used to detect characteristics of natural elements.

[00133] Further, other sensing devices may be used such as a soil measurement device, air quality measurement device, water quality measurement device or other suitable measurement device for measuring characteristics of the environment around the natural elements in the environment.

[00134] In general terms, the data capture module is suitable for and may be specifically adapted for capturing data associated with features of natural elements, as well as related functions. The characterisation module is suitable for and may be specifically adapted for retrieving data captured by the data capture module from sensors to determine characteristics associated with the natural elements, as

well as related functions. The segmentation module is suitable for and may be specifically adapted for creating discrete data sets and organising the data into segments that correspond with individual natural elements, as well as related functions. The sequencing module is suitable for and may be specifically adapted for sequencing the data, as well as related functions. The localisation module is suitable for and may be specifically adapted for determining a location, as well as related functions. The control module is suitable for and may be specifically adapted for controlling each of the modules and also for generating an output.

[00135] It will be understood that the arrangement and construction of the modules or engines may be adapted accordingly depending on system and user requirements so that various functions may be performed by different modules or engines to those described herein, and that certain modules or engines may be combined into single modules or engines or the functions of the herein described modules or engines may be separated out into different modules or engines.

[00136] Figs. 1B and 1C depict a general-purpose computer system 1300, upon which the various arrangements described can be practiced.

[00137] As seen in Fig. 1B, the computer system 1300 includes: a computer module 1301; input devices such as a keyboard 1302, a mouse pointer device 1303, a scanner 1326, a camera 1327, and a microphone 1380; and output devices including a printer 1315, a display device 1314 and loudspeakers 1317. An external Modulator-Demodulator (Modem) transceiver device 1316 may be used by the computer module 1301 for communicating to and from a communications network 1320 via a connection 1321. The communications network 1320 may be a wide-area network (WAN), such as the Internet, a cellular telecommunications network, or a private WAN. Where the connection 1321 is a telephone line, the modem 1316 may be a traditional "dial-up" modem. Alternatively, where the connection 1321 is a high capacity (e.g., cable) connection, the modem 1316 may be a broadband modem. A wireless modem may also be used for wireless connection to the communications network 1320.

[00138] The computer module 1301 typically includes at least one processor unit 1305, and a memory unit 1306. For example, the memory unit 1306 may have semiconductor random access memory (RAM) and semiconductor read only memory (ROM). The computer module 1301 also includes an number of input/output (I/O) interfaces including: an audio-video interface 1307 that couples to the video display 1314, loudspeakers 1317 and microphone 1380; an I/O interface 1313 that couples to the keyboard 1302, mouse 1303, scanner 1326, camera 1327 and optionally a joystick or other human interface device (not illustrated); and an interface 1308 for the external modem 1316 and printer 1315, as well as sensors as described herein. In some implementations, the modem 1316 may be incorporated within the computer module 1301, for example within the interface 1308. The computer module 1301 also has a local network interface 1311, which permits coupling of the computer system 1300 via a connection 1323 to a local-area communications network 1322, known as a Local Area Network (LAN). As illustrated in Fig. 1B, the local communications network 1322 may also couple to the wide network 1320 via a connection 1324, which would typically include a so-called "firewall" device or device

of similar functionality. The local network interface 1311 may comprise an Ethernet circuit card, a Bluetooth® wireless arrangement or an IEEE 802.11 wireless arrangement; however, numerous other types of interfaces may be practiced for the interface 1311.

[00139] The I/O interfaces 1308 and 1313 may afford either or both of serial and parallel connectivity, the former typically being implemented according to the Universal Serial Bus (USB) standards and having corresponding USB connectors (not illustrated). Storage devices 1309 are provided and typically include a hard disk drive (HDD) 1310. Other storage devices such as a floppy disk drive and a magnetic tape drive (not illustrated) may also be used. An optical disk drive 1312 is typically provided to act as a non-volatile source of data. Portable memory devices, such optical disks (e.g., CD-ROM, DVD, Blu-ray Disc™), USB-RAM, portable, external hard drives, and floppy disks, for example, may be used as appropriate sources of data to the system 1300.

[00140] The components 1305 to 1313 of the computer module 1301 typically communicate via an interconnected bus 1304 and in a manner that results in a conventional mode of operation of the computer system 1300 known to those in the relevant art. For example, the processor 1305 is coupled to the system bus 1304 using a connection 1318. Likewise, the memory 1306 and optical disk drive 1312 are coupled to the system bus 1304 by connections 1319. Examples of computers on which the described arrangements can be practised include IBM-PC's and compatibles, Sun Sparcstations, Apple Mac™ or a like computer systems.

[00141] Various methods as described herein may be implemented using the computer system 1300 wherein the processes to be described, may be implemented as one or more software application programs 1333 executable within the computer system 1300 as modules. In particular, the steps of the method of determining a location relative to the position of one or more natural elements within the environment or the method of determining a set of characteristics associated with natural elements in an environment for determining a location relative to the position of one or more natural elements within the environment may be effected by instructions 1331 (see Fig. 1C) in the software 1333 that are carried out within the computer system 1300. The software instructions 1331 may be formed as one or more code modules, each for performing one or more particular tasks. The software may also be divided into two separate parts, in which a first part and the corresponding code modules performs the herein described methods and a second part and the corresponding code modules manage a user interface between the first part and the user.

[00142] The software may be stored in a computer readable medium, including the storage devices described below, for example. The software is loaded into the computer system 1300 from the computer readable medium, and then executed by the computer system 1300. A computer readable medium having such software or computer program recorded on the computer readable medium is a computer program product. The use of the computer program product in the computer system 1300 preferably effects an advantageous apparatus for determining a location relative to the position of one or more natural elements within the environment or an apparatus for determining a set of characteristics

associated with natural elements in an environment for determining a location relative to the position of one or more natural elements within the environment.

[00143] The software 1333 is typically stored in the HDD 1310 or the memory 1306. The software is loaded into the computer system 1300 from a computer readable medium, and executed by the computer system 1300. Thus, for example, the software 1333 may be stored on an optically readable disk storage medium (e.g., CD-ROM) 1325 that is read by the optical disk drive 1312. A computer readable medium having such software or computer program recorded on it is a computer program product. The use of the computer program product in the computer system 1300 preferably effects an apparatus for determining a location relative to the position of one or more natural elements within the environment or an apparatus for determining a set of characteristics associated with natural elements in an environment for determining a location relative to the position of one or more natural elements within the environment.

[00144] In some instances, the application programs 1333 may be supplied to the user encoded on one or more CD-ROMs 1325 and read via the corresponding drive 1312, or alternatively may be read by the user from the networks 1320 or 1322. Still further, the software can also be loaded into the computer system 1300 from other computer readable media. Computer readable storage media refers to any non-transitory tangible storage medium that provides recorded instructions and/or data to the computer system 1300 for execution and/or processing. Examples of such storage media include floppy disks, magnetic tape, CD-ROM, DVD, Blu-ray™ Disc, a hard disk drive, a ROM or integrated circuit, USB memory, a magneto-optical disk, or a computer readable card such as a PCMCIA card and the like, whether or not such devices are internal or external of the computer module 1301. Examples of transitory or non-tangible computer readable transmission media that may also participate in the provision of software, application programs, instructions and/or data to the computer module 1301 include radio or infra-red transmission channels as well as a network connection to another computer or networked device, and the Internet or Intranets including e-mail transmissions and information recorded on Websites and the like.

[00145] The second part of the application programs 1333 and the corresponding code modules mentioned above may be executed to implement one or more graphical user interfaces (GUIs) to be rendered or otherwise represented upon the display 1314. Through manipulation of typically the keyboard 1302 and the mouse 1303, a user of the computer system 1300 and the application may manipulate the interface in a functionally adaptable manner to provide controlling commands and/or input to the applications associated with the GUI(s). Other forms of functionally adaptable user interfaces may also be implemented, such as an audio interface utilizing speech prompts output via the loudspeakers 1317 and user voice commands input via the microphone 1380.

[00146] Fig. 1C is a detailed schematic block diagram of the processor 1305 and a "memory" 1334. The memory 1334 represents a logical aggregation of all the memory modules (including the HDD 1309 and semiconductor memory 1306) that can be accessed by the computer module 1301 in Fig. 1B.

[00147] When the computer module 1301 is initially powered up, a power-on self-test (POST) program 1350 executes. The POST program 1350 is typically stored in a ROM 1349 of the semiconductor memory 1306 of Fig. 1B. A hardware device such as the ROM 1349 storing software is sometimes referred to as firmware. The POST program 1350 examines hardware within the computer module 1301 to ensure proper functioning and typically checks the processor 1305, the memory 1334 (1309, 1306), and a basic input-output systems software (BIOS) module 1351, also typically stored in the ROM 1349, for correct operation. Once the POST program 1350 has run successfully, the BIOS 1351 activates the hard disk drive 1310 of Fig. 1B. Activation of the hard disk drive 1310 causes a bootstrap loader program 1352 that is resident on the hard disk drive 1310 to execute via the processor 1305. This loads an operating system 1353 into the RAM memory 1306, upon which the operating system 1353 commences operation. The operating system 1353 is a system level application, executable by the processor 1305, to fulfil various high level functions, including processor management, memory management, device management, storage management, software application interface, and generic user interface.

[00148] The operating system 1353 manages the memory 1334 (1309, 1306) to ensure that each process or application running on the computer module 1301 has sufficient memory in which to execute without colliding with memory allocated to another process. Furthermore, the different types of memory available in the system 1300 of Fig. 1B must be used properly so that each process can run effectively. Accordingly, the aggregated memory 1334 is not intended to illustrate how particular segments of memory are allocated (unless otherwise stated), but rather to provide a general view of the memory accessible by the computer system 1300 and how such is used.

[00149] As shown in Fig. 1C, the processor 1305 includes a number of functional modules including a control unit 1339, an arithmetic logic unit (ALU) 1340, and a local or internal memory 1348, sometimes called a cache memory. The cache memory 1348 typically includes a number of storage registers 1344 - 1346 in a register section. One or more internal busses 1341 functionally interconnect these functional modules. The processor 1305 typically also has one or more interfaces 1342 for communicating with external devices via the system bus 1304, using a connection 1318. The memory 1334 is coupled to the bus 1304 using a connection 1319.

[00150] The application program 1333 includes a sequence of instructions 1331 that may include conditional branch and loop instructions. The program 1333 may also include data 1332 which is used in execution of the program 1333. The instructions 1331 and the data 1332 are stored in memory locations 1328, 1329, 1330 and 1335, 1336, 1337, respectively. Depending upon the relative size of the instructions 1331 and the memory locations 1328-1330, a particular instruction may be stored in a single memory location as depicted by the instruction shown in the memory location 1330. Alternately, an instruction may be segmented into a number of parts each of which is stored in a separate memory location, as depicted by the instruction segments shown in the memory locations 1328 and 1329.

[00151] In general, the processor 1305 is given a set of instructions which are executed therein. The processor 1105 waits for a subsequent input, to which the processor 1305 reacts to by executing another

set of instructions. Each input may be provided from one or more of a number of sources, including data generated by one or more of the input devices 1302, 1303, data received from an external source across one of the networks 1320, 1302, data retrieved from one of the storage devices 1306, 1309 or data retrieved from a storage medium 1325 inserted into the corresponding reader 1312, all depicted in Fig. 1B. The execution of a set of the instructions may in some cases result in output of data. Execution may also involve storing data or variables to the memory 1334.

[00152] The disclosed localisation arrangements use input variables 1354, which are stored in the memory 1334 in corresponding memory locations 1355, 1356, 1357. The localisation arrangements produce output variables 1361, which are stored in the memory 1334 in corresponding memory locations 1362, 1363, 1364. Intermediate variables 1358 may be stored in memory locations 1359, 1360, 1366 and 1367.

[00153] Referring to the processor 1305 of Fig. 1C, the registers 1344, 1345, 1346, the arithmetic logic unit (ALU) 1340, and the control unit 1339 work together to perform sequences of micro-operations needed to perform "fetch, decode, and execute" cycles for every instruction in the instruction set making up the program 1333. Each fetch, decode, and execute cycle comprises:

[00154] a fetch operation, which fetches or reads an instruction 1331 from a memory location 1328, 1329, 1330;

[00155] a decode operation in which the control unit 1339 determines which instruction has been fetched; and

[00156] an execute operation in which the control unit 1339 and/or the ALU 1340 execute the instruction.

[00157] Thereafter, a further fetch, decode, and execute cycle for the next instruction may be executed. Similarly, a store cycle may be performed by which the control unit 1339 stores or writes a value to a memory location 1332.

[00158] Each step or sub-process in the processes as described herein is associated with one or more segments of the program 1333 and is performed by the register section 1344, 1345, 1347, the ALU 1340, and the control unit 1339 in the processor 1305 working together to perform the fetch, decode, and execute cycles for every instruction in the instruction set for the noted segments of the program 1333.

[00159] The methods of localisation and characteristics determination may alternatively be implemented in dedicated hardware such as one or more integrated circuits performing the functions or sub functions of the modules as described. Such dedicated hardware may include graphic processors, digital signal processors, or one or more microprocessors and associated memories.

[00160]

[00161] The following describes a method and associated system for determining a location relative to the position of one or more natural elements within an environment. The natural elements may be, for

example, a tree, bush, plant etc. The natural elements may be arranged individually and/or separately in the natural environment, or may be arranged such that they combine to form a trellis or hedge like structure. The natural environment may be a farm, orchard, field, nursery or other type of natural environment in which natural elements are grown, nurtured, farmed etc.

[00162] In order to perform localisation, an initial data set may be captured to provide a benchmark against which a current data set can be compared. Bootstrapping may provide a specific reference point so the data can be associated with one or more geo-referenced points, for example, within the natural environment.

[00163] A current data set may be captured using the same reference point(s) as that used during the capture of the initial data set. The current data set may be compared to the initial data set to find a match, which enables a location to be determined based on the reference point.

[00164] When capturing the initial data set and the current data set, the following three steps may be performed.

Step 1 - Data Capture

[00165] Feature data of natural elements in an environment is captured by the data capture module at discrete intervals and transformed into discrete data sets [e.g. a slice of data]. This data is stored in the database. For example, the feature data may include characteristics such as height, volume or density measurements (or any combination thereof). Other features or characteristics may also be captured, such as colour and thermal imaging for example.

Step 2 - Segmentation

[00166] The feature or characteristic data within the discrete data sets is analysed to segment the discrete data sets. This segmentation step is performed by the segmentation module and identifies which of the discrete data sets are associated with one or more natural elements in the environment and which of the discrete data sets are associated with gaps or boundaries within the environment.

[00167] The segmentation of the discrete data sets requires the analysis of the feature data within each of the discrete data sets to identify whether the feature data relates to a natural element. An example of a suitable method for performing segmentation is based on the Hidden Semi-Markov Model (HSMM). Feature data determined from individual discrete data sets may be placed into the HSMM model to perform segmentation. Optionally, the feature data could be hand tuned to reduce errors. As a further option, a state duration distribution (such as a Gaussian distribution) could be applied to the HSMM model in order to define the expected spatial distances within the natural environment.

[00168] After segmentation has been performed, one or more discrete data sets may be associated with a particular natural element.

Step 3 - Characterisation

[00169] Each natural element is then characterised by the characterisation module using the feature data within the segmented discrete data sets. The characterisation could be by way of a basic descriptor or a more complex signature descriptor.

[00170] A basic descriptor uses a single value (data point) associated with a particular feature of the natural element.

[00171] For example, a height value for natural element n may be determined from one or more discrete data sets that have been associated with n via the segmentation process. The maximum height value in each of three discrete data sets associated with a specific tree may be averaged to provide an indicative height value for that tree. Other methods may be used to determine the indicative height value.

[00172] A signature descriptor uses multiple values within a plurality of discrete data sets that are associated with a specific natural element. For example, a height signature for a natural element m may be determined from multiple discrete data sets that have been associated with m via the segmentation process. The height signature may consist of the maximum height value taken from each of the associated discrete data sets for m. This may result in a signature descriptor consisting of a series of numbers associated with a particular natural feature of the natural element, where the series of numbers are taken from a plurality of discrete data sets associated with that natural element.

[00173] The natural feature being measured for the purposes of characterisation may be the same or different to the natural feature being measured for the purposes of segmentation.

Localisation

[00174] After characterising the segmented captured data, localisation may be performed by the localisation module. Localisation may include one or more of the following tasks. Determining that the current data set being captured is associated with a data set that was captured prior to the current data set. Determining that a current discrete data set being captured is associated with a discrete data set that was captured prior to the current discrete data set. Determining that equipment being used to capture the current data set is located at a pre-defined location based on earlier captured data. Determining that a data set (whether the whole data set, a part or portion of the whole data set, or a discrete data set within the data set) within a stored medium matches a further data set (whether the whole further data set, a part or portion of the whole further data set, or a discrete data set within the further data set) within the same or different stored medium, where the data set and further data set were captured in the same environment at different times. In each of these localisation tasks, a non geo-reference or a geo-reference may be provided as a location reference point.

[00175] Localisation is performed by finding an association between two sets of data, e.g. by finding a match between a first sequence of basic descriptors in a first data set (or first distinct data set) and a second sequence of basic descriptors in a second data set (or second distinct data set). The sequencing module groups the data sets into a defined sequence length. Alternatively, as a further example, a match

may be found between a first sequence of signature descriptors in a first data set (or first distinct data set) and a second sequence of signature descriptors in a second data set (or second distinct data set).

[00176] One example method for matching signature descriptors is the use of a Hidden Markov Model (HMM). Other sequence matching methods are also envisaged, such as the use of a particle filter for example.

Particle Filters

[00177] Particle filters could be used in a way that combines aspects of the basic localiser method and the localisation method using HMM.

[00178] A database is initially produced in the same way as already described.

[00179] It is assumed that the location is unknown. Evenly weighted (equally likely) hypotheses (particles) randomly distributed are placed all over the environment (e.g. farm).

[00180] A current tree measurement is taken (either a discrete slice or a segmented tree), and a similarity measurement is taken using the data in the database.

[00181] The first most likely location is not necessarily considered the best match.

[00182] All potential possibilities are gathered from the entire database, weighted by a similarity score where more similar equals a higher weight.

[00183] The similarity weight is multiplied by the particle (hypotheses likelihood) weight.

[00184] In a new iteration, a prediction is made whether the movement is a little forwards, or backwards, or stayed still (but not jumped four rows to the left for example). This prediction is equivalent to the "transition matrix" as described herein. In this case, all the particles are moved to possible new locations, slightly forwards, backwards or in the same spot, because the hypotheses contain the possibility of any valid motion.

[00185] This process is repeated predicting where particles (hypotheses) may go, and updating those based on the sensing/similarity to database.

[00186] Over a few iterations, the particles (hypothesis) with the largest weight will be in the true location.

[00187] Further, renormalisation and deleting of unlikely particles may be applied to maintain efficiency and tractability on finite machines, as well as accuracy.

Bootstrapping

[00188] An initial reference point is needed if it is a requirement that the result provides an indication of a location within the natural environment. E.g. an indication of whereabouts the current data set was captured within an entire orchard, an area of an orchard, a particular tree within a particular row, etc.

[00189] However, an initial reference point is not strictly needed if the only result required is to identify when the current data set matches a previous data set to identify a specific tree. For example, if a specific tree needs to be located for some purpose and that tree was identified in the initial data set, then an initial reference point is not required. The specific tree would be identified when the current data set is matched with the initial data set.

[00190] The initial reference point is not required to be a geo-reference point. However, a geo-reference point may be used initially if it is required to associate the captured data with a specific geo-reference to provide location information. The geo-reference point may be captured by a GPS device or may be provided manually by using map references.

[00191] An example of a reference point that is not a geo-reference point may be a row and tree number identifying where the start of the capture of the initial data set occurred.

Specific example using LADAR

[00192] The following portion of the description provides a specific example of data capture using laser scanning sensors.

[00193] Using a laser measurement system (such as a SICK LMS 291), data is captured at 75 frames per second in a 180 degree vertical window. A first scan captures data at 0, 1, 2, 3, through to 179, 180 degrees at 1 degree intervals. A second scan captures data at 0.25, 1.25, 2.25, 3.25 through to 179.25, 180.25 degrees at 1 degree intervals. A third scan captures data at 0.5, 1.5, 2.5, 3.5 through to 179.5, 180.5 degrees at 1 degree intervals. A fourth scan captures data at 0.75, 1.75, 2.75, 3.75 through to 179.75, 180.75 degrees at 1 degree intervals. The scanned data is then interleaved to produce a resolution of approximately 0.25 degrees.

[00194] The scanned data produces a point cloud, which is a set of continuous valued points that may be located anywhere. To start providing a reference point for the points within the point cloud, a determination of the position of the laser measurement system relative to points is required. An odometer located on the wheels of the vehicle holding the laser measurement system may be used to determine how far the laser measurement system is moving relative to the target natural elements. It will be understood that other methods of determining distance relative to the target natural elements may be used.

[00195] All point cloud data captured within a defined distance based on the odometer reading is then combined by projecting the scanned points into a common reference frame. For example, the defined distance may be 0.2 metres. Fig. 2 shows an example of the distribution of data in a point cloud.

[00196] This common reference frame is then discretised by voxelising the data therein to produce 0.2 metre slices perpendicular (or at least substantially perpendicular) to the direction of travel of the laser measurement system. By taking these series of vertical measurements and combining them into 0.2 metre slices in the horizontal plane, a 3D point cloud of data is produced containing 0.2 metre³ voxels. That is, a series of adjacent vertical slices of data points is produced where each slice is based on data

collected over a 0.2 metre range. It will be understood that the resolution of data capture may be adjusted up or down.

[00197] Each data point (0) is referenced to an X value, a Y value and a Z value to identify its location in the 3D point cloud.

[00198] The X value is dependent on the 0.2 metre slice it is a part of and is based on the position of the laser measurement system relative to the target natural element along the horizontal X-axis. According to this example, the X value is calculated using the odometer attached to the vehicle upon which the sensor is mounted.

[00199] The Y value (height) is calculated from the degree position (or elevation) within the 180 degree scan and the range detected by the sensor as measured by the laser measurement system.

[00200] The Z value (depth) also based on the degree position (or elevation) within the 180 degree scan and the range detected by the sensor.

[00201] In this example, the X, Y, Z values are captured assuming that the robot moves perfectly in line with the row, while positioned so it is looking at exactly 90 degrees to the trees. It is also assumed that there is no pitch in the vehicle motion. However, it will be understood that, if the robot is not constrained to move perfectly in line with the row (e.g. it can have yaw motion / changes in heading) and that the vehicle may also pitch with undulating ground, then each of X, Y, Z values may be a function of vehicle pose (north, east, down, roll, pitch, yaw) and sensor (range, bearing).

[00202] That is, the sensor returns polar co-ordinates in the form of a range and elevation value where light bounces off an object. The polar co-ordinates are converted to Cartesian co-ordinates to produce an (x, y) value for the sensor, where the x value is the depth value (Z) and the y value is the height value (Y).

[00203] The (x, y) sensor values are transformed or converted into a common frame using the X, Y, Z co-ordinate system with spatial units in metres. A grid of 0.2 metre cubes is used so that all the original free form points in the raw point cloud that fall into any one individual cube are combined by averaging their Cartesian location (X, Y, Z) to produce the location value associated with that cube. Therefore, the captured data is down-sampled while still maintaining a suitable sharpness or resolution in the data, and a more uniform density of points. That is, each cube contains one averaged point location or none.

[00204] Each slice of data points is considered a discrete data set, thus the system produces multiple discrete data sets.

How LIDAR measurements relate to height, volume, density

Height

[00205] The laser returns range/elevation values where the light bounces off an object, such as a tree. The laser returns a "no-return" or "max-range" indication for non-returns where light has not bounced off

an object. These range/elevation values and "max-range" indications are interpreted in software accordingly.

[00206] For all valid range/elevation pairs, a conversion is performed from polar co-ordinates (range, elevation) to Cartesian co-ordinates (x, y in sensor frame) as discussed above. The results are referenced into the common frame using X, Y, Z co-ordinates as discussed above.

[00207] The highest Y value in all of the cells that had a data point (i.e. did not include a "max-range" indication) within them is taken to be the maximum height value. It will be understood that other variations are envisaged for determining the height value, including, for example, only counting 0.2 metre cubes that have > N raw points (from the point cloud before creating the voxels), ignoring the top M% of points because it is assumed that there may be noise, etc.

[00208] Referring to figure 3, it can be seen that the highest Y value in which a value was returned was in the voxel having co-ordinates (0, 0.6, 0.4). Therefore, the maximum height value is calculated from the average Y value of all the points associated with that voxel. It will be understood that alternative ways of determining a maximum height value may be used. For example, the maximum non-averaged Y value from all points in the voxel may be used.

Density

[00209] By counting the number of raw points within a slice, an estimate of the density in that slice can be obtained. That is, a point value is not returned if there is a gap in the canopy or between trees and so point value counts approximately equate to a density measurement.

[00210] Alternatively, density may be calculated from the volume measurement measured using the method below. In this case, density may be calculated by dividing the volume value calculated below by the total volume of all cubes in the slice.

[00211] Referring to figure 3, it can be seen that four voxels in the slice of data returned values (i.e. did not return a max-range indication). Therefore, the density value is given a value of 4 points. As an alternative, the density value may be calculated by counting the total number of points within the 4 voxels.

Volume

[00212] In each 0.2 metre slice (X value), sections of 0.2 metre by 0.2 metre (Y value = height, and Z value = depth measurement) were created to produce 0.008m³ voxels. By taking a count of each 0.008m³ voxel (i.e. 0.2 metre³ blocks) that had at least one point detected therein, the approximate volume was calculated.

[00213] E.g. if 100 blocks were detected with at least one range/elevation pair returned from the sensor, the volume would be calculated by $100 \times 0.008 = 0.8$ m³ volume for that particular 0.2 metre slice.

[00214] Referring to figure 3, as there are 4 voxels with points measured, the volume value returned for this slice is $4 \times 0.008 = 0.032$ m³ volume for the slice.

[00215] It will be understood that any combination of these different values (height, density, volume) could be used. Also, other characteristic measurements such as colour, thermal characteristics etc. could be used.

Segmentation

[00216] Using any of the features above, segmentation is performed by the segmentation module on the data set using HSMM to determine where the data points specifically relate to trees, gaps or borders. This information is then used later so that only tree related data is used to calculate height, volume or density etc.

[00217] Referring to Fig. 4, it can be seen that various data slices maybe allocated as a TREE1, GAP or TREE2 depending on the height data value in those slices. For example, because the height data includes in slice F and slice G are below a predetermined value, the system determines that those data points relate to a gap between the natural elements.

[00218] Using basic or signature descriptors, each natural element within the environment can be characterised by the characterisation module.

Basic Descriptors

[00219] A single value for each tree may be used in the data set to compare to the previous baseline data set. These single values are obtained using the segmentation data. That is, a number of slices related to a single tree are analysed to determine the descriptor value.

Height Example

[00220] 5 slices of data identified as a tree from the segmentation step are analysed to determine the height of the tree. For example, Slices A to E identified above are analysed. The height is the highest height value taken from those 5 slices of data, e.g. the height value in Slice C.

[00221] Height values from multiple trees are grouped together in a sequence (e.g. a length of 5 trees from tree 1 to tree 5):

[Height A, Height B, Height C, Height D, Height E]

[00222] This sequence is then compared by the characterisation module to the baseline data set to find a match.

Volume Example

[00223] For volume values, an average of all the volume values for all slices associated with a tree is taken as the volume for that tree.

[00224] Volume values from multiple trees are grouped together in a sequence (e.g. a length of 5 trees from tree 1 to tree 5):

[Volume A, Volume B, Volume C, Volume D, Volume E]

[00225] This sequence is then compared by the characterisation module to the baseline data set to find a match.

Signature Descriptors

[00226] A number of feature values from consecutive 0.2 metre slices associated with a particular tree are grouped together to provide a signature value. The slices are determined by the segmentation module in the segmentation step, which identifies the individual trees. The length of the signature may be different dependent on the segmentation of the slices or width of the slices. For example, a first tree may be defined by a signature taken from 5 slices, whereas a second tree may be defined by a signature taken from 6 slices.

[00227] As the signatures may vary in width (because of the number of cells that contributed to the measurement of the tree features) and because perfect signature alignment is not guaranteed, it will be understood that comparison functions may be used to test multiple alignments to find the best match when comparing whether two signatures are similar.

Height Example

[00228] As shown in an example in Fig. 5, the highest point in each slice is used to determine the height for that 0.2 metre slice as discussed above.

[00229] This is repeated for each group of 0.2 metre slices associated with a particular tree to provide the signature.

[00230] E.g. [21.8; 22; 22.2; 21.8; 21.4] may be the height signature descriptor for the tree identified in Fig. 5 by Slices A to E.

Volume Example

[00231] The volume values calculated for each 0.2 metre slice associated with a particular tree are grouped together to form a volume signature descriptor for that tree. See above for volume calculation for each 0.2 metre slice.

Slice A Slice B Slice C Slice D Slice E [FOR TREE 1]

0.8 0.9 0.6 0.7 0.8

[00232] Producing a signature (volume) of [0.8, 0.9, 0.6, 0.7, 0.8].

[00233] This is repeated for each group of 0.2 metre slices associated with a particular tree.

[00234] Sequence matching by the sequencing module of the basic descriptors or signature descriptors in two or more sets of data enables monitoring of the natural elements to occur over time. The monitoring of the natural elements may include monitoring the natural features as previously discussed or may include monitoring other factors associated with the environment where it is important to understand the relative location of the point where the monitoring is taking place. For example, soil samples may be obtained at

regular intervals, crop yield may be determined for particular trees or plants, or indeed any other factor that is capable of being measured.

[00235] Further details of various aspects of the present invention are now described.

Data Collection and Datasets

[00236] A field trial was performed at an almond orchard in Mildura (Victoria, Australia). As seen in figure 6, a number of robots (601, 603, 605) were utilised in the data collection, however the work presented is based solely on the data collected by the robot known as *Shrimp*.

[00237] Data from two of the robot's sensors was utilised. The first one is a novatel navigation system, utilising DGPS to estimate the robot's position. The second one is a SICK LMS-291 2D Laser Scanner (LIDAR). The LIDAR is mounted vertically and oriented to the right, giving a sheet of measurements at every frame. Combining these two sensors, a 3D point cloud is created, an excerpt of which is presented in figure 7. Every point in the 3D point cloud is represented by an X, Y and Z coordinate.

[00238] Nine different datasets were collected during the trial. Three of these were collected by the robot moving in a straight line past both sides of the same row. A fourth dataset was collected by the robot traversing 6 different rows, including the row in the smaller datasets. Furthermore, a fifth dataset was collected by the robot traversing the same row in a sinusoidal motion, resulting in the data being rather distorted.

[00239] Additionally, 3 datasets containing only empty space were utilised, without any trees. To be noted is that these empty datasets were collected on the path right before entering the row of trees, and not in an empty area inside the actual rows. This resulted in the ground structure of these datasets being different from the empty spaces in the inner parts of the rows, as the ground in the rows has a hill-like structure not shared by the area next to the rows.

[00240] In order to simplify later references to the datasets, they are defined as:

[00241] The 3 shorter datasets - The 3 datasets collected by the robot moving in a straight line past both sides of the same row.

[00242] The longer dataset - The dataset collected by the robot traversing 6 rows in a straight line.

[00243] The 4 shorter datasets - The 3 shorter datasets and the corresponding row of the longer dataset.

[00244] The distorted dataset - The dataset collected by the robot moving in a sinusoidal motion.

[00245] The empty datasets - The 3 datasets containing only ground and no trees. Additionally, the same row was often viewed or seen from different sides as separate rows unless the similarity between the two sides is explicitly investigated. Utilising this view, the shorter and distorted datasets contain 2 rows and the larger dataset contains 10 rows.

[00246] Finally, in order to give an estimate of the size of the datasets, it should be mentioned that each row contains 58 trees, with an extent of roughly 320 meters, and is represented by approximately 1.5 million points. The empty datasets are significantly smaller, each having a length of approximately 10 meters.

[00247] There exist numerous approaches to segmenting 3D data, however many of these are general solutions, created to segment data consisting of undefined types of objects. Assuming that better results can be obtained by orchard specific methods, allowing integration of the known orchard structure, we focus on these methods. One such method uses Gaussian mixture models with unknown number of clusters to segment the individual points into trees. The method was evaluated in a peach orchard, showing good results, however the trees in the utilised datasets appear to have suffered from significantly less over- lap than the trees in an almond orchard.

[00248] Instead of a per-point segmentation, an alternative method based on splitting of the row into "slices" of 0.2 meters could be used. Based on the height of each slice, the dataset is segmented into trees by employing a Hidden Semi-Markov Model. Furthermore, this method also allows an a priori estimate of the widths of the tree to be incorporated into the segmentation.

[00249] In order to characterise the segmented trees, it is necessary to use a descriptor to describe the 3D data. There exists a multitude of 3D descriptors, one of the more well-known being spin-images, which describes the local neighbourhood with regard to an estimated normal direction. Other descriptors based on local normal estimation include shape contexts and point feature histogram. Furthermore, there exist other descriptors, such as shape distributions and shape functions, which are based on the distribution of the points.

[00250] One interesting localisation model is Dynamic Time Warping used to perform sequence-based visual localisation. High precision localisation can be achieved through sequence matching even when the information in each individual element is severely limited.

[00251] Another approach to localisation, which allows more modeling freedom, is to use Hidden Markov Models (HMMs). This was used to perform indoor localisation based on the recognition of landmarks such as door and corridors. Additionally, there exists more complex HMM localisation methods that integrate both landmark observations and robot odometry.

Further Description of Embodiments

[00252] First it is examined how to segment the orchard into individual trees using the segmentation module. In order to do so, the Hidden Semi-Markov Model approach is utilised. The model is extended, a number of different variants of the segmentation method are described and their performance is evaluated.

[00253] Secondly, it is investigated how to characterise the segmented trees using the characterisation module. One significant attribute of the utilised datasets is that the trees are sparsely sampled. This is a major problem for the spin-images, shape contexts and point feature histograms, as they all rely on the

correct estimation of surface normals. Furthermore, if the robot moves at non-uniform speed, or happens to turn slightly, the trees will be non-uniformly sampled. This is problematic when utilising shape distributions and shape functions, as it assumes that the point distribution is consistent.

[00254] Recognising these problems, a set of descriptors for the specific application is defined. It is shown how these can be used to build a map of the orchard and introduce a simple localisation algorithm. Additionally, it is investigated how the map of the orchard may best be built from multiple measurements of each tree.

[00255] A Hidden Markov Model is utilized with the previously introduced descriptors to create a robust localisation method. Performance is evaluated and it is shown that it is robust both to measurement noise and segmentation errors.

[00256] A problem with building the map of the orchard from multiple measurements is that segmentation errors, though supposedly few, risk corrupting the map. In order to address this problem, a method for detecting segmentation errors when performing localization is described.

[00257] In conclusion, the contributions presented in the document can be summarised as follows:

[00258] A method and system for segmenting individual trees from a 3D point cloud is described.

[00259] A number of tree descriptors are defined and their usefulness evaluated.

[00260] A robust localisation method and system is described and its performance evaluated.

[00261] A method and system for detecting segmentation errors when performing localisation is described.

Hidden Markov Models

[00262] Hidden Markov Models (HMMs) are used in a variety of areas such as speech recognition, computational biology and computer vision. At the core of the model is a set of states, each state producing an observation in accord with a probabilistic distribution. At each time, the model is in one active state, the state currently producing the observation. Furthermore, the probability of transitioning between the different states is also modeled. Having modeled the complete HMM it is possible to estimate the most probably sequence of states given a sequence of observations.

[00263] Three of the common algorithms for estimating the sequence of states are described.

Furthermore, it is described how the model may be utilised with multiple observation features as well as how to avoid numerical underflow problems. Additionally, in order to aid in the derivation of the introduced algorithms, the utilised probability theorems are provided at the end of the description. However, before presenting the optimisation methods, it is necessary to define the notation.

[00264] First of all, the probability of transitioning from state $q_t = i$ to state $q_{t+1} = j$ is defined to be independent of both previous states and of time. These assumptions make it possible to create a

transition matrix A with element $A_{ij} = P(q_{t+1} = j | q_t = i)$. The states and state transitions can be visualised in a state transition diagram, an example of which is seen in figure 8.

[00265] Secondly, the observation distribution of each state, $P(x|q)$, is modeled. This allows the likelihood of state $q_t = j$ given measurement x_t to be calculated. In order to simplify notation, the likelihood of $q_t = j$ for all observations are collected in a vector B_j such that $B_j(t) = P(x_t | q_t = j)$.

[00266] To use the model it is necessary to define the a priori probabilities of each state. These prior probabilities are denoted $\pi_j = P(q_1 = j)$ where $\sum \pi_j = 1$

[00267] To further simplify the notation we define $x_{1:T}$ to be the sequence of observations from x_1 to x_T . Similarly $q_{1:T}$ is defined to be the state-sequence from q_1 to q_T . Furthermore, the total number of states is defined as N .

Definition of Optimal States

[00268] A common application of Hidden Markov Models is to find the most probable state sequence given a sequence of observations. In order to investigate the problem however, it is necessary to properly define the optimality expression. In many applications it is of interest to find the individually most probable states given all measurements, formally expressed as

$$\hat{q}_{FB} = (\arg \max_{q_1} P(q_1, x_{1:T}), \arg \max_{q_2} P(q_2, x_{1:T}), \dots, \arg \max_{q_T} P(q_T, x_{1:T})). \quad (2.1)$$

[00269] One problem with the optimisation is that it is only possible in an offline situation. The equivalent causal problem is expressed as

$$\hat{q}_F = (\arg \max_{q_1} P(q_1, x_1), \arg \max_{q_2} P(q_2, x_{1:2}), \dots, \arg \max_{q_T} P(q_T | x_{1:T})). \quad (2.2)$$

[00270] The common denominator for both equation (2.1) and equation (2.2) is that they both target the individually most probable states. This approach is applicable in many instances, however it risks creating invalid state sequences if there exists any element $A_{ij} = 0$ in the transition matrix. In such situations it is better to optimise with regard to the jointly most probably state sequence, formally expressed as

$$\hat{q}_V = \arg \max_{q_1, q_2, \dots, q_T} P(q_{1:T}, x_{1:T}). \quad (2.3)$$

[00271] All three optimisation criteria are common in practical problems, with optimisation algorithms readily available for each one. The non-causal optimisation of equation (2.1) is performed using the Forwards-Backwards algorithm. Removing the anti-causal part of the Forwards-Backwards algorithm yields the Forwards algorithm used for optimising equation (2.2). The optimisation of equation (2.3) is done using the Viterbi algorithm.

[00272] It will be understood that it is possible to create the presented optimality criteria with conditional probabilities instead of joint probabilities. Joint probabilities were chosen over conditional probabilities to remove a scale factor. This has no effect on the choice of optimal sequence since the scale factor is the same for all states, however it slightly simplifies the derivation of the optimisation algorithms.

The Forwards Algorithm

[00273] The Forwards algorithm solves (2.2) using recursive updates. Begin by introducing

$$\alpha_t(j) = P(q_t = j, x_{1:t}), \quad (2.4)$$

[00274] recursively updated by

$$\begin{aligned} \alpha_t(j) &= P(q_t = j, x_{1:t}) = P(x_t | q_t = j) P(q_t = j, x_{1:t-1}) \\ &= P(x_t | q_t = j) \left(\sum_i P(q_t = j, x_{1:t-1}, q_{t-1} = i) \right) \\ &= P(x_t | q_t = j) \left(\sum_i P(q_t = j | x_{1:t-1}, q_{t-1} = i) P(x_{1:t-1}, q_{t-1} = i) \right) \\ &= P(x_t | q_t = j) \left(\sum_i P(q_t = j | q_{t-1} = i) P(q_{t-1} = i, x_{1:t-1}) \right) \\ &= B_j(t) \left(\sum_i A_{ij} \alpha_{t-1}(i) \right). \end{aligned} \quad (2.5)$$

[00275] Initialisation of α is performed by setting

$$\alpha_1(j) = P(q_1 = j, x_1) = P(x_1 | q_1 = j)P(q_1 = j) = B_j(1)\pi_j. \quad (2.6)$$

Having calculated α , the most probable state at each time is calculated as

$$q_t^* = \arg \max_j (\alpha_t(j)). \quad (2.7)$$

The complete algorithm is presented in algorithm 1.

Algorithm 1 The HMM Forwards Algorithm

```

for  $j = 1 \rightarrow N$  do                                     » Initialisation
     $\alpha_1(j) = B_j(1)\pi_j$ 
end for

for  $t = 2 \rightarrow T$  do                                     » Recursion
    for  $j = 1 \rightarrow N$  do
         $\alpha_t(j) = B_j(t)(\sum_i A_{ij}\alpha_{t-1}(i))$ 
    end for
end for

for  $t = 1 \rightarrow T$  do                                     » State Maximisation
     $q_t^* = \arg \max_j (\alpha_t(j))$ 
end for

```

The Forwards-Backwards algorithm

[00276] The Forwards-Backwards algorithm is the non-causal extension of the Forwards algorithm, employing a similar recursive algorithm for solving (2.1). Begin by introducing

$$\begin{aligned}
 \gamma_t(j) &= P(q_t = j, x_{1:T}) = P(x_{t+1:T} | x_{1:t}, q_t = j)P(x_{1:t}, q_t = j) \\
 &= P(x_{1:t}, q_t = j)P(x_{t+1:T} | q_t = j) \\
 &= \alpha_t(j)\beta_t(j)
 \end{aligned} \quad (2.8)$$

[00277] where

$$\alpha_t(j) = P(q_t = j, x_{1:t}), \quad (2.9)$$

[00278] as in the Forwards algorithm, and

$$\beta_t(j) = P(x_{t+1:T} | q_t = j). \quad (2.10)$$

[00279] The recursive update of α is performed using (2.5) while β is updated by employing

$$\begin{aligned}
\beta_t(j) &= P(x_{t+1:T}|q_t = j) = \sum_i P(x_{t+1:T}, q_{t+1} = i|q_t = j) \\
&= \sum_i P(x_{t+1}|x_{t+2:T}, q_{t+1} = i, q_t = j)P(x_{t+2:T}, q_{t+1} = i|q_t = j) \\
&= \sum_i P(x_{t+1}|q_{t+1} = i)P(x_{t+2:T}|q_{t+1} = i, q_t = j)P(q_{t+1} = i|q_t = j) \\
&= \sum_i P(x_{t+1}|q_{t+1} = i)P(x_{t+2:T}|q_{t+1} = i)P(q_{t+1} = i|q_t = j) \\
&= \sum_i B_i(t+1)\beta_{t+1}(i)A_{ji}.
\end{aligned} \tag{2.11}$$

[00280] The α and β variables are initialised by

$$\alpha_1(j) = P(q_1 = j, x_1) = P(x_1|q_1 = j)P(q_1 = j) = B_j(1)\pi_j \tag{2.12}$$

[00281] and

$$\beta_T(j) = P(x_{T+1:T}|q_T = j) = P(\emptyset|q_T = j) = 1. \tag{2.13}$$

[00282] Having calculated α and β , γ is calculated by

$$\gamma_t(j) = \alpha_t(j)\beta_t(j) \tag{2.14}$$

[00283] and the most probable state at each time calculated as

$$q_t^* = \arg \max_j \{\gamma_t(j)\}. \tag{2.15}$$

[00284] The algorithm in its entirety is presented in algorithm 2

Algorithm 2 The HMM Forwards-Backwards Algorithm

```

for  $j = 1 \rightarrow N$  do                                     » Initialisation
     $\alpha_1(j) = B_j(1)\pi_j$ 
     $\beta_T(j) = 1$ 
end for

for  $t = 2 \rightarrow T$  do                                     » Recursion - Forwards
    for  $j = 1 \rightarrow N$  do
         $\alpha_t(j) = B_j(t)(\sum_i A_{ij}\alpha_{t-1}(i))$ 
    end for
end for

for  $t = 1 \rightarrow T - 1$  do                                 » Recursion - Backwards
    for  $j = 1 \rightarrow N$  do
         $\beta_t(j) = (\sum_i B_i(t+1)\beta_{t+1}(i)A_{ji})$ 
    end for
end for

for  $t = 1 \rightarrow T$  do                                     » State Maximisation
    for  $j = 1 \rightarrow N$  do
         $\gamma_t(j) = \alpha_t(j)\beta_t(j)$ 
    end for

     $q_t^* = \arg \max_j \gamma_t(j)$ 
end for

```

The Viterbi Algorithm

[00285] The Viterbi algorithm solves (2.3) by employing a dynamic programming approach. Start by defining

$$\delta_t(j) = \max_{q_1, q_2, \dots, q_{t-1}} P(q_{1:t-1}, q_t = j, x_{1:t}), \quad (2.16)$$

thus δ is the joint probability of the most likely state sequence and the observations. The update step is

$$\begin{aligned}
 \delta_t(j) &= \max_{q_1, q_2, \dots, q_{t-1}} P(q_{1:t-1}, q_t = j, x_{1:t}) \\
 &= \max_{q_1, q_2, \dots, q_{t-1}} P(x_t | q_{1:t-1}, q_t = j, x_{1:t-1}) P(q_{1:t-1}, q_t = j, x_{1:t-1}) \\
 &= \max_{q_1, q_2, \dots, q_{t-1}} P(x_t | q_t = j) P(q_t = j | q_{1:t-1}, x_{1:t-1}) P(q_{1:t-1}, x_{1:t-1}) \\
 &= \max_{q_1, q_2, \dots, q_{t-1}} P(x_t | q_t = j) P(q_t = j | q_{t-1}) P(q_{1:t-1}, x_{1:t-1}) \\
 &= \max_i \left(P(x_t | q_t = j) P(q_t = j | q_{t-1} = i) \max_{q_1, q_2, \dots, q_{t-2}} P(q_{1:t-1} = i, q_{1:t-2}, x_{1:t-1}) \right) \\
 &= \max_i B_j(t) A_{ij} \delta_{t-1}(i),
 \end{aligned} \quad (2.17)$$

calculated as

[00286] Initialisation of δ is done by

$$\delta_1(j) = P(q_1 = j, x_1) = P(x_1|q_1)P(q_1) = B_j(1)\pi_j. \quad (2.18)$$

[00287] In order to do find the most probable sequence, the previous maximising state is stored in a variable Φ for each time and state:

$$\phi_t(j) = \arg \max_i B_t(j)A_{ij}\delta_{t-1}(i). \quad (2.19)$$

[00288] Having calculated δ for the entire sequence, the most probable final state is found as

$$q_T^* = \arg \max_j \delta_T(j). \quad (2.20)$$

[00289] Given the final state, the most probably state-sequence is found using Φ to *backtrack* along the sequence:

$$q_t^* = \phi_{t+1}(q_{t+1}^*). \quad (2.21)$$

[00290] The complete algorithm is presented in algorithm 3.

Algorithm 3 The HMM Viterbi Algorithm

<pre> for $j = 1 \rightarrow N$ do $\delta_1(j) = B_j(1)\pi_j$ $\phi_1(j) = 0$ end for </pre>	» Initialisation
<pre> for $t = 2 \rightarrow T$ do for $j = 1 \rightarrow N$ do $\delta_t(j) = \max_i B_j(t)A_{ij}\delta_{t-1}(i)$ $\phi_t(j) = \arg \max_i B_j(t)A_{ij}\delta_{t-1}(i)$ end for end for </pre>	» Recursion
<pre> $q_T^* = \arg \max_j \delta_T(j)$ </pre>	» Termination
<pre> for $t = T - 1 \rightarrow 1$ do $q_t^* = \phi_{t+1}(q_{t+1}^*)$ end for </pre>	» Back-Tracking

Use of Multiple Features

[00291] Previous sections have treated the observation x as a scalar value, however it is possible to use multivariate observations. Let

$$P(\mathcal{X}_i|q_j) = P((x_i^1, x_i^2)|q_j) = P(x_i^1, x_i^2|q_j) \quad (2.22)$$

where x_i^1 and x_i^2 denote observations of different features such as height and width. Furthermore, assuming that x_i^1 and x_i^2 are independent, (2.22) may be simplified to

$$P(\mathcal{X}_i|q_j) = P(x_i^1|q_j)P(x_i^2|q_j). \quad (2.23)$$

This allows multiple features to be used simply by replacing $B_j(t) = P(x|q_j)$ with $B_j(t) = P(x_i^1|q_j)P(x_i^2|q_j)$ in the used optimisation algorithm.

Finally, it should be observed that it is possible to use different features for different states without doing any changes to the optimisation algorithm. Setting

$$B_j(t) = P(x_i^1|q_j) \quad (2.24)$$

and

$$B_i(t) = P(x_i^2|q_i), \quad (2.25)$$

this equals defining

$$p(x_i^2|q_j) = P(x_i^2|q_i) = K = 1 \quad (2.26)$$

over an area of x large enough such that all observations fall within this area. This results in an incorrect probability for all states due to the fact that $\int_a^b 1 \neq 1$ in the general case. However, due to the error being the same for all state sequences, this does not affect the choice of state sequence.

Avoiding Numerical Underflow

[00292] A practical problem for all three presented optimisation algorithms is the under-flow problem resulting from the large number of products smaller than 1. This problem can be resolved either through scaling or by working in the logarithmic domain. The standard solution for the Forwards and Forwards-Backwards algorithms is the scaling method, however this method does not generalise to HSMMs, therefore we present the logarithmic approach. The core idea of the logarithmic approach is to optimise the logarithm of the probability, this effectively removes any risk of underflow problems since

$$\log(a \cdot b) = \log a + \log b. \quad (2.27)$$

Forwards Algorithm

[00293] The logarithmic version of the Forwards algorithm is presented in algorithm 4.

Algorithm 4 The Logarithmic HMM Forwards Algorithm

```

 $\hat{\pi} = \log(\pi)$  ▷ Calculate Log Probabilities
 $\hat{A} = \log(A)$ 
 $\hat{B} = \log(B)$ 

for  $j = 1 \rightarrow N$  do ▷ Initialisation
     $\alpha_1(j) = \hat{B}_j(1) + \hat{\pi}_j$ 
end for

for  $t = 2 \rightarrow T$  do ▷ Recursion
    for  $j = 1 \rightarrow N$  do
         $\alpha_t(j) = \hat{B}_j(t) + \log(\sum_i \exp(\hat{A}_{ij} + \alpha_{t-1}(i)))$ 
    end for
end for

for  $t = 1 \rightarrow T$  do ▷ State Maximisation
     $q_t^* = \arg \max_j (\alpha_t(j))$ 
end for

```

Forwards-Backwards Algorithm

[00294] The logarithmic version of the Forwards-Backwards algorithm is presented in algorithm 5.

Algorithm 5 The Logarithmic HMM Forwards-Backwards Algorithm

```

 $\hat{\pi} = \log(\pi)$  ▷ Calculate Log Probabilities
 $\hat{A} = \log(A)$ 
 $\hat{B} = \log(B)$ 

for  $j = 1 \rightarrow N$  do ▷ Initialisation
     $\alpha_1(j) = \hat{B}_j(1) + \hat{\pi}_j$ 
     $\beta_T(j) = 0$ 
end for

for  $t = 2 \rightarrow T$  do ▷ Recursion - Forwards
    for  $j = 1 \rightarrow N$  do
         $\alpha_t(j) = \hat{B}_j(t) + \log(\sum_i \exp(\hat{A}_{ij} + \alpha_{t-1}(i)))$ 
    end for
end for

for  $t = 1 \rightarrow T - 1$  do ▷ Recursion - Backwards
    for  $j = 1 \rightarrow N$  do
         $\beta_t(j) = \log(\sum_i \exp(\hat{B}_i(t+1) + \beta_{t+1}(i) + \hat{A}_{ji}))$ 
    end for
end for

for  $t = 1 \rightarrow T$  do ▷ State Maximisation
    for  $j = 1 \rightarrow N$  do
         $\gamma_t(j) = \alpha_t(j) + \beta_t(j)$ 
    end for
     $q_t^* = \arg \max_j (\gamma_t(j))$ 
end for

```

Viterbi Algorithm

[00295] Due to the fact that

$$\log(\max f(\cdot)) = \max(\log f(\cdot)), \quad (2.28)$$

[00296] the derivation of the modified Viterbi algorithm is very straight-forward. The resulting algorithm is presented in algorithm 6.

Algorithm 6 The Logarithmic HMM Viterbi Algorithm

```

 $\bar{\pi} = \log(\pi)$  ▷ Calculate Log Probabilities
 $\bar{A} = \log(A)$ 
 $\bar{B} = \log(B)$ 

for  $j = 1 \rightarrow N$  do ▷ Initialisation
     $\delta_1(j) = \bar{B}_j(1) + \bar{\pi}_j$ 
     $\phi_1(j) = 0$ 
end for

for  $t = 2 \rightarrow T$  do ▷ Recursion
    for  $j = 1 \rightarrow N$  do
         $\delta_t(j) = \max_i (\bar{B}_j(t) + \bar{A}_{ij} + \delta_{t-1}(i))$ 
         $\phi_t(j) = \arg \max_i (\bar{B}_j(t) + \bar{A}_{ij} + \delta_{t-1}(i))$ 
    end for
end for

 $q_T^* = \arg \max_j \delta_T(j)$  ▷ Termination
for  $t = T - 1 \rightarrow 1$  do ▷ Back-Tracking
     $q_t^* = \phi_{t+1}(q_{t+1}^*)$ 
end for

```

Hidden Semi-Markov Model

[00297] The Hidden Semi-Markov Model (HSMM) is an extension of the Hidden Markov Model where the time spent in each state is explicitly modeled. In this chapter the model is introduced and the corresponding optimisation algorithms used in the document. This extension is important for the reasons described below.

[00298] First, the probability of a Hidden Markov Model remaining in the same state for exactly d observations is

$$P(\text{Exactly } d \text{ consecutive observations of state } j) = (A_{jj})^{d-1} (1 - A_{jj}). \quad (3.1)$$

[00299] This gives, since

$$(A_{jj})^{d-1} (1 - A_{jj}) \propto \exp(d \log(A_{jj})), \quad (3.2)$$

[00300] that the probability of the model remaining in the same state is effectively de-creasing exponentially with time, thus limiting the usefulness of HMMs. To further stress this point, figure 9

compares the implicit duration distribution 901 of a HMM with the explicit duration distribution 903 of a HSMM.

[00301] As mentioned, the key feature of the HSMM is that the duration spent in each state is explicitly modeled using a probability distribution. This allows the transitions between states to be defined in the same way as in HMMs while allowing complete freedom in modeling the duration distribution. Taken together, this allows most of the simplicity of the HMM to be kept while making the model much more accurate for many problems.

[00302] In order to specify the notation, we define

$$C_j(d) = P(\text{Exactly } d \text{ consecutive observations of state } j). \quad (3.3)$$

[00303] Furthermore, the integer variable D is introduced to define the maximum duration spent in a state. This variable adds no information to the model from a theoretical viewpoint, however it is necessary to limit the computational complexity of the optimisation algorithms.

[00304] As only the HSMM variant of the Viterbi algorithm is used in this document, the HSMM variants of the Forwards and Forwards-Backwards algorithms are not presented here. However, they are explained in detail in "Hidden Semi-Markov Models (HSMMS)", 2002 by Kevin P. Murphy.

3.1 The Viterbi Algorithm

[00305] In order to derive the HSMM version of the Viterbi Algorithm the notation introduced in "Hidden Semi-Markov Models (HSMMS)", 2002 by Kevin P. Murphy is used.

- $G_t = (Q_t, L_t)$ is a generalised form of the ordinary state containing both the ordinary state, Q_t , and the duration of the state, L_t . The suffix t means that the state ends at time t , i.e. $Q_{t+1} \neq Q_t$.
- G_{t_p} is the previous state, i.e. the state exactly before G_t .
- G_t^- are all states prior to G_t .
- $Y(G_t)$ are all observations of G_t . In the same way $Y(G_t^-)$ are all observations prior to G_t .

[00306] This allows a δ , similar to the one used in the HMM Viterbi algorithm, to be introduced as

$$\begin{aligned}
\delta_t(g) &= \max_{G_t^-} P(G_t = g, G_t^-, Y(G_t), Y(G_t^-)) \\
&= \max_{G_t^-} P(Y(G_t) | G_t = g) P(G_t = g, G_t^-, Y(G_t^-)) \\
&= \max_{G_t^-} P(Y(G_t) | G_t = g) P(G_t = g | G_t^-) P(G_t^-, Y(G_t^-)) \\
&= \max_{G_{t_p}^- = g'} \left(P(Y(G_t) | G_t = g) P(G_t = g | G_{t_p} = g') \max_{G_{t_p}^-} \left(P(G_{t_p} = g', G_{t_p}^-, Y(G_{t_p}), Y(G_{t_p}^-)) \right) \right). \tag{3.4}
\end{aligned}$$

[00307] Due to the fact that t_p is a random variable, equation (3.4) cannot be implemented in practice. In order to resolve this, $G_t = (Q_t, L_t)$ is inserted. Furthermore, a variable F is introduced to signify when a state finishes:

$$F_t = \begin{cases} 1 & , \text{ if } Q_{t+1} \neq Q_t \\ 0 & , \text{ else} \end{cases} \tag{3.5}$$

[00308] It is also used that

$$P(j, d | i, d') = P(d | j) P(j | i)$$

[00309] thus assuming both that the duration of the current state is independent of the previous state and that the transition probabilities are independent of the duration of the states. Applying this to (3.4) yields

$$\begin{aligned}
\delta_t(j, d) &= \max_{G_t^-} P(Q_t = j, L_t = d, F_t = 1, y_{1:t}, G_t^-) \\
&= \max_{i, d'} \left(P(y_{1:t-d+1:t} | j, d) P(j, d | i, d') \max_{G_{t-d}^-} \left(P(Q_{t-d} = i, L_{t-d} = d', F_{t-d} = 1, y_{1:t-d}, G_{t-d}^-) \right) \right) \\
&= \max_{i, d'} \left(\left(\prod_{s=t-d+1}^t B_j(s) \right) P(d | j) P(j | i) \delta_{t-d}(i, d') \right) \\
&= \max_{i, d'} \left(\left(\prod_{s=t-d+1}^t B_j(s) \right) C_j(d) A_{ij} \delta_{t-d}(i, d') \right), \tag{3.6}
\end{aligned}$$

[00310] It can be observed that the update step is similar to the ordinary HMM algorithm, the difference being that the duration probabilities are factored in and that it is necessary to search for the maximising arguments over the duration of the states. Furthermore the δ in the HMM algorithm is defined as the joint probability of being in state j at time t . In the HSMM algorithm however it is defined as the joint probability of transitioning from state j at time t .

[00311] In order to do a proper initialisation it is necessary to initialise δ not only for $t = 1$ but for all $t \leq D$. The initialisation is expressed as

$$\delta_t(j, t) = \left(\prod_{s=1}^t B_j(s) \right) C_j(t) \pi_j. \quad (3.7)$$

[00312] As in the HMM algorithm the sequence path has to be stored in a storage variable ϕ . In order to store the path sequence, it is necessary to store both the previous state and the duration spent in the previous state. This is expressed as

$$\begin{aligned} (i^*, d^*) &= \arg \max_{1 \leq i \leq N, 1 \leq d' \leq D} \left(\left(\prod_{s=t-d'+1}^t B_j(s) \right) C_j(d') A_{ij} \delta_{t-d'}(i, d') \right) \\ \phi_t(j, d) &= (t-d, i^*, d^*) \end{aligned}$$

[00313] for the update steps. Furthermore, it is necessary to initialise the storage variable as

$$\phi_t(j, t) = (0, 0, 0)$$

[00314] for all $t \leq D$.

[00315] The last state is found as

$$\begin{aligned} (j_n^*, d_n^*) &= \arg \max_{1 \leq j \leq N, 1 \leq d' \leq N} \delta_T(j, d') \\ t_n &= T. \end{aligned}$$

[00316] and the back-tracking as

$$(t_n, j_n^*, d_n^*) = \phi_{t_{n-1}}(j_{n-1}^*, d_{n-1}^*). \quad (3.8)$$

[00317] The complete algorithm is presented in algorithm 7.

Algorithm 7 The HSMM Viterbi Algorithm

```

for  $j = 1 \rightarrow N$  do                                     » Initialisation
  for  $t = 1 \rightarrow D$  do
     $\delta_t(j, t) = \left( \prod_{s=1}^t B_j(s) \right) C_j(t) \pi_j$ 
     $\phi_t(j, t) = (0, 0, 0)$ 
  end for
end for

for  $t = 2 \rightarrow T$  do                                     » Recursion
  for  $j = 1 \rightarrow N$  do
    for  $d = 1 \rightarrow D$  do
      if  $d \geq t$  then
        continue
      end if
       $\delta(t, j, d) = \max_{1 \leq i \leq N, 1 \leq d' \leq D} \left( \prod_{s=t-d+1}^t B_j(s) \right) C_j(d) A_{ij} \delta_{t-d}(i, d')$ 
       $(i^*, d^*) = \arg \max_{1 \leq i \leq N, 1 \leq d' \leq D} \left( \prod_{s=t-d+1}^t B_j(s) \right) C_j(d) A_{ij} \delta_{t-d}(i, d')$ 
       $\phi_t(j, d) = (t - d, i^*, d^*)$ 
    end for
  end for
end for

 $(j_n^*, d_n^*) = \arg \max_{1 \leq j \leq N, 1 \leq d' \leq N} \delta_T(j, d')$                                      » Termination
 $t_n = T$ 

while  $(t_n - d_n) > 1$  do                               » Back-Tracking
   $(t_n, j_n^*, d_n^*) = \phi_{t_n}(j_n^*, d_n^*)$ 
end while

```

Avoiding Numerical Underflow

[00318] Similarly as for the HMM optimisation algorithms, the HSMM variants suffer from underflow problems. As mentioned above, for HMMs this may be resolved either through the use of scaling or by working in the logarithmic domain. However, scaling is not a valid approach for HSMMs, thus only the logarithmic method is applicable. Using the same approach as presented in the results above results in algorithm 8.

Algorithm 8 The Logarithmic HSMM Viterbi Algorithm

```

 $\bar{\pi} = \log(\pi)$  » Calculate Log Probabilities
 $\bar{A} = \log(A)$ 
 $\bar{B} = \log(B)$ 
 $\bar{C} = \log(C)$ 

for  $j = 1 \rightarrow N$  do » Initialisation
  for  $t = 1 \rightarrow D$  do
     $\delta_t(j) = (\sum_{s=1}^t \bar{B}_j(s)) + \bar{C}_j(t) + \bar{\pi}_j$ 
     $\phi_t(j) = 0$ 
  end for
end for

for  $t = 2 \rightarrow T$  do » Recursion
  for  $j = 1 \rightarrow N$  do
    for  $d = 1 \rightarrow D$  do
      if  $d \geq t$  then
        continue
      end if

       $\delta_t(j, d) = \max_{1 \leq i \leq N, 1 \leq d' \leq D} (\sum_{s=t-d+1}^t \bar{B}_j(s)) + \bar{C}_d(t) + \bar{A}_{ij} + \delta_{t-d}(i, d')$ 
       $(i^*, d^*) = \arg \max_{1 \leq i \leq N, 1 \leq d' \leq D} (\sum_{s=t-d+1}^t \bar{B}_j(s)) + \bar{C}_d(t) + \bar{A}_{ij} + \delta_{t-d}(i, d')$ 
       $\phi_t(j, d) = (t - d, i^*, d^*)$ 
    end for
  end for
end for

 $(j_n^*, d_n^*) = \arg \max_{1 \leq j \leq N, 1 \leq d' \leq N} \delta_T(j, d')$  » Termination
 $t_n = T$ 

while  $(t_n - d_n) > 1$  do » Back-Tracking
   $(t_n, j_n^*, d_n^*) = \phi_{t_n}(j_n^*, d_n^*)$ 
end while

```

Segmentation

[00319] A key part of running an orchard is to keep an up to date inventory of the trees, this is necessary both to estimate the crop yield and health, as well as for general farm management. Currently, much of this labour is performed manually on selected parts of the orchard, and the results extrapolated to account for the entire orchard. Automation of this process would thus not only result in reduced labour, but also in more accurate estimates. However, in order to perform estimates on a per tree level, it is

necessary to determine the extent of the individual trees. This process is denoted as “segmentation”, as it involves dividing the rows into segments based on the extent of the individual trees.

[00320] Segmentation is based on a 3D point cloud, presented above. The reason for basing the segmentation on 3D data is that it has a high signal to noise ratio, due to the contrasting depths in the data. Furthermore, it is robust to illumination variance. This can be contrasted to the possible use of visual sensors, suffering both from the general similarity of the environment while also being sensitive to illumination changes.

[00321] A method based on a Hidden Semi-Markov Model (HSMM) is used, integrating knowledge about both tree spacing and measured tree features. In order to improve the performance further, a number of different variants of the introduced method are described. Finally, the different methods are evaluated and their results compared.

Modeling the orchard

[00322] A 3 state HSMM model for modeling an orchard is known. In this example, the model contains 3 states: *tree*, *gap* and *border*. The *tree* represents a tree in the orchard, *gap* represents a space without a tree and *border* represents the transition between the other two states. Of importance is that *border* does not only represent the transition between *tree* and *gap* but also from *tree* to *tree* and *gap* to *gap*. This setup assures that all legal state sequences effectively delimit the individual trees with *borders*.

[00323] The observations of the HSMM are calculated by extracting features from the 3D point cloud. This is done by dividing the rows into vertical “slices” of a specified width. A feature is then extracted from each slice. This approach gives a sequence of observations that can be input to the Viterbi algorithm as described herein.

[00324] In order to finalise the model, it is necessary to specify the state duration distributions. These distributions are a key part of the model and allow the integration of a priori knowledge of tree width to be integrated into the model.

Transition Probabilities

[00325] In order to set the notation, define the states as

$$\begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix} = \begin{pmatrix} q_{tree} \\ q_{gap} \\ q_{border} \end{pmatrix}. \quad (4.1)$$

[00326] To assure that the time spent in each state is uniquely controlled by the state duration densities, the self-transition probability is set to 0 for all states. Furthermore, set the transition probability between *tree* and *gap* to 0. This gives the transition probability from *tree* and *gap* to *border* is 1. The transition probability from *border* to *tree* and *gap* is set to 0.5. The complete transition matrix is given as

$$A = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0.5 & 0.5 & 0 \end{pmatrix}. \quad (4.2)$$

[00327] and the corresponding state transition diagram is presented in figure 10.

Initial Probabilities

[00328] To adhere to the standard of bounding *gaps* and *trees* by *borders*, the initial probabilities are defined as

$$\pi = (0 \quad 0 \quad 1). \quad (4.3)$$

Choice of Observation Feature

[00329] Identifying the state sequence is dependent on how informative the observation features are.

There exists a multitude of possible feature and feature combinations. A problem with the height feature is that it is heavily affected by small branches, making it hard to detect the borders between overlapping trees.

[00330] Another feature is volume, a benefit being that it is less heavily affected by minor branches. A third feature is a point-count, giving a rough estimate of the density of the slice. In order to set the notation, we denote these 3 features as the original features in order to distinguish them from the features that may be derived from them.

[00331] Furthermore, the level of noise has a great effect on the performance of the model, thus being necessary to take into consideration when choosing feature. As the noise level is closely related to the slice width, we discuss the choice of slice resolution in the next section.

Resolution

[00332] The resolution is defined to be the width of each slice. It is difficult to quantify the exact effects of the resolution, however some broad points can be noted. A larger resolution, thinner slices, allows for a more exact segmentation of the dataset. On the other hand, a too small resolution makes the features more susceptible to noise. It is thus necessary to balance segmentation preciseness against the noise level. Having no method for performing such analyses theoretically, the problem was examined by calculating and plotting the three original features for different resolution. The usefulness of each resolution was determined by how easy it was to detect the borders between the individual trees using visual inspection. From the results of the analysis, it was determined that a resolution of 0.2m is suitable. Having a larger resolution than this had significant negative effects on the noise level while a lower resolution did not improve the noise level by any significant degree.

Height

[00333] The height feature is calculated by finding the largest height of the slice. In order to get an estimate of its usefulness, the feature was calculated and plotted for one examined row of trees. Extracts

of the row presented in figures 11A (measured height over first 80 meters) and 11B (measured height from 80 to 160 meters) show that most trees and gaps are distinct. However it also shows that the border seems to be unclear between some trees, presumably because of overlap between the trees.

Density

[00334] The density feature is calculated by counting the number of points in the slice. One negative aspect of the feature is that it is correlated to the speed of the robot. It is possible that this can be handled by dividing the point feature with the current speed of the robot, however this may be problematic if the robot is not moving in a straight line. However, addressing this issue is out of the scope of this document. One reason for this choice is that most of the datasets were collected by the robot moving at uniform speed, thus any negative effects from the correlation should be minimal.

[00335] The usefulness of the density feature was evaluated in the same way as the usefulness of the height feature. The plots in figures 12A and 12B show that most trees are clearly distinct. Furthermore, the feature seems not to be as heavily affected by overlapping branches as the height feature. A negative aspect of the feature however is that it does not appear to be as sensitive to the very small trees as the height feature.

Volume

[00336] In order to calculate the volume feature, the slice is first split into uniform voxels of width 0.2m. Thereafter the volume is computed by adding the volume of all voxels containing at least one point. The usefulness of the feature was evaluated in the same way as the height and density features. Figures 13A and 13B show that most trees are very distinct. Furthermore, the feature seems to be less noisy than the density feature while also suffering from the same lack of sensitivity to very small trees. Taking these points into consideration, it was decided to use volume as the primary observation feature.

Volume Difference

[00337] A problem with using height difference to indicate border states is that difference calculations amplifies the noise of the data. As can be seen from figure 14 the noise level is clearly too large for the used dataset. In order to negate the effect of the noise, it was decided to introduce an ad hoc feature termed "difference of moving averages".

[00338] The "difference of moving averages" feature is computed by calculating the difference between one large moving average and one small moving average. The large moving average, calculated by filtering the volume feature with a rectangular filter of width 30, gives an estimate of the volume in a larger neighbourhood around the current slice. The small moving average, calculated by filtering the volume feature with a rectangular filter of width 5 gives a smoothed estimate of the volume at the current slice. The definition of the feature means that it yields large values at borders between large trees. Similarly, it yields large negative values both for tree centres and at the borders between gaps and trees. An example of resulting feature values is presented in figure 15. The measured volume 1501 and the difference of moving averages 1503 are shown. Note that the difference of moving averages yields clear peaks between large trees.

Hand-Tuned Observation Distributions

[00339] It is known to achieve good results using hand-tuned observation distributions. Using this approach, the observation distributions were hand-tuned by visually inspecting figures 13A and 13B. The observation distribution of *gap* was set such that it is more likely when a small volume is observed. Similarly, the observation distribution of *tree* was set such that it is more likely when a large volume is observed. The observation distribution based on the "difference of moving averages" was set to be moderately large for small values and large for large values. The reason for having the distribution moderately large for small values is that these may both indicate a tree centre and a border between a gap and a tree. The resulting observation distributions are presented in figures 16 A and 16B. In Fig. 16A the observation distributions for the tree 1601 and gap 1603 states are shown. In Fig. 16B, the observation distribution 1605 for the border state is shown.

State Duration Distributions

[00340] The state duration distributions are used to embed knowledge of the spatial dimensions of the orchard into the HSMM model. It has previously been known to approach the problem by modelling the duration of the *trees* and *gaps* with a Gaussian distribution centered on a mean. Furthermore, the standard deviation of the Gaussian was made fairly broad to allow for smaller trees. Finally, the Gaussian was truncated at a certain point as to set an upper limit of the tree width.

[00341] This approach is reasonable for trees; however it is problematic for gaps. The reason is that the gap between a replanted tree and its full grown neighbours is at most half the width of a tree. Modelling the width of the gap with the same distribution as the trees would thus introduce a bias into the model, possibly forcing small trees to be labeled as gaps. Furthermore, there exist many small gaps between large trees in the dataset, their extent being badly modelled by a Gaussian based on the width of the trees. Taking these issues into consideration, the duration distribution of the *gap* state is set to be uniform.

[00342] The duration distribution of *border* state is set to Kronecker's delta, thus limiting the duration of the border to 1.

[00343] In order to limit both the computational complexity and the width of the trees, the maximum duration D is set to 35 observations (7 meters). The Gaussian *tree* distribution is set to have a mean of 25 observations (5 meters) and a standard deviation of 5 observations (1 meter). The duration distributions are displayed in figures 17 and 18. Figure 17 shows the state duration distributions for the tree 1701 and gap 1703. Figure 18 shows the state duration distribution for the border 1801.

Model Alterations and Extensions

[00344] A 3 state model was presented above. This section describes how this model can be altered and extended to increase both its performance and its simplicity of use. First it is described how the observation and duration distributions can be learnt from a labeled dataset. Secondly, the model is extended by introducing a ground removal pre-processing step. Thirdly, it is described how the model can

be extended to a 4 state model in order to increase its performance with regard to small trees. Finally, the possible usage of the height and density features is described.

Learnt Observation Distributions

[00345] Instead of using hand-tuned observation distributions, such as those presented in above, it is possible to learn the distributions from a labeled dataset. The histograms of the observations for the 3 different states are shown in figures 19A, 19B and 19C. Figures 19A and 19B show that both tree and gap distributions are fairly close to Gaussian while the border distribution in figure 19C is clearly non-Gaussian. Despite this, we still decide to model all observation distributions as Gaussian in order to investigate if it is possible to achieve good results using a very simple learning method.

[00346] Assuming that the distributions are Gaussian, they can be learnt by estimating the mean and variance of the observations for the different states. Performing this operation, using a dataset labeled with the model described in the “modeling the orchard” section, yields the observation distributions presented in figure 20. The learnt observation distributions for tree 2001, gap 2003 and border 2005 are shown.

Learnt Tree Duration Distributions

[00347] The tree duration distribution described in the “modeling the orchard” section was modeled as a Gaussian distribution. Figure 21 shows that this model fits the data rather well, with a central Gaussian and a few outliers originating from the smaller trees in the dataset. Furthermore, this allows the tree duration distribution to be learnt from a labeled dataset by estimating the mean and variance of the duration. This, combined with the results presented in the learnt observation distributions section, gives that all critical parameters in the HSMM model can be learnt from a labeled dataset.

Ground Removal

[00348] One potential problem with the use of the volume feature is that the measured volume is not only the volume of the trees but also the volume of the ground. Given that the volume of the ground remains constant, this is not a problem. However, if there is a difference in the ground volume estimates, then the ground might have negative effects on the segmentation, most notably with regard to small trees. One situation where there is such a difference, is in the ground just before entering the row of trees. As this ground is flat, compared to the slightly hilly structure of the ground in the rows, a slight increase in volume is estimated due to the back of the “hill” not being occluded.

[00349] In order to avoid such effects, the ground was removed from the dataset in a pre-processing step using readily available matlab functionality. The effect of the pre-processing can be seen in figures 22A and 22B, showing that most of the ground has been correctly removed. Furthermore, comparisons of the estimated volumes with and without ground removal, presented in figure 23, show that the pre-processing might make the small trees slightly more distinct. Figure 23 shows a comparison of volume measurements before 2301 and after 2303 removing the ground. Note that the two small trees are more distinct after the ground has been removed.

[00350] In order to use the pre-processing method with the HSMM model, it is necessary to change the *tree* and *gap* observation distributions. These observations distributions are presented in figure 24. Figure 24 shows the hand-tuned observations distributions with ground removed for tree 2401 and gap 2403.

State Model

[00351] A weakness of the 3 state model presented in the "modeling the orchard" section is that it sometimes fails to detect small trees. There are two different reasons for this. First, the durations of the very small trees are much shorter than the modeled duration of the *tree* state. Secondly, these trees yield very small volume measurements, thus in many cases appearing more likely to originate from the *gap* state rather than the *tree* state according to the used observation distributions. In order to solve these problems a fourth state is introduced, a state specifically designed to represent a *small tree*.

[00352] The small tree is given an observation distribution with a significantly lower mean than tree. The observation distributions of all states are presented in figures 26A and 26B. Furthermore, the duration distribution is modeled to be uniform with a maximum duration of 5 observations (1 meter), the short duration chosen as the 3 state model appears to correctly handle trees larger than this width. Figure 26A shows the hand-tuned observation distributions for tree 2601, gap 2603 and small tree 2605. Figure 26B shows the hand-tuned observation distributions, for the ground re-moved data, for tree 2607, gap 2609 and small tree 2611.

[00353] As the trees modeled with *small tree* are very thin, any *borders* delimiting the trees are bound to represent a large part of these trees. Furthermore, the *border* state is modeled on the transitions to and from large trees, making it suboptimal to model transitions to and from small trees. This problem is avoided by deciding that *small trees* are bounded by *gaps* instead of *borders*. This yields the transition matrix:

$$A = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \quad (4.4)$$

and the extended state transition diagram in figure 25. Note that the explicit state durations are not shown.

[00354] A problem with the presented model is that both *gap* and *small tree* lacks a minimum duration. This presents a risk that the model will switch back and forth between *gap* and *small tree* under the influence of noise. In order to avoid this problem and make the model more robust to noise, a minimum duration of 5 observations is introduced for the *gap* state.

Density Observations

[00355] The discussions in this section have focused on the volume feature, however all presented methods may also be used in conjunction with the density feature. To be noted is that no hand-tuned

observation distributions were created for this feature, instead it was only used in conjunction with learnt observation distributions.

Height Observations

[00356] In order to evaluate the feasibility of using the height feature, a set of hand-tuned height observation distributions were created. Furthermore, using this labeled dataset, learning of the height distributions was done in the same way as was presented in the “learnt observation distribution” section.

Combined Volume and Height Observations

[00357] The discussion in the “choice of observation feature” section implies that the volume feature is good for delimiting large trees while the height feature is good for detecting small trees. In order to utilise both of these advantages, it is possible to combine both features. One way of combining the features is to allow each state to utilise both volume and height observations. However, since the height observations have negative effects on the segmentation of large trees, different observation features for different states were used. The *tree* and *border* states are defined to be observed by the volume feature and the *gap* and *small tree* to be observed by the height feature. Furthermore, this combination both for hand-tuned and learnt observation distributions.

Qualitative Evaluation

[00358] This section presents a qualitative evaluation of the results of the models introduced in the “modeling the orchard” and “model alterations and extensions” sections. First of all however, it is explained how small, medium and large sized trees, terms are defined that are necessary to explain the performance of the segmentation.

[00359] Large trees are defined as the trees which are full grown and, much of the time, overlapping with neighbouring trees. A section of large trees is seen in figure 27. Small trees are trees which have a width smaller than or equal to 1 meter, an example of which is seen in figure 28. Medium sized trees are the trees between these two extremes, they are smaller than the large trees and do not overlap their neighbours, at the same time being larger than the small trees and thus easier to detect. An example of a medium sized tree is presented in figure 29.

HMM vs HSMM

[00360] In order to show the benefit of the HSMM model, we compare it to an ordinary HMM model. The used HMM model is based the HSMM model discussed above, the difference being that the duration distributions are replaced with self transition matrices defined as:

$$A = \begin{pmatrix} 0.8 & 0 & 0.2 \\ 0 & 0.9 & 0.1 \\ 0.5 & 0.5 & 0 \end{pmatrix}. \quad (4.5)$$

[00361] Fig. 30 shows an extract of the resulting segmentation using an ordinary hidden Markov model.

[00362] Fig. 31 shows an extract of volume measurements 3101 and stated changes 3103 when using an ordinary Hidden Markov Model. The state being 1 equals tree, 0.5 equals border and 0 equals gap. Note the quick changes between tree and border that occur on numerous occasions.

[00363] Fig. 32 shows an extract of the resulting segmentation using the 3 state HSMM model.

[00364] The resulting segmentation when using this model is presented in figure 30, showing that many trees are far too wide. Furthermore, figure 31 shows that some trees are very thin. Given the limitations of the HMM model, it is impossible to address both these problems at the same time. These results can be compared to those presented in figure 32, showing that the resulting segmentation when utilising the HSMM model is not affected by these two problems.

Hand-Tuned vs Learnt Distributions

[00365] Using the learning methods described above to learn the duration and observation distributions, the resulting segmentation is similar to the one obtained with the hand-tuned equivalent. However, the segmentation errors, although few, appear to be slightly more frequent when using learnt distributions. Additionally, it should be noted that the learning does not appear to be sensitive to the choice of labeled dataset as long as it contains all types of states.

Ground Removal

[00366] Using the ground removal pre-processing does not yield any notable effect on the detection of small trees when utilising the volume feature. However, removing the ground prevents the volume variations outside the rows from creating false trees.

[00367] When using the tuned height feature, the effects of the ground removal is negligible. However, when using the learnt height feature, the results appear slightly worse. This is assumed to be due to new height variations occurring because of all points being removed in some slices while not in others.

3 vs 4 State

[00368] Both the 3 state and 4 state models yield similarly good results for large sized trees. The 3 state model also segments all medium sized trees correctly. However, the 4 state model labeled a few medium sized trees slightly wider than 5 observations (1 meter) as *small trees*. This limits the extent of the tree to 5 observations, resulting in its non-central parts being labeled as gaps. Furthermore, the minimum gap duration sometimes causes ground between two trees to be labeled as part of the trees, an example of which is seen in figure 33. Figure 33 shows an extract of the resulting segmentation when the gap between trees is labeled as part of the trees.

[00369] Regarding the small trees, the 3 state model fails to detect many of them. At the same time, the 4 state model is able of segmenting all small trees in the examined datasets, even when ground removal is not applied.

Choice of Feature

[00370] Using the volume feature yields very good results, properly segmenting trees of all sizes. However, unless ground removal is used, it also tends to create false trees in the open areas directly outside the rows. The density feature yields similar, albeit slightly worse results.

[00371] The height feature yields good results for small and medium sized trees, however it often fails to correctly segment overlapping trees. Furthermore, it does not appear to have any problem with creation of false trees in the empty datasets.

[00372] Utilising the combination of volume and height observations discussed in the "combined volume and height observations" section yields very good results when using the 4 state model with hand-tuned distributions. However, when the observation distributions are learnt, the introduced lower limit of the duration of the gap causes some segmentation errors when small and large trees are close to each other, an example of this is shown in figure 34. Fig. 34 shows a segmentation failure when the 4 state model is used with learn volume and height observation distributions. Nevertheless, using learnt distributions with the 3 state model yields very good results, even with regard to small trees. Notably, the hand-tuned observation distributions utilised with the 3 state model yields poor results with regard to small trees, assumedly because of bad tuning of the observation distributions.

[00373] The evaluation shows that there are multiple methods that yield equally good results. Using the volume feature and a 4 state model yields good results inside the rows, however, unless ground removal is used, it may yield false trees just before entering the rows. Furthermore, the method yields equally good results both with hand-tuned and learnt distributions.

[00374] Additionally, the combination of height and volume features was also shown to yield good results. When using hand-tuned distributions, the 4 state model yields better results. However, using learnt distributions, similar results can be obtained using the 3 state model. Furthermore, this method appears robust both with regard to detection of small trees as well as the creation of false trees before entering the rows.

[00375] Excerpts of the performance of the 4 state model, using hand-tuned volume and height distributions are shown in figures 35 and 36, showing that the large trees are correctly segmented. Figure 35 shows an overview of the resulting segmentation using the 4 state model with hand-tuned volume and height observation distributions. Fig. 36 shows an extract of the resulting segmentation using the 4 state model with hand-tuned volume and height observation distributions.

[00376] Furthermore, figures 37 and 38 show that small and most medium sized trees are correctly segmented. Figure 37 shows two medium sized trees segmented using the 4 state model with hand-tuned volume and height observation distributions. Figure 38 shows two small trees segmented using the 4 state model with hand-tuned volume and height observation distributions. Additionally, an empty dataset is shown in figure 39, showing that no false trees have been created. Figure 39 shows an empty part just before entering the rows segmented using the 4 state model with hand-tuned volume and height observation distributions. Finally, figure 40 shows an example of a medium sized tree incorrectly labeled

as a *small tree*, thus giving it a too small extent. Figure 40 shows a medium sized tree labeled as small tree, resulting in parts of the tree being labelled as gap due to the limited duration of the small tree state.

Quantitative Evaluation

[00377] This section presents a quantitative evaluation of the methods discussed above. The evaluation is split into three different parts. First, all methods are compared against a ground truth dataset, giving an estimate of the relative performance of the different methods. Furthermore, this part focuses on the performance with regard to segmentation errors and detection of trees, not the accuracy of the borders between trees.

[00378] Secondly, a number of the best performing methods are evaluated visually on a larger dataset. This evaluation is performed both with regard to segmentation errors as well as inaccurate border placement between trees.

[00379] Finally, the segmentation methods' robustness is evaluated on a dataset distorted by the collection process.

Method Comparison

[00380] This section evaluates the performance of all methods discussed above on the 4 shorter and the 3 empty datasets described above.

[00381] The evaluation performed on the empty datasets is used to determine how susceptible the different segmentation methods are to creating false trees. However, as described in the "ground removal" section, the ground structure is slightly different from the ground inside the rows. Nevertheless, we consider the ground similar enough to give an estimate of the susceptibility to creation of false trees.

[00382] The evaluation on the shorter datasets is done by applying the evaluated method on the dataset and comparing the resulting trees to ground truth data. The ground truth data is created by using the hand-tuned volume 4 state model and calculating the centres of the segmented trees. Furthermore, the ground truth segmentation was visually inspected to ensure that it was entirely correct. The datasets were then segmented using the evaluated methods and the results compared to the ground truth data.

[00383] For both dataset types, the resulting segmentations were counted according to:

- True Positives (TP) - The number of centres found within 2 meters of a ground truth centre.
- Double Positives (DP) - The number of times more than one centre is found within 2 meters of a ground truth centre
- False Positives (FP) - The number of times a tree is found without there being a ground truth centre within 2 meters
- False Negatives (FN) - The number of ground truth centres where no tree is found within 2 meters.

[00384] There are two reasons for using the relatively large matching distance of 2 meters. First, there is navigational inaccuracy between the different datasets, meaning that the centre of one tree in one dataset

is slightly different in another. Secondly, the large matching distance allows the evaluation to focus on large segmentation errors, while we focus on evaluating the segmentation accuracy in the “multiple row robustness” section below.

Hand-Tuned Volume Distributions

[00385] The results obtained using the hand-tuned volume observation distributions are presented in table 4.1. Visual inspection shows that all large trees are found and correctly segmented, and that all false negatives originate from the small trees being labeled as gaps. Furthermore, all false positives are from false trees in the empty datasets.

[00386] The results show that the addition of a fourth state clearly improves the performance with regard to the small trees. Furthermore, though ground removal is not necessary to properly segment the small trees, it is necessary to avoid false trees being created in the empty areas outside the rows.

Method	TP	DP	FP	FN
3 state with Ground	440	0	0	16
3 state without Ground	443	0	0	13
4 state with Ground	456	0	2	0
4 state without Ground	456	0	0	0

Table 4.1: The performance of the segmentation method using hand-tuned volume distributions on the shorter and empty datasets.

Learnt Volume Distributions

[00387] The results obtained using learnt volume and duration distributions are presented in table 4.2. The results show that the method is on par with the equivalent hand- tuned method.

Method	TP	DP	FP	FN
3 state with Ground	440	0	0	16
3 state without Ground	444	0	0	12
4 state with Ground	456	0	2	0
4 state without Ground	456	0	0	0

Table 4.2: The performance of the segmentation method using learnt volume distributions on the shorter and empty datasets.

Learnt Density Distributions

[00388] The results obtained using learnt density and duration distributions are presented in table 4.3. The results show that the method is worse when ground has not been removed, assumedly because of variations in the number of points in the ground.

[00389] The method performs better after ground has been removed, however it is still considerably worse than the volume feature.

Method	TP	DP	FP	FN
3 state with Ground	435	0	6	21
3 state without Ground	442	0	0	14
4 state with Ground	441	0	6	15
4 state without Ground	456	0	0	0

Table 4.3: The performance of the segmentation method using learnt density distributions on the shorter and empty datasets.

Hand-Tuned Height Distributions

[00390] The results obtained using hand-tuned height distributions are presented in table 4.4. The results show that the height feature works well when the accuracy demand of the segmentation is low and ground has not been removed.

Method	TP	DP	FP	FN
3 state with Ground	452	0	0	4
3 state without Ground	455	0	0	1
4 state with Ground	456	0	0	0
4 state without Ground	445	0	6	11

Table 4.4: The performance of the segmentation method using hand-tuned height distributions on the shorter and empty datasets.

Learnt Height Distributions

[00391] The results obtained using learnt height distributions are presented in table 4.5. The results show that learnt height observations works slightly better than the hand-tuned equivalent, assumedly because of bad tuning of the latter.

Method	TP	DP	FP	FN
3 state with Ground	456	0	0	0
3 state without Ground	440	0	0	16
4 state with Ground	456	0	0	0
4 state without Ground	456	0	0	0

Table 4.5: The performance of the segmentation method using learnt height distributions on the shorter and empty datasets.

Combined Hand-tuned Volume and Height Distributions

[00392] The results obtained using hand-tuned volume and height distributions are presented in table 4.6. The results show that the effects of the ground removal seem to be rather non-intuitive; It improves the performance with regard to small trees for the 3 state model but decreasing the performance for the 4

state model. This non-intuitive behaviour is assumed to be due to all points being removed in some slices but not in others, causing the height to vary even in areas where the ground is flat. Nevertheless, both the 3 state model without ground and the 4 state model with ground yield perfect results.

Method	TP	DP	FP	FN
3 state with Ground	440	0	0	16
3 state without Ground	456	0	0	0
4 state with Ground	456	0	0	0
4 state without Ground	454	0	0	2

Table 4.6: The performance of the segmentation method using hand-tuned volume and height distributions on the shorter and empty datasets.

Combined Learnt Volume and Height Distributions

[00393] The results obtained using learnt volume and height distributions are presented in table 4.7.

Notably, these results show that the 3 state model properly detects all small trees without ground removal, something which the equivalent hand- tuned method failed to do. This difference is assumed to be due to bad tuning of the hand-tuned observation distributions. Furthermore, the performance of the 4 state model is similar to the performance obtained with the equivalent hand- tuned method.

Method	TP	DP	FP	FN
3 state with Ground	456	0	0	0
3 state without Ground	440	0	0	16
4 state with Ground	456	0	0	0
4 state without Ground	456	0	0	0

Table 4.7: The performance of the segmentation method using learnt volume and height distributions on the shorter and empty datasets.

[00394] The performed evaluation shows that using only volume or using a combination of height and volume features both yield good results. The former however demands the ground-removal pre-processing step in order to avoid creation of false trees outside the rows while the latter however performs best when ground has not been removed. Moreover, as the evaluation is not focused on the accuracy of the segmentation, the height feature also performs well.

Multiple Row Robustness

[00395] This section evaluates the performance of a limited number of methods on the larger dataset described above. The aim of the evaluation is both to determine the performance on a larger dataset as well as to get an estimate of the segmentation accuracy of the different methods. The evaluation is performed by visually inspecting the segmented dataset and counting the resulting:

- True Positives (TP) - Trees labeled as trees.
 - False Positives (FP) - Gaps labeled as trees
 - False Negatives (FN) - Trees labeled as gaps
 - Merge Errors (ME) - Segmentation errors where two trees have been merged into one.
 - Border Errors (BE) - A segmentation where the border has been incorrectly placed between two trees.
- As the evaluation focuses on the segmentation accuracy, we count all errors, even small ones, where it would have been possible to better place the border using the human eye.

[00396] The evaluated methods and the evaluation results are presented in table 4.8. These results show that the volume feature performs slightly better without ground removal. Furthermore combined hand-tuned observation distributions used with a 4 state model yields equally good results. Additionally, the same results can be achieved using combined learnt observations with a 3 state model. Furthermore, it should be mentioned that the border errors when using these methods are small, being of the scale seen in figure 41. Figure 41 shows one of three border errors encountered when segmenting the longer dataset using learnt volume distributions. Note that the border between the second and third tree is slightly misplaced.

Method	TP	FP	FN	ME	BE
Hand-Tuned Volume 4 State With Ground	580	0	0	0	2
Learnt Volume 4 State With Ground	580	0	0	0	2
Hand-Tuned Volume 4 State Without Ground	579	0	0	1	2
Learnt Volume 4 State Without Ground	579	0	0	1	2
Hand-Tuned Combined 3 State With Ground	573	0	7	0	1
Learnt Combined 3 State With Ground	580	0	0	0	2
Hand-Tuned Combined 4 State With Ground	580	0	0	0	2
Learnt Combined 4 State With Ground	577	0	0	3	3

Table 4.8: The performance of the segmentation methods applied on the larger dataset (TP - true positives, FP - false positives, FN - false negatives, SE - segmentation error)

Data Collection Robustness

[00397] The evaluation presented in this section aims to evaluate the segmentation methods robustness to the data collection process. The evaluation is performed in the same way as in the previous section, the difference being that the distorted dataset presented above is used. An excerpt of the dataset is seen in figure 42. Figure 42 shows an excerpt of the dataset collected by the robot moving in a sinusoidal motion.

[00398] The evaluation results are presented in table 4.9, showing that the dataset can be segmented correctly despite the distortions. Furthermore, none of the examined methods appear to yield better results than the others.

[00399] It could be noted however, that even though the data can be correctly segmented, it may still be too distorted to be of any use for characterisation. As an example of this, figure 43 shows the volume of individual trees calculated from the four shorter datasets as well as from the distorted dataset. Figure 43 shows estimated volumes of trees when the data has been collected by a robot moving in a straight line 4301 and in a sinusoidal curve 4303.

Method	TP	FP	FN	ME	BE
Hand-Tuned Volume 4 State With Ground	106	0	0	0	2
Learnt Volume 4 State With Ground	106	0	0	0	2
Hand-Tuned Volume 4 State Without Ground	106	0	0	0	2
Learnt Volume 4 State Without Ground	106	0	0	0	2
Hand-Tuned Combined 3 State With Ground	104	0	2	0	2
Learnt Combined 3 State With Ground	106	0	0	0	2
Hand-Tuned Combined 4 State With Ground	106	0	0	0	2
Learnt Combined 4 State With Ground	106	0	0	0	2

Table 4.9: The performance of the segmentation methods when applied on the distorted dataset. (TP - true positives, FP - false positives, FN - false negatives, SE - segmentation error)

[00400] Evaluating the resulting methods, it has been shown that a combination of volume and height features yields the best results. Using a 4 state model and hand-tuned observation distributions, it can both accurately segment large trees as well as detect small trees. Furthermore, the method does not create any false trees, neither inside nor outside the rows. Furthermore, similar results can be obtained using a 3 state model with learnt volume and height distributions.

[00401] Additionally, if the area outside the rows is not taken into consideration, using only the volume feature and a 4 state model yields similar results. Furthermore, this method yields equivalent results both when using learnt and hand-tuned observation distributions. Finally, if the ground removal pre-processing is used, the method does not yield any false trees outside the rows.

Characterisation

[00402] As mentioned earlier, one goal is to utilise the segmented trees to perform GPS independent localisation. A key step in this process is to be able to tell the different trees from each other. In order to address this, descriptors are introduced to describe the characteristics of the trees.

[00403] A simplifying factor when performing localisation in an orchard is that the trees are aligned into rows, resulting in all trees appearing in sequences. This simplifies the localisation as it allows the localisation to depend on sequence matching instead of one-to-one matching. This is important as many of the trees are individually very similar. Furthermore, the trees are both sparsely and non- uniformly sampled, further limiting the usable information that can be extracted from them. The use of sequence matching allows this lack of information in the individual trees to be compensated by combining the information from multiple trees.

[00404] Two different descriptor types are introduced based on the features described above denoted as the simple and signature descriptors. Thereafter, a method is described for performing localisation in a map built from the introduced descriptors. Using this localisation method, the informativeness of the different descriptors is evaluated. Furthermore, the possibility of building the map both from a single descriptor of each tree as well as by combining multiple tree descriptors is described.

Descriptors

[00405] In order to characterise the segmented trees, it is necessary to use a descriptor to describe the 3D data. A common trait for many existing 3D descriptors, e.g. spin-images, shape contexts and point feature histograms, is that they describe the local neighbourhood with regard to an estimated normal direction. However, since the dataset is sparsely sampled, these descriptors are considered to be unlikely to yield good results.

[00406] Another common set of descriptors, including shape distributions and shape functions, are based on the distribution of points. However, these may also be problematic as it depends on the sampling of points being consistent. This is not true in the utilised datasets, as uneven speed or a slight turn result in varying sampling densities. It will be understood that such problems can be negated by adequate re-sampling.

[00407] Two new descriptor types have been described based on the features discussed above. The first descriptor type, denoted the simple descriptor, is a single feature value describing the tree. The two simple descriptors examined are the simple volume and simple height descriptors. The second descriptor type examined, the signature descriptor, is slightly more complex, storing a sequence of features for each tree.

Simple Volume Descriptor

[00408] The simple volume descriptor stores the estimated volume of the tree, calculated as described above. In order to estimate the informativeness and consistency of the descriptor, it is plotted for four different datasets in figure 44. Figure 44 shows the volume calculated over the same row seen four times. The figure shows that the volume is both fairly consistent between runs and that it is also different between trees, therefore being both informative and consistent.

Simple Height Descriptor

[00409] The simple height descriptor stores the maximum height of the tree. Plotting the descriptor for four different datasets yields figure 45. Figure 45 shows the height calculated over the same row seen four times. The figure shows that the descriptor is both consistent and informative, while also appearing to be slightly less noisy than the volume descriptor.

Volume Signature Descriptor

[00410] The volume signature descriptor stores the estimated volume of each 0.2m slice of the tree. A rough estimate of the usefulness of the descriptor was obtained by plotting the descriptor of the same tree

for multiple datasets. A typical example of this is presented in figure 46, showing that the signature is rather noisy. Figure 46 shows the volume signature of a tree seen four times from the same side.

Height Signature Descriptor

[00411] The height signature descriptors stores the maximum height of each 0.2m slice of the tree. An example of the descriptor plotted for the same tree over multiple datasets can be seen in figure 47. Figure 47 shows the height signature of a tree seen 4 times. The noise level appears to be on par with the volume descriptor, with possibly slightly less noise.

[00412] Figures 67-74 shows a number of descriptor similarity distributions.

[00413] Figures 67-70 show how 4 shorter datasets were utilized to calculate the self-similarity distribution of the same tree seen again from the same side.

[00414] Figures 71-74 show the comparison of the descriptor of each tree compared to the descriptor of all other trees in all datasets.

Comparing Descriptors

[00415] In order to use the descriptors described herein it is necessary to define how two descriptors of the same type are compared. This is done here, both for the simple descriptor and the signature descriptor.

[00416] Additionally, it should be mentioned that having defined how to compare two descriptors, it is possible to calculate the self-similarity distributions of the descriptors. These are described below and are utilised in the localisation algorithm discussed below.

Simple Descriptor

[00417] The difference, δ_t , between two simple descriptors is calculated as

$$\delta_t(d_1, d_2) = |d_1 - d_2| \quad (5.1)$$

where d_1 and d_2 are simple descriptors.

Signature Descriptor

[00418] In order to calculate the difference, δ_t , between two signature descriptors it is necessary to take into consideration that the signatures may be of different length. This problem is approached by finding the best fit between the two signatures and calculating the difference for this fit.

[00419] To begin with, denote the longer sequence d_l and the shorter sequence d_s , the difference in length between them being N . In order to find the best match between the signatures, we offset d_s for $N + 1$ different steps and determine which signature which is least different. However, it is also necessary to

zero pad the offset vector such that it is of the same length as d_i . Therefore this zero padded and offset vector are defined as:

$$d_i^p(i) = [\text{zeros}(i) \ d_i \ \text{zeros}(N - i)] \quad (5.2)$$

where $\text{zeros}(k)$ is a vector of length k consisting of zeros. The difference between d_i and $d_i^p(i)$ is calculated using the $L1$ norm of the difference between the two vectors:

$$\delta_i^l(d_i, d_s) = \|d_i - d_i^p(i)\|_1. \quad (5.3)$$

The difference between two signatures is then calculated as

$$\delta_s(d_i, d_s) = \min_i \delta_i^l(d_i, d_s). \quad (5.4)$$

Localisation Method

[00420] The idea of the localisation method is to match a sequence from one dataset against all sequences in a map of the orchard. The location of the current sequence is then obtained by finding the best matching sequence in the map.

[00421] To begin with, a tree in the dataset to be localised is chosen. Assuming a sequence length of N , the descriptor of the tree is combined with the descriptors of the $N - 1$ previous trees to form a sequence of descriptors. The same procedure is performed for all trees, exempting the $N - 1$ first, in the map. The difference between the individual sequences is then calculated using the $L1$ -norm as:

$$\delta_s(s_1, s_2) = \sum_{n=1}^N |\delta_i(s_1^n, s_2^n)|, \quad (5.5)$$

where s_i^n denotes element n of sequence s_i . The best match when matching sequence s against K other sequences is then defined as

$$s^* = \arg \min_{1 \leq j \leq K} \delta_s(s, s_j), \quad (5.6)$$

thus giving the matching position of the tree in the map.

Maps From Single Descriptors

[00422] The performance is evaluated when performing localisation on a map created from a single descriptor of each tree. Localisation is performed in three different situations. In the first situation, localisation is performed on a single row, where the compared descriptors have been seen from the same side. Secondly, the same localisation is performed but on multiple rows. Finally, localisation is performed on a single row but instead match against descriptors seen from the other side of the row. The matching is performed for all possible trees in the compared datasets and the ratio of correct matches calculated.

Same Side Single Row Matching

[00423] In this method, the 4 shorter datasets are utilised to perform 4-time cross validation of the localisation method. In order to avoid any problems with ambiguities, the matching of the trees is kept separate for the two different sides, i.e. a tree in the first dataset seen from one side is only matched against trees seen from the same side.

[00424] The results when using the four different descriptors discussed above are displayed in table 5.1. The results show that the simple descriptors perform significantly worse than the signature descriptors, a result which is expected given their limited amount of information. Furthermore, the two height descriptors appear to be slightly better than their volume counterparts.

Length	Simple V	Simple H	V Signature	H Signature
1	13.56	21.93	77.35	86.36
3	61.66	71.00	97.16	99.78
5	85.19	89.46	99.85	100.0
10	99.74	99.57	100.0	100.0
20	100.0	100.0	100.0	100.0

Table 5.1: The ratio of correct matches in percent for the four different descriptors when matching on a single row.

Same Side Multiple Row Matching

[00425] In this method the larger dataset was used in combination with the 3 smaller datasets. The localisation was done by matching the trees from the smaller dataset with the larger dataset. In order to avoid ambiguities, the same row seen from the other side was removed from the larger dataset during the matching.

[00426] The results of the matching are presented in table 5.2, showing that it is necessary to utilise a longer sequence when matching against a larger dataset. Furthermore, the performance of the signature descriptors is considerably better than the simple descriptors. Finally, the height variants of the descriptors appear to perform slightly better than their volume counterparts.

Length	Simple V	Simple H	V Signature	H Signature
1	4.03	10.09	66.57	79.25
3	42.98	53.43	94.03	100.0
5	75.23	82.04	99.38	100.0
10	92.83	98.29	100.0	100.0
20	100.0	100.0	100.0	100.0

Table 5.2: The ratio of correct matches in percent for the four different descriptors when matching over multiple rows.

Opposite Side Single Row Matching

[00427] In this method, we perform localisation on a single row against a map where the trees have only been seen from the other side. The evaluation is performed using 4-time cross validation, the results being presented in table 5.3. These results show that the performance of the matching is significantly worse compared to when matching against descriptors from the same side. Nevertheless, it also shows that it is possible to perform localisation in this situation given a long enough tree sequence.

Length	Simple V	Simple H	V Signature	H Signature
1	8.06	13.04	21.59	47.67
3	36.27	44.41	59.92	91.26
5	53.95	64.77	81.98	98.90
10	87.76	92.89	97.63	100.0
20	100.0	100.0	100.0	100.0

Table 5.3: The ratio of correct matches when matching a tree with descriptors seen from the other side in a single row.

Maps From Multiple Descriptors

[00428] It is examined whether better results are achieved by utilising multiple descriptors for each tree. In order to merge multiple descriptors, two different methods are described.

[00429] When using simple descriptors, it is possible to average the descriptors' values. Assuming that the mean of the measurement error is zero, the averaged value should be closer to the correct value, thus giving that the performance of the matching should improve.

[00430] This approach however, is not applicable for signature descriptors, as the signatures may be of different lengths. Instead the multiple instances of the descriptors are stored, and the compared descriptor is matched against all stored descriptors. The difference between the tree and compared descriptor is defined as the difference between the two best matching descriptors.

[00431] These approaches of combining multiple measurements are investigated in two different situations. In the first situation, the trees are seen multiple times from the same side and their measurements combined. In the second situation, the trees are seen once from each side and their

measurements combined. In order to determine if this usage of multiple measurements increases the localisation performance, single row localisation is performed and the results compared to the results obtained during same side single raw matching.

Descriptors From The Same Side

[00432] In this situation, multiple descriptors of the same tree seen multiple times from the same side are combined to perform localisation on a single row. The method is evaluated by using 3 of the shorter datasets to build a map and 1 dataset to perform the matching. Furthermore, this is done four times in order to perform 4-time cross validation.

[00433] The performance of averaging the simple descriptors are presented in table 5.4, showing a slight increase in performance compared to table 5.1.

Length	Simple V	Simple H
1	14.04	21.60
3	66.44	75.39
5	92.58	93.74
10	100.0	100.0
20	100.0	100.0

Table 5.4: The ratio of correct matches in percent when averaging the simple descriptors seen from the same side.

[00434] The results when storing multiple descriptors are presented in table 5.5. These results show only a very slight increase in performance when storing multiple simple descriptors. However, there is a notable increase in performance when storing multiple signature descriptors.

Length	Simple V	Simple H	V Signature	H Signature
1	13.61	22.68	87.26	93.95
3	66.22	77.63	98.66	100.0
5	89.80	94.43	100.0	100.0
10	100.0	100.0	100.0	100.0
20	100.0	100.0	100.0	100.0

Table 5.5: The ratio of correct matches in percent when storing multiple descriptors seen from the same side.

[00435] Based on these results, it is concluded that a distinct increase in matching performance can be obtained by using multiple descriptors. When using simple descriptors, the best results are obtained by averaging them, while it is necessary to store multiple descriptors when using signature descriptors.

Descriptors From The Opposite Side

[00436] In this situation, multiple descriptors of the same tree seen from different sides are combined to perform localisation on a single row. The map is created by choosing one of the shorter datasets to

create a map and combining the two descriptors of the tree. The three other datasets are then matched against this map. Furthermore, this is done four times using 4-time cross validation.

[00437] The results when averaging the simple descriptors are presented in table 5.6, showing a decrease in performance compared to table 5.1.

Length	Simple V	Simple H
1	11.66	14.83
3	53.17	58.32
5	76.18	83.53
10	99.40	98.64
20	100.0	100.0

Table 5.6: The ratio of correct matches in percent when averaging the simple descriptors seen from the opposite side.

[00438] Table 5.7 shows the results when multiple instances of the descriptors are stored. These results show a decrease in performance for the simple descriptors while the performance of the signature descriptors appear to be unaffected by the additional descriptor.

Length	Simple V	Simple H	V Signature	H Signature
1	10.44	17.71	74.51	86.25
3	53.17	66.07	98.06	99.93
5	80.28	88.87	99.69	100.0
10	99.49	99.15	100.0	100.0
20	100.0	100.0	100.0	100.0

Table 5.7: The ratio of correct matches in percent when storing multiple descriptors seen from opposite sides.

[00439] Based on these results, it may be preferable to keep separate descriptors for each side of the row rather than combining them.

[00440] It has been shown how descriptors may be utilised to perform localisation in the orchard, both on a single row and over multiple rows. It has also been shown that it is possible to perform localisation even when the trees have previously only been seen from the other side of the row.

[00441] Additionally, the change in performance when using multiple descriptors of each tree to create a map has been described. This increases the localisation performance when descriptors seen from the same side are combined. However, combining descriptors seen from different sides decreases the localisation performance. Furthermore, when utilising the simple descriptors the best results are obtained by averaging them. For the signature descriptors however it is necessary to store the multiple descriptors in the map.

[00442] Finally, based on all examined localisation and map building methods, it has been shown that the height signature descriptor yields the best localisation results.

Localisation

[00443] The localisation method described above was dependent on no segmentation errors having occurred, an assumption that may not always be applicable. In order to resolve this, a HMM based localisation method is described that properly handles segmentation errors. Furthermore, multiple variants of the localisation method are proposed in order to handle different localisation scenarios. Finally, the performance and robustness of the presented localisation methods are described using both the datasets from the orchard as well as randomly generated datasets. Usage scenarios and how localisation may be employed in these instances is first described.

Usage

[00444] In order to perform localisation, a map of the orchard must first be created. This can be done by using the segmentation and characterisation methods discussed above, resulting in a map where each tree is represented by a descriptor. Given that such a map exists, the problem lies in matching a new sequence of observed trees, collected during another survey of the orchard, with the corresponding trees in the map.

[00445] There are two distinct uses of the localisation. The first one is to find the current position in the map based on the trees that have been observed so far, thus enabling autonomous localisation in the orchard without the use of a GPS system. This localisation is denoted as online localisation, as it is possible to employ in an online situation.

[00446] The second use is to observe a part of the orchard and batch process all observations in order to match them with the map. This makes it possible to associate observations from a new survey with the trees in the original map, allowing the change of the individual trees to be monitored. Furthermore, since it does not depend on any high performance GPS system, it makes it both easier and cheaper to carry out surveys. As this localisation can only be performed after all data has been collected, the method is denoted as offline localisation.

[00447] Another point to consider is what additional information that is available regarding the order that the trees are observed in. On one extreme, no additional information is available; the next tree may be any tree. However, given that the segmentation is reasonably ok, it can be assumed to be one of the nearby trees. On the other extreme, the odometry of the vehicle may be used to give a qualified estimate of the next tree. Two methods are described, both bordering the former extreme. The first method, denoted the undirected method, assumes no knowledge of which direction the next tree is observed, however it is assumed to be one of the nearby trees. The second method, denoted the directed method, assumes knowledge of which direction the next tree is observed, thus slightly limiting the problem.

[00448] In both online and offline situations, knowing the transition direction is equivalent to knowing the movement direction of the robot. Since it is only necessary to determine the sign of the bidirectional movement in a global reference frame, this can be done using only a compass and information of the vehicle's forwards-backwards direction. Furthermore, if it is assumed that the vehicle does not change direction in the middle of a row, the offline situation allows the problem to be solved by ordering the observations in all rows according to a global reference frame, e.g. defining that the beginning of a row is always in the southern end of the row.

Localisation Method

[00449] The core idea of the localisation method is to model the orchard as a Hidden Markov Model with each state representing an individual tree seen from a specific side. Furthermore, a height signature descriptor, described above, is stored for each state. Similarly, a descriptor is calculated for each newly observed tree, allowing the map descriptors to be compared to the observed descriptors.

[00450] The transition matrix is used to model the order that the trees are observed in. Furthermore, the possibility of segmentation errors is integrated by allowing transitions to non-adjacent states. In addition, two different variants of the transition matrix is presented, allowing both for directed and undirected localisation. Finally, the matching is done using either the Forwards or Forwards-Backwards algorithm depending on whether online or offline localisation is performed.

Transition Matrix

[00451] The transition matrix models the order that the trees are observed in. In order to handle segmentation errors, it is necessary to allow for transitions to non-adjacent trees. Furthermore, row-endings are explicitly modeled to allow for transitions both to other rows and to the same row. Additionally, two different transition matrices are presented; a directed transition matrix, where the direction to the next observation is known, and an undirected transition matrix, where the direction to the next observation is unknown.

Directed Transition Matrix

[00452] To begin with, denote the number of states as N , thus giving a transition matrix of size $N \times N$. If the segmentation is correct, the transitions will always be from one state to the next. If it is assumed that the localisation is only performed on one row, this could be represented as a transition matrix A where the individual elements are defined as

$$A_{ij} = \begin{cases} 1 & , \text{if } j = i + 1 \\ 0 & , \text{else} \end{cases} \quad (6.1)$$

[00453] where element A_{ij} denotes the probability of transitioning from state i to state j . It is however necessary to incorporate the possibility of segmentation errors, thus allowing both for self-transitions as well as transitions to non-adjacent trees. Theoretically, the optimal model should match the actual probabilities of segmentation failures in the orchard. However, since these probabilities are unknown, the transition matrix is designed to allow for larger segmentation errors than is to be expected. An initial version of the transition matrix is given as

$$\bar{A}_{ij} = \begin{cases} 0.10 & , \text{ if } j = i + 0 \\ 0.70 & , \text{ if } j = i + 1 \\ 0.10 & , \text{ if } j = i + 2 \\ 0.05 & , \text{ if } j = i + 3 \\ 0.05 & , \text{ if } j = i + 4 \\ 0 & , \text{ else} \end{cases} \quad (6.2)$$

However, the above transition matrix is not correct as

$$\sum_j \bar{A}_{ij} \neq 1 \text{ if } i > N - 4. \quad (6.3)$$

Furthermore, it does not handle transitions between different rows.

In order to model the transitions between different rows, transition elements are managed between trees belonging to different rows separately. First, introduce

$$N_{step} = j - i, \quad (6.4)$$

denoting the length of a transition. Secondly, introduce N_{start} to mark the first tree in a row and N_{end} to mark the last tree in a row. Furthermore, introduce

$$N_{inside\ end} = N_{end} - j \quad (6.5)$$

to represent how far j is inside a row with regard to its end, and

$$N_{inside\ start} = j - N_{start} \quad (6.6)$$

to represent how far j is inside a row with regard to its start.

To begin with, update all transition elements between trees in the same row according to (6.2). In the case that the two trees are not in the same row, i.e. $N_{inside\ end} < 0$, denote

$$N_{outside} = -N_{inside\ end}. \quad (6.7)$$

Finally, the value corresponding to N_{step} is extracted from (6.2), however it is equally split among all trees where

$$N_{inside\ start} + 1 = N_{outside}, \quad (6.8)$$

allowing for transitions from one row to any other row, and itself, with equal probability.

Undirected Transition Matrix

[00454] In the case that the direction of the robot is not known, it is necessary to allow transitions in both directions. This can be achieved by creating an initial transition matrix by

$$\tilde{A}_{ij} = \begin{cases} 0.025 & , \text{if } j = i - 4 \\ 0.025 & , \text{if } j = i - 3 \\ 0.05 & , \text{if } j = i - 2 \\ 0.35 & , \text{if } j = i - 1 \\ 0.10 & , \text{if } j = i + 0 \\ 0.35 & , \text{if } j = i + 1 \\ 0.05 & , \text{if } j = i + 2 \\ 0.025 & , \text{if } j = i + 3 \\ 0.025 & , \text{if } j = i + 4 \\ 0 & , \text{else} \end{cases} \quad (6.9)$$

As for the directed transition matrix, it is necessary to manage transitions between different rows. Using the same notations, a transition between trees in two different rows means that either $N_{inside\ start} < 0$ or $N_{inside\ end} < 0$. Depending on which of the two situations, denote $N_{outside}$ as either

$$N_{inside\ start} + 1 = N_{outside} \quad (6.12)$$

and where

$$N_{inside\ end} + 1 = N_{outside} \quad (6.13)$$

$$N_{outside} = -N_{inside\ start} \quad (6.10)$$

or

$$N_{outside} = -N_{inside\ end} \quad (6.11)$$

As for the directed transition matrix, the value corresponding to N_{step} is extracted from (6.9), however it is equally split both among all trees where

$$N_{inside\ start} + 1 = N_{outside} \quad (6.12)$$

and where

$$N_{inside\ end} + 1 = N_{outside} \quad (6.13)$$

Initial Probabilities

[00455] As the initial position in the orchard is unknown, the initial probabilities are set as

$$\pi_i = 1/N \quad \forall i. \quad (6.14)$$

Observation Distributions

[00456] For all observed trees, the corresponding height signature descriptor is used, as observation feature. To compare the descriptors to those in the map, the differences between these observed descriptors and those representing the states are calculated as described above. As a means to calculate the likelihood of the state from this difference, the self-similarity distribution shown in Fig. 60 is used. One problem however is that the self-similarity distribution is not monotonic. Using the self-similarity distribution directly would thus result in two very similar descriptors yielding a moderately large likelihood. Conversely, two descriptors with moderately similar descriptors would result in a large likelihood. A visual representation of the problem is presented in figure 48. Fig. 48 shows an example of a non monotonic self-similarity distribution 4801. Note that the likelihood of a medium sized error 4803 is larger than the likelihood of a small error 4805.

[00457] In order to avoid this counter-intuitive behaviour, we define the likelihood of the state to be the probability of observing a difference larger than the measured difference, visualised in figure 49. To be noted is that this is equivalent to defining the likelihood as the complementary cumulative distribution function. Fig. 49 shows an example of a self-similarity distribution 4901. The probability of a difference being larger than a certain value 4903 has been marked.

[00458] The resulting likelihood function based on the self-similarity distribution in Fig. 60 is presented in figure 50. Fig. 50 shows the complementary cumulative difference distribution of the height signature descriptor.

Matching Algorithm

[00459] Matching is performed using the optimisation algorithms described above. In the case of online localisation, the Forwards algorithm is employed. Similarly, the Forwards-Backwards algorithm is employed for offline localisation. Furthermore, in the case that the observed trees are not in the same order as in the map, but the direction between each observation is known, the used optimisation algorithm may be altered to integrate this knowledge into the optimisation. This direction based localization is explained in detail below.

[00460] It should be noted that it is possible to make use of the Viterbi algorithm instead of the Forwards and Forwards-Backwards algorithm.

Randomised Datasets

[00461] The performance of the localisation method is partly evaluated on a set of randomised datasets, created by perturbing the original datasets presented above.

[00462] First of all, noise is added to the height feature measured at each slice. Thereafter the segmentation is altered in order to simulate an imperfect segmentation. Initially, this is done by introducing 1 noise to the border positions. This does not represent an actual segmentation error but simply the impreciseness of the segmentation. In order to simulate segmentation errors, some pairs of trees are merged into one, some trees are split in half, and some are labelled as gaps.

[00463] In order for the evaluation to be valid, it is imperative that the magnitude of the noise, both with regard to feature measurements and border positions, are based on the equivalent observed levels within the data. The following sections describe how this is measured.

Feature Noise

[00464] In order to estimate the distribution of the noise from the data, measurements from the four shorter datasets were aligned and the differences between the measurements in the same slice were calculated. Due to inaccuracy in the navigational system, the measurements become unaligned after a certain distance, demanding the datasets to be aligned again after approximately 50 to 100 observations. Calculating the measurement differences using 1000 measurements from each dataset yielded the difference distribution presented in Fig. 51 (which shows the estimated height noise distribution), showing that the distribution appears to be fairly Gaussian. Based on this notation, it was decided to model the observation noise as Gaussian with mean 0 and a standard deviation of 0.25 meter estimated from the difference distribution.

[00465] In order to introduce the feature noise into the randomised datasets, a value is drawn from the distribution at each slice and added to the measurement of the original dataset at this slice.

Border Noise

[00466] In order to estimate the distribution of the border noise, the durations of the same trees in different datasets were compared. The comparison was done by choosing one tree in one dataset and then calculating the difference between the duration in the chosen dataset and the durations in the other datasets. Doing this for all trees and datasets yields the distribution presented in figure 52. The figure shows that the estimated duration noise distribution appears to be Gaussian.

[00467] In order to calculate the border noise distribution from the duration noise distribution it is necessary to make some assumptions. First it is assumed that the duration noise distribution is indeed Gaussian. Secondly, it is noted that the duration noise is due to movement in the two borders delimiting the tree. Assuming that the two delimiting borders move independently, which is not entirely true given the effects of the duration distribution, it follows that the border noise distributions are independent Gaussians. This gives that the standard deviation of the border noise distribution and the duration noise distribution are related by

$$\sigma_{duration} = \sqrt{2} \cdot \sigma_{noise}, \quad (6.15)$$

[00468] allowing the border noise distribution to be calculated from the estimated duration noise distribution. Utilising this relation gives that an appropriate model of the border noise is a Gaussian with zero mean and a standard deviation of 1.0 slices.

[00469] In order to introduce the border noise into the randomised datasets, a value is drawn from the distribution at each border in the original dataset. The drawn value is then rounded to the nearest integer and the border position moved by this many steps.

[00470] The correctness of the feature and border noise modelling can be estimated by calculating the self-similarity distributions from the randomised dataset and comparing it to the self-similarity distribution estimated from the datasets. Both these distributions are presented in figures 53 and 54, showing that the distributions are rather different. Fig. 53 shows the self-similarity histogram estimated from real measurements. Fig. 54 shows the self-similarity histogram estimated from simulated measurements. Furthermore, the simulated self-similarity distribution has a notable larger mean error. Nevertheless, we consider the distributions to be similar enough to accept the performed noise modelling.

Merge Errors

[00471] In order to simulate trees being incorrectly merged during segmentation, merge errors are introduced into the randomised datasets. This error is introduced by removing the border between two trees. Furthermore, the merging algorithm also ensures that no more than two trees are merged together.

Split Errors

[00472] A possible segmentation error is that some trees are split into two during segmentation. In order to model this, a border is introduced in the middle of the tree. Furthermore the splitting algorithms ensures that only trees with more than 3 observations are split, thus avoiding impossible splitting of very small trees.

Detection Errors

[00473] A possible error during segmentation is that some trees are labelled as gaps. In a real application, the probability of this happening is negligible for all but the small trees. Nevertheless, the effect of the error is investigated both when large and small trees are labelled as gaps. In order to introduce the error into the randomised dataset, the state of a tree is simply changed to gap.

Evaluation

[00474] The performance of the localisation method was evaluated by matching a sequence of observations from one dataset against a map created from another dataset. For each such matching, the ratio of correctly matched trees was calculated. As the counting of correct matches is non-trivial after

segmentation errors have been added, the counting is explained in detail above. These matching tests were performed both for the original datasets as well as the randomised datasets, thus giving an estimate of both the performance in the current application as well as the robustness to noise and segmentation errors.

[00475] The localisation was evaluated both when performing localisation on a single row as well as on multiple rows. Furthermore, the different variants of the localisation method were separately evaluated, allowing their performance to be compared. Finally, it should be noted that each test was done multiple times, as to ensure that the estimated performance was based on the matching of at least 10000 trees.

Counting Correct Matches

[00476] In the performed evaluation, the criteria for a match being considered correct is defined as:

- A correctly segmented tree is correctly matched when the matched tree is the same as the original tree.
- For a split tree, the correct match for both new trees is the original tree
- For a merged tree, a match is considered to be correct if the new tree is matched with either of the two original trees.
- Trees labelled as gaps are not part of the matching process and yield neither correct nor incorrect matches.

Evaluation Setup

[00477] The evaluation was performed both using the randomised datasets and the original datasets. When evaluating the performance using the randomised datasets, the performance was evaluated by changing one parameter while keeping the other parameters fixed.

[00478] In order to simulate the measurement noise, the fixed value of the feature noise standard deviation was set to 0.25 meters, the value calculated above in the "feature noise" section. Similarly, the fixed value of the border noise standard deviation was set to 1.0 slices as described above in the "border noise" section. In order to accommodate for possible segmentation errors, the fixed probability of any specific segmentation error occurring was set to 2%. This meant that approximately 5% of all trees were affected by segmentation errors, a larger ratio than encountered in the existing datasets.

[00479] When matching single randomised rows, an original dataset was used to create the map while a corresponding randomised row was matched against the map. Similarly, when multiple rows were matched, a map with 10 rows was created from the large dataset. A randomised dataset was then created from the large dataset and matched against the map.

[00480] When evaluating the performance using the original datasets, observations from one dataset were matched against a map created from another dataset. No noise was added to the observations, however the performance when introducing varying amounts of segmentation errors was evaluated. When matching on a single row, 4-time cross validation was employed, using one dataset to create the

map and another dataset to match against it. Similarly when matching multiple rows, the large dataset was used to create the map while the three smaller datasets were matched against it.

[00481] The settings applied both for the randomised and measured datasets are presented in table 6.1.

Dataset	Border StdDev	Feat. StdDev	Seg. Error Prob.
Randomised	0.0-3.0 slices	0.25 m	0.02
Randomised	1.0 slices	0.0-1.0 m	0.02
Randomised	1.0 slices	0.25 m	0.00-0.10
Measured	-	-	0.00-0.10

Table 6.1: The parameters used for determining the performance and robustness of the evaluated localisation methods.

Single Row Localisation

[00482] The performance of the localisation methods with regard to border noise is presented in figure 55, showing that offline localisation performs better than online localisation. Fig. 55 shows the ratio of correctly matched trees using offline, directed 5501 and undirected 5503, and online, directed 5505 and undirected 5507, localization when varying the border noise standard deviation. The ratio of introduced segmentation errors 5509 is also displayed. The figure also shows that the offline localisation is less affected by the usage of robot direction than the online localisation, especially at low noise levels.

[00483] Figure 56 shows the performance of the localisation methods with regard to feature noise, again showing that the offline localisation performs better than the online localisation. Fig. 56 shows the ratio of correctly matched trees using offline, directed 5601 and undirected 5603, and online, directed 5607, localisation when varying the feature noise standard deviation. The ratio of introduced segmentation errors 5609 is also displayed. Furthermore, the figure shows that the undirected localisation methods degrade significantly worse than the directed localisation methods. This is true for the online localisation even from low noise levels while the severe performance degradation in the offline situation only begins at noise levels significantly larger than those occurring in the original datasets.

[00484] The performance with regard to segmentation errors in the randomised datasets is presented in figure 57 shows the ratio of correctly matched trees using offline, directed 5701 and undirected 5703, and online, directed 5705 and undirected 5707, localisation when varying the ratio of segmentation errors introduced to the randomised datasets. The ratio of introduced segmentation errors 5709 is also displayed. This shows that offline localisation degrades gracefully with regard to increased segmentation errors. Similarly directed online localisation degrades well while the undirected online localisation performs worse. These results can be compared to the performance with regard to segmentation errors in the original datasets presented in figure 58. Fig. 58 shows the ratio of correctly matched trees using offline, directed 5801 and undirected 5803, and online, directed 5805 and undirected 5807, localisation when varying the ratio of segmentation errors introduced to the original datasets. The ratio of introduced segmentation errors 5809 is also displayed. The figure shows that the performance in the original datasets are similar to the performance in the randomised datasets, however one notable difference is

that the matching for the undirected offline and the directed online localisation is notably worse when there are no segmentation errors. Nevertheless, it should also be noted that all methods degrade better in the original datasets.

[00485] Fig. 59 shows the ratio of correctly matched trees using offline, directed 5901 and undirected 5903, on online, directed 5905 and undirected 5907, localisation when varying the border noise standard deviation. The ratio 5907 of introduced segmentation errors 5909 is also displayed.

[00486] Fig. 60 shows the ratio of correctly matched trees using offline, directed 6001, and undirected 6002, and online, directed 6005 and undirected 6007, localisation when varying the feature noise standard deviation. The ratio of introduced segmentation errors 6009 is also displayed.

[00487] Fig. 61 shows the ratio of correctly matched trees using offline, directed 6101, and undirected 6103, and online, directed 6105 and undirected 6107, localisation when varying the feature noise standard deviation. The ratio of introduced segmentation errors 6109 is also displayed.

[00488] Fig. 62 shows the ratio of correctly matched trees using offline, directed 6201, and undirected 6203, and online, directed 6205 and undirected 6207, localisation when varying the feature noise standard deviation. The ratio of introduced segmentation errors 6209 is also displayed.

Multiple Row Localisation

[00489] The performance of the localisation methods with regard to border noise is presented in figure 55. The figure shows that the performance is similar to the performance of the single row matching, the notable difference being that all methods degrade slightly worse.

[00490] The performance with regard to feature noise is presented in figure 56, showing that the performance is very similar to the single row localisation. As for the performance with regard to border noise, the matching on multiple rows degrade slightly worse with regard to increased feature noise.

[00491] The performance with regard to segmentation errors in the randomised datasets is presented in figure 57, showing that results are on par with the single row localisation. Furthermore, the results in figure 58 show that the results in the original datasets are similar.

[00492] The evaluation shows that the directed offline localisation yields good results even when the datasets suffer from large amounts of noise and segmentation errors. Using undirected offline localisation yields similar albeit slightly worse results for low noise levels, however as the noise level increases the difference between the two localisation methods becomes more distinct. Furthermore, the directed online localisation performs worse than the two offline methods. Finally, the undirected online localisation performs significantly worse than the other methods, especially at large noise levels.

[00493] A localisation method is described that is robust both to noise and segmentation errors. Furthermore, the effect of integrating the direction of the robot into the localisation has been described, showing that it has a distinct positive impact on the performance. Finally, the performance of both online

and offline localisation has been compared, showing that the offline localisation yields distinctively better results.

[00494] The major use of the online localisation is, naturally, to use it in an online situation. However, this is dependent on the segmentation being performed online as well. Furthermore, if the goal is to estimate the robot's position, there exist multiple other plausible approaches that don't depend on the segmentation. A related approach would be to utilise a Hidden Markov Model where each state represents a slice. Another approach would be to utilise a particle filter and a grid map based on slice observations.

[00495] There exist several alternatives for improving the localisation performance. One approach would be to utilise multiple types of descriptors, such as both height and volume signatures. Furthermore, as described above, an increase in performance can be obtained by storing multiple variants of the same descriptor for each tree. Additionally, it can be assumed that a more complete integration of odometry data can greatly improve the performance with regard to segmentation errors.

Robust Segmentation

[00496] A possible usage of the localisation method described herein is to continuously update a map with new tree observations, giving a map that is always up to date. A problem however is the incorrect segmentations that sometimes occur in the segmentation step. As described herein it is still possible to match most of these trees with the correct corresponding tree in the map. However, even if multiple descriptors of each tree is stored, the segmentation errors risk eventually corrupting the map. Furthermore, if the localisation method is used to monitor the changes of the individual trees, the segmentation error will result in an incorrect estimate of the tree's growth. In order to avoid these problems, a method for detecting segmentation errors can be introduced. As this error detection adds a certain robustness to the segmentation, this error detection is denoted "robust segmentation".

[00497] The segmentation error detection is based on comparing the observed tree with the corresponding tree in the map, thus it assumes that localisation has been performed. Having matched two trees, their height signature descriptors are compared. If the difference between the descriptors is large enough, a segmentation error is assumed to have occurred. The performance of this method is described using both randomised and real datasets and show that it correctly detects most segmentation errors while only labelling a very small amount of correct segmentations as incorrect.

Segmentation Error Detection

[00498] In order to detect segmentation errors, the difference, D , between the observation descriptor and the matched map descriptor is calculated. Introducing a threshold T , a segmentation error is defined to have occurred if

$$D > T. \quad (7.1)$$

Evaluation

[00499] The performance of the error detection method was evaluated both with the randomised datasets as well as the original datasets. The reason for evaluating the performance on the randomised datasets was that it allowed the method's robustness to noise to be evaluated. This is important as an increase in noise levels makes the matched descriptors less similar, possibly causing correct segmentations to be labelled as incorrect. Similarly to the evaluation performed above, the evaluation is performed by varying one noise parameter while keeping the others fixed. The investigated noise levels are presented in table 7.1.

Dataset	Border StdDev	Meas. StdDev	Seg. Error Prob.
Randomised	0.0-3.0 slices	0.25 m	0.02
Randomised	1.0 slices	0.0-1.0 m	0.02
Randomised	1.0 slices	0.25 m	0.00-0.06
Measured	-	-	0.00-0.06

Table 7.1: The parameters used for determining the performance and robustness of the segmentation error detection method.

[00500] In order to quantify the performance, the result of the error detection for each tree was labelled as either:

- True Positive (TP) - A tree is correctly segmented and labelled as correct.
- False Negative (FN) - A tree is correctly segmented and labelled as incorrect
- True Negative (TN) - A tree is incorrectly segmented and labelled as incorrect.
- False Positive (FP) - A tree is incorrectly segmented and labeled as correct.

[00501] This allows the True Positive Ratio (TPR), defined as

$$TPR = \frac{TP}{TP + FN}, \quad (7.2)$$

[00502] and False Positive Ratio (FPR), defined as

$$FPR = \frac{FP}{FP + TN} \quad (7.3)$$

[00503] to be calculated. These ratios are calculated for each noise setting, based on the matching of more than 10000 trees, for values of T ranging from 0 to 100. The result is presented as a Receiver Operating Characteristic (ROC) curve for each evaluated setting.

Results

[00504] The ROC-curves obtained for different rates of segmentation errors inserted into the original datasets are presented in figure 63, showing that approximately 95% of all percent of all segmentation errors can be detected while labelling less than 1% of the correct segmentations as incorrect. Fig. 63 shows the segmentation error detection performance when introducing varying amounts of segmentation errors into the original datasets. Probability of segmentation error: 0.00 (6301), 0.02 (6303), 0.04 (6305), 0.06 (6307). The results when introducing different amounts of segmentation errors into randomised datasets are presented in figure 64, showing similar results. Fig. 64 shows the segmentation error detection performance when introducing varying amounts of segmentation errors into the randomised dataset. Probability of segmentation error: 0.00 (6401), 0.02 (6403), 0.04 (6405), 0.06 (6407). Furthermore, the similarity of the ROC- curves for different error ratios indicate that the method is robust to the ratio of segmentation errors.

[00505] The performance of the segmentation error detection method for different border noise and feature noise levels is shown in figures 65 and 66. Fig. 65 shows the segmentation error detection performance when varying the border noise standard deviations. Border Noise Standard Deviation: 0.0 slices (6501), 1.0 slices (6503), 2.0 slices (6505), 3.0 slices (6507). Fig. 66 shows the segmentation error detection performance when varying the feature noise standard deviations. Feature Noise Standard Deviation: Probability of Segmentation error: 0.0m (6601), 0.25m (6603), 0.50m (6605), 1.0m (6607). Both these figures show that the performance of the error detection decreases with increasing noise. Nevertheless, even for noise levels double the size that was measured in the datasets, the method detects over 90% of all segmentation errors while only labeling very few correct segmentations as incorrect.

[00506] The results show that it is possible to detect more than 95% of all segmentation errors while labeling less than 1% of the correct segmentations as incorrect. Furthermore, the method appears robust to increasing noise levels.

[00507] The results also show that it is possible to detect more than 95% of all segmentation errors while labelling less than 1% of the correct segmentations as incorrect. Furthermore, it has been shown that the method is robust to increases in noise levels.

[00508] It will be understood that it may be necessary to utilise a more complex error detection method. A direct extension could be to utilise multiple types of descriptors, such as height and volume, or a descriptor which describes the characteristics of the tree in greater detail.

[00509] It has been shown that the orchard 3D data could be segmented into individual trees using a 3 state Hidden Semi-Markov Model with both high accuracy and repeatability. It was shown a number of alterations were described such as the introduction of a fourth state, ground removal,

and the use of different observation features. Finally, the different variants were described and it was shown that the best results were obtained by using a combination of height and volume observations without employing any ground removal pre-processing step. When using hand-tuned observation distributions, the best results were obtained when using the mentioned settings in conjunction with a 4 state model. However, the 3 state model yielded very similar results. Furthermore, when learning the observation distributions from a labelled dataset, the 3 state variant yielded better results than the 4 state equivalent.

[00510] Secondly, a set of descriptors were introduced for characterising the trees. The descriptors were used with a simple matching method, showing that the descriptors were informative and consistent enough to allow localisation. Furthermore, it was shown that the height signature descriptor was the most informative of the investigated descriptors. Additionally, different methods of combining multiple measurements when building the map of the orchard were described.

[00511] Thirdly, a robust localisation method were described based on a Hidden Markov Model. The method was used both on single and multiple rows, showing that it yields similar results in both situations. Furthermore, it was shown that integration of the direction of the observed trees had a notable increase on the performance. Additionally, it was shown that offline localisation performs distinctively better than the online equivalent.

[00512] A method for detecting segmentation errors after having performed localisation was described. The method showed that most segmentation errors can be detected while only labelling very few correct segmentations as incorrect.

[00513] It will be understood that GPS independent localisation system may be used by basing the 3D dataset on robot odometry instead of GPS positioning.

[00514] The same HSMM method could be utilized, but base the segmentation on the amount of "tree pixels" observed with a visual sensor. Additionally, the system may utilise image processing approaches such as curve fitting to delimit individual trees. Another alternative could be to utilise a single-point laser rangefinder to measure the distance to the trees at a certain height.

[00515] Regarding characterisation, it will be understood that the trees may be described using image descriptors.

[00516] The described online localisation method may be dependent on the segmentation being performed online. One approach for performing the localisation online could be to utilise a HMM method based on the individual slice measurements, another to use a particle filter with a grid map based on slice observations.

Basic probability Theorems

The derivations presented above make use of a few basic probability theorems, these are presented here.

Marginal Probability The marginal probability $P(x)$ can be calculated from the joint probability $P(x, y)$ by

$$P(x) = \sum_y P(x, y). \quad (\text{I})$$

Conditional Probability The conditional probability $P(x|y)$ can be expressed as

$$P(x|y) = \frac{P(x, y)}{P(y)} \text{ if } P(y) \neq 0. \quad (\text{II})$$

In the case that $P(y) = 0$ then $P(x|y)$ is undefined.

The Chain Rule The chain rule, derived by rewriting (II), is expressed as

$$P(x, y) = P(x|y)P(y). \quad (\text{III})$$

Bayes' Theorem The classic Bayes' theorem, derived from (II) and (III), states that

$$P(y|x) = \frac{P(x|y)P(y)}{P(x)}. \quad (\text{IV})$$

Direction based localisation

Two further methods for performing localisation are described. The first one, directed variant, relies on the observations being observed in the same order as the map, while the second one, in the undirected variant, does not have this restriction. However, as the performance of the directed variant is better, it would be beneficial if the directed variant could be utilised even if the order of the observations were different.

One approach to achieving this is to use the known direction of the robot to determine how it transitions from one tree to the next. Using this information, it would be possible to create a direction vector C defined as

$$C_j = \begin{cases} 1 & , \text{ if the next observed tree is to the front of the current observation} \\ 0 & , \text{ if the direction is unknown or the row changes} \\ -1 & , \text{ if the next observed tree is to the back of the current observation} \end{cases} \quad (C.1)$$

This could be utilised by slightly modifying the employed optimisation algorithm to use three different transition matrices: one forwards transition matrix, A^f , one backwards transition matrix, A^b , and one undirected transition matrix, A^u . The transition matrix used at each step would then be based on the corresponding element of the direction vector. The resulting Forwards algorithm is presented in algorithm 9, however corresponding variants of the Forwards-Backwards and Viterbi algorithm can be created in the same way.

Algorithm 9 The Direction Based HMM Forwards Algorithm

```

for  $j = 1 \rightarrow N$  do                                     ▶ Initialisation
     $\alpha_1(j) = B_j(1)\pi_j$ 
end for

for  $t = 2 \rightarrow T$  do                                     ▶ Recursion

    if  $C(t-1) == 1$  then
         $A = A^f$ 
    else if  $C(t-1) == 0$  then
         $A = A^u$ 
    else if  $C(t-1) == -1$  then
         $A = A^b$ 
    end if

    for  $j = 1 \rightarrow N$  do
         $\alpha_t(j) = B_j(t)(\sum_i A_{ij}\alpha_{t-1}(i))$ 
    end for
end for

for  $t = 1 \rightarrow T$  do                                     ▶ State Maximisation
     $q_t^* = \arg \max_j (\alpha_t(j))$ 
end for
  
```

CLAIMS

1. A method of determining a location relative to the position of one or more natural elements within an environment, the method comprising the steps of:
 - measuring a set of characteristics associated with the natural elements, wherein the set of characteristics comprises at least one characteristic associated with the natural elements;
 - creating a plurality of discrete data sets from the measured set of characteristics;
 - associating the discrete data sets with individual natural elements;
 - sequencing data within the discrete data sets to create a current data sequence; and
 - determining a location relative to the position of one or more natural elements within the environment based on a comparison of the current data sequence with a stored data sequence.
2. The method of claim 1 wherein the stored data sequence is based on a previous characteristic data set created by measuring a second set of characteristics associated with the natural elements, the method further comprising the steps of associating data within the previous characteristic data set with the specific individual natural elements; and
 - sequencing the associated data within the previous characteristic data set.
3. The method of claim 1 wherein:
 - data within the discrete data sets being associated with individual natural elements is data associated with a first characteristic within the first set of characteristics;
 - data being sequenced to create the current data sequence is data associated with a second characteristic within the first set of characteristics, and
 - the first characteristic and second characteristic are different.
4. The method of claim 1 wherein:
 - data within the discrete data sets being associated with individual natural elements is data associated with a first characteristic within the first set of characteristics;
 - the data being sequenced to create the current data sequence is data associated with a second characteristic within the first set of characteristics, and
 - the first characteristic and second characteristic are the same characteristic.
5. The method of claim 1 wherein the step of associating the discrete data sets with individual natural elements comprises the step of:
 - segmenting the characteristic data set into discrete portions, and
 - associating each discrete portion with one of the individual natural elements.
6. The method of claim 5 wherein the step of segmenting the characteristic data set into discrete portions comprises the step of segmenting the characteristic data set using a Hidden Semi-Markov Model.

7. The method of claim 5 wherein the step of segmenting the characteristic data set into discrete portions comprises the step of taking discrete measurements of the first set of characteristics at predefined intervals.
8. The method of claim 7 wherein the predefined intervals are time intervals.
9. The method of claim 7 wherein the predefined intervals are distance intervals.
10. The method of claim 1 wherein the step of sequencing data within the discrete data sets to create a current data sequence comprises the step of grouping the discrete data sets into a defined sequence length.
11. The method of claim 1 wherein the step of sequencing data within the discrete data sets comprises:
obtaining a plurality of descriptor values, wherein each descriptor value is based on a plurality of single data points, where each single data point is selected from one of a plurality of discrete data sets that are associated with one individual natural element, and
positioning the obtained descriptor values in a defined sequence.
12. The method of claim 11 wherein the descriptor value is calculated from the single data points by selecting one of the single data points or calculating a new value based on the single data points.
13. The method of claim 1 wherein the step of sequencing data within the discrete data sets comprises:
obtaining a plurality of descriptor values, wherein each descriptor value is based on a plurality of data points, where each data point in the plurality of data points is selected from one of a plurality of discrete data sets that are associated with one individual natural element, and
positioning the obtained descriptor values in a defined sequence.
14. The method of claim 1 wherein the step of determining a location comprises the step of using particle filters on the current data sequence.
15. The method of claim 1 wherein the step of determining a location comprises the step of using a HMM algorithm.
16. The method of claim 15 wherein the HMM algorithm comprises one of a Viterbi algorithm, Forwards algorithm and Forwards-Backwards algorithm.
17. The method of claim 1 wherein the set of characteristics comprises one or more of height, volume, density, colour, and temperature characteristics of natural elements in the environment.
18. The method of claim 1 further comprising the steps of:

performing further measurements before, during or after determining a location and associating the further measurements with a natural element based on the determined location.

19. The method of claim 1 further comprising the steps of determining a potential error based on the comparison of the current data sequence and stored data sequence, wherein the potential error is created from discrete data sets not being associated with the correct individual natural element.
20. A system for determining a location relative to the position of one or more natural elements within an environment, the system comprising a data capture module, a segmentation module, a sequencing module, a characterisation module and a localisation module adapted to perform the method according to any one of claims 1 to 19.
21. A method of determining a set of characteristics associated with natural elements in an environment for determining a location relative to the position of one or more natural elements within the environment, the method comprising the steps of:
 - measuring a set of characteristics associated with the natural elements, wherein the set of characteristics comprises at least one characteristic associated with the natural elements;
 - creating a plurality of discrete data sets from the measured set of characteristics;
 - associating the discrete data sets with individual natural elements;
 - sequencing data within the discrete data sets to create a data sequence; and
 - storing the data sequence suitable for determining a location within the environment.
22. The method of claim 21, wherein data within the discrete data sets being associated with individual natural elements is data associated with a first characteristic within the first set of characteristics;
 - the data being sequenced to create the current data sequence is data associated with a second characteristic within the first set of characteristics, and
 - the first characteristic and second characteristic are different.
23. The method of claim 21, wherein data within the discrete data sets being associated with individual natural elements is data associated with a first characteristic within the first set of characteristics;
 - the data being sequenced to create the current data sequence is data associated with a second characteristic within the first set of characteristics, and
 - the first characteristic and second characteristic are the same characteristic.
24. The method of claim 21, wherein the step of associating the discrete data sets with individual natural elements comprises the step of:
 - segmenting the characteristic data set into discrete portions, and
 - associating each discrete portion with one of the individual natural elements.

25. The method of claim 24, wherein the step of segmenting the characteristic data set into discrete portions comprises the step of segmenting the characteristic data set using a Hidden Semi-Markov Model.
26. The method of claim 24, wherein the step of segmenting the characteristic data set into discrete portions comprises the step of taking discrete measurements of the first set of characteristics at predefined intervals.
27. The method of claim 26, wherein the predefined intervals are time intervals.
28. The method of claim 26, wherein the predefined intervals are distance intervals.
29. The method of claim 21, wherein the step of sequencing data within the discrete data sets to create a current data sequence comprises the steps of:
grouping the discrete data sets into a defined sequence length.
30. The method of claim 21, wherein the step of sequencing data within the discrete data sets comprises:
obtaining a plurality of descriptor values, wherein each descriptor value is based on a plurality of single data points, where each single data point is selected from one of a plurality of discrete data sets that are associated with one individual natural element, and
positioning the obtained descriptor values in a defined sequence.
31. The method of claim 30, wherein the descriptor value is calculated from the single data points by selecting one of the single data points or calculating a new value based on the single data points.
32. The method of claim 21, wherein the step of sequencing data within the discrete data sets comprises:
obtaining a plurality of descriptor values, wherein each descriptor value is based on a plurality of data points, where each data point in the plurality of data points is selected from one of a plurality of discrete data sets that are associated with one individual natural element, and
positioning the obtained descriptor values in a defined sequence.
33. The method of claim 21 further comprising the step of determining a location value based on a GPS signal, and storing the location value with the current data sequence.

34. A system for determining a set of characteristics associated with natural elements in an environment for determining a location relative to the position of one or more natural elements within the environment, the system comprising a data capture module, a segmentation module, a sequencing module, a characterisation module and a localisation module adapted to perform the method according to any one of claims 21 to 33.

**The University of Sydney
Patent Attorneys for the Applicant/Nominated Person
SPRUSON & FERGUSON**

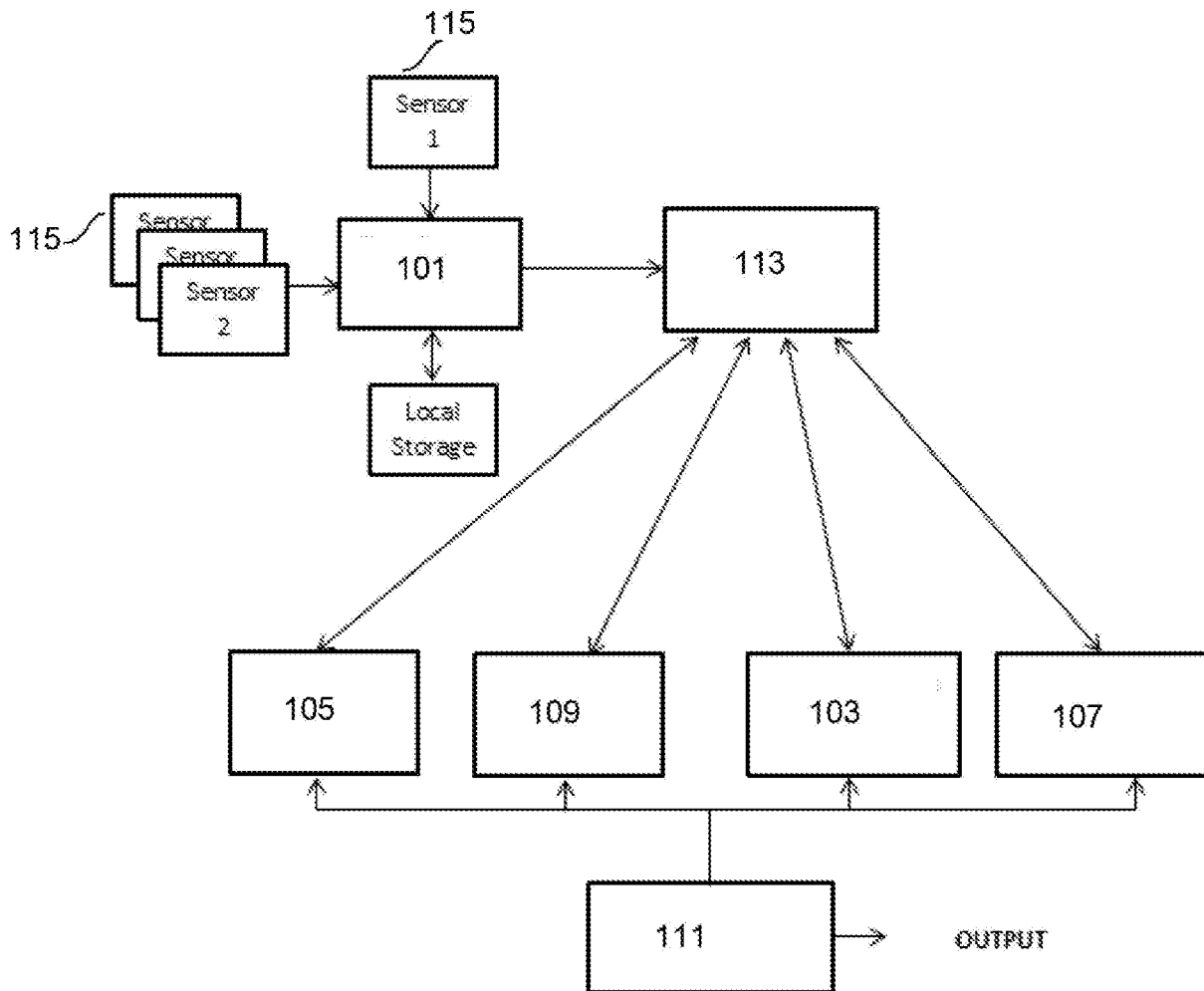


FIGURE 1A

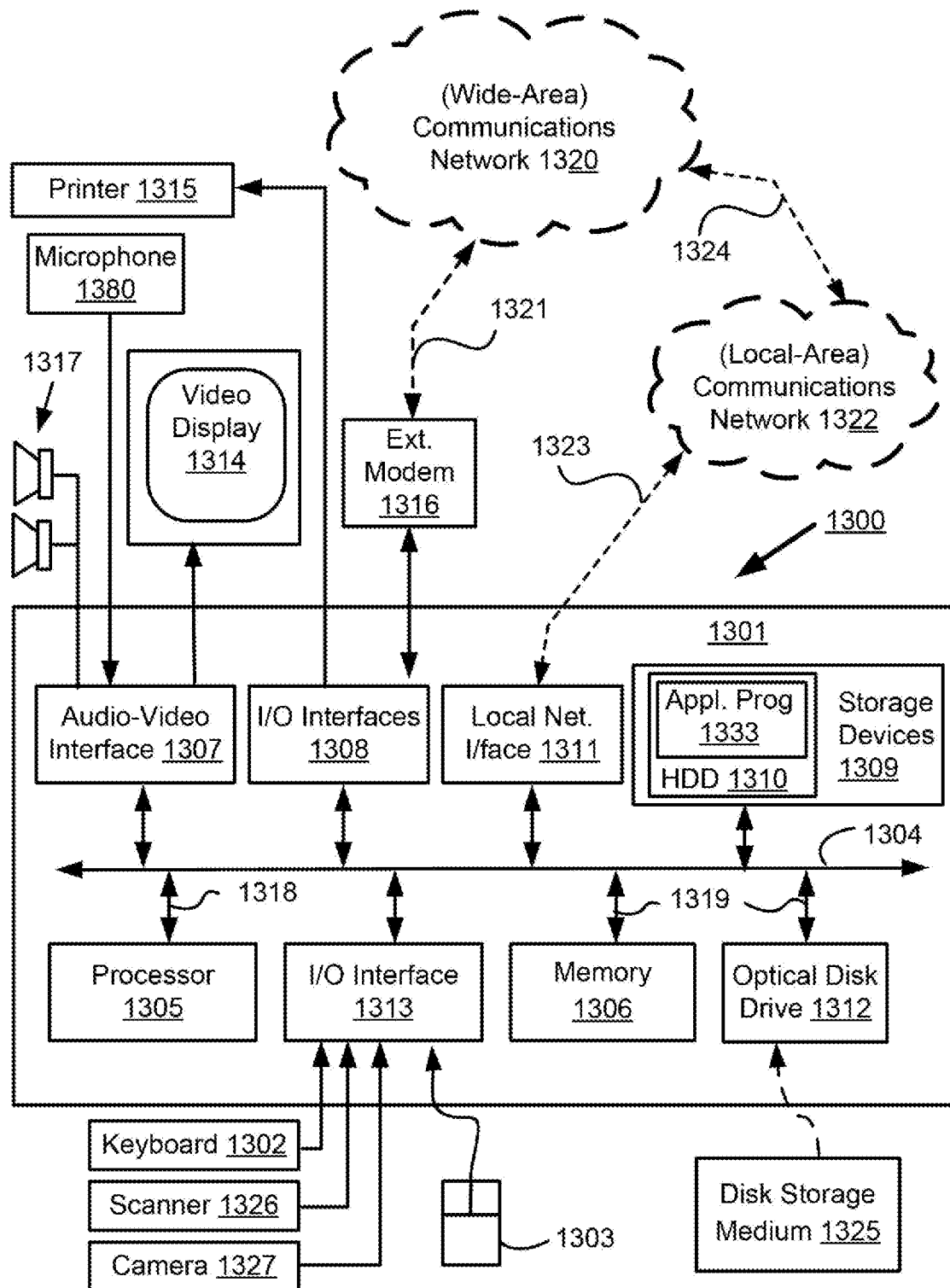


Fig. 1B

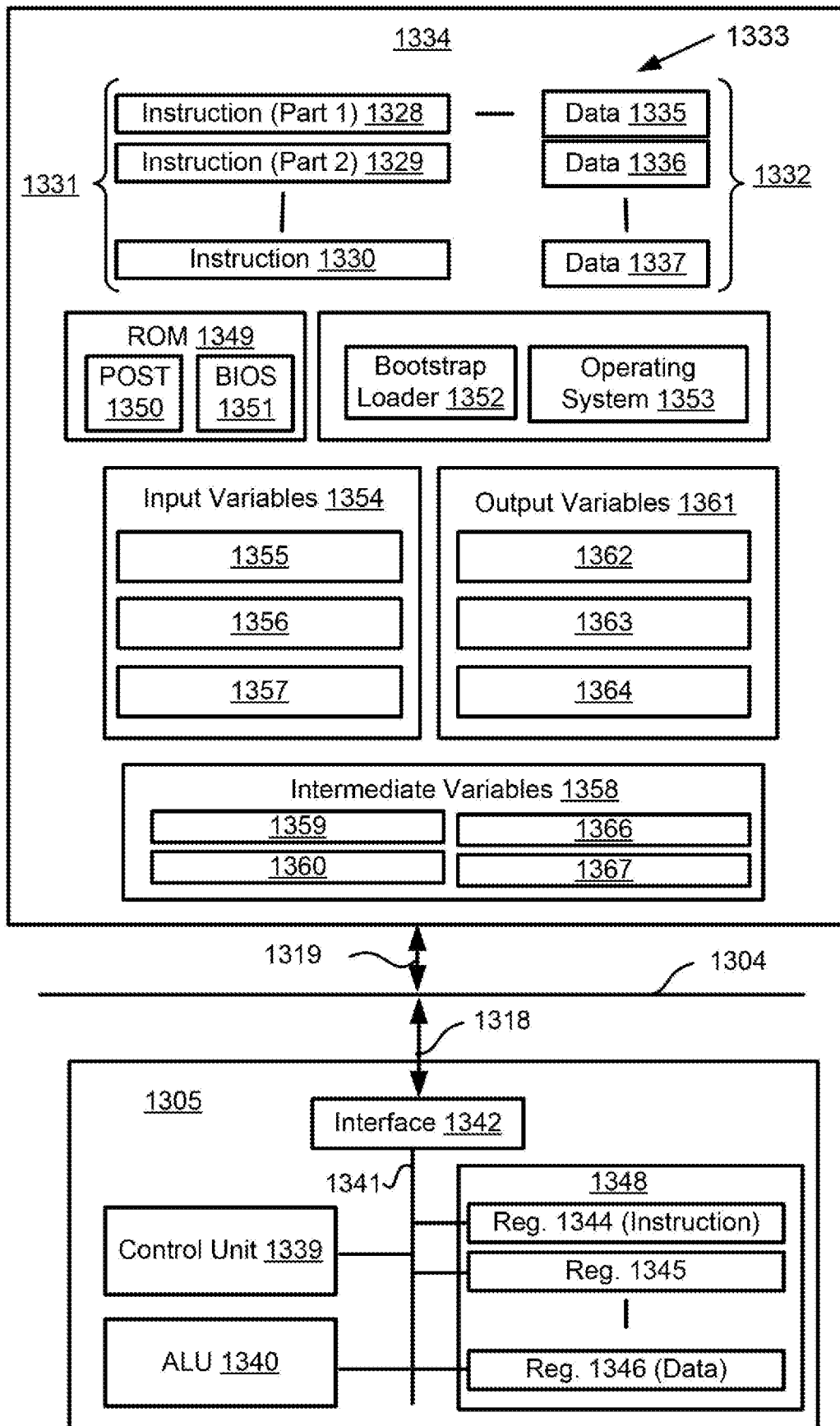


Fig. 1C

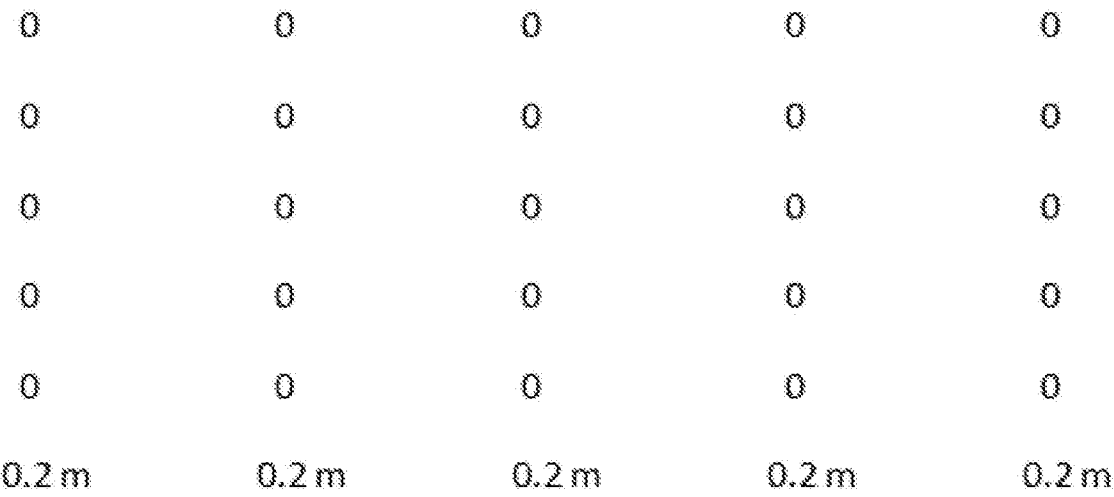


FIGURE 2

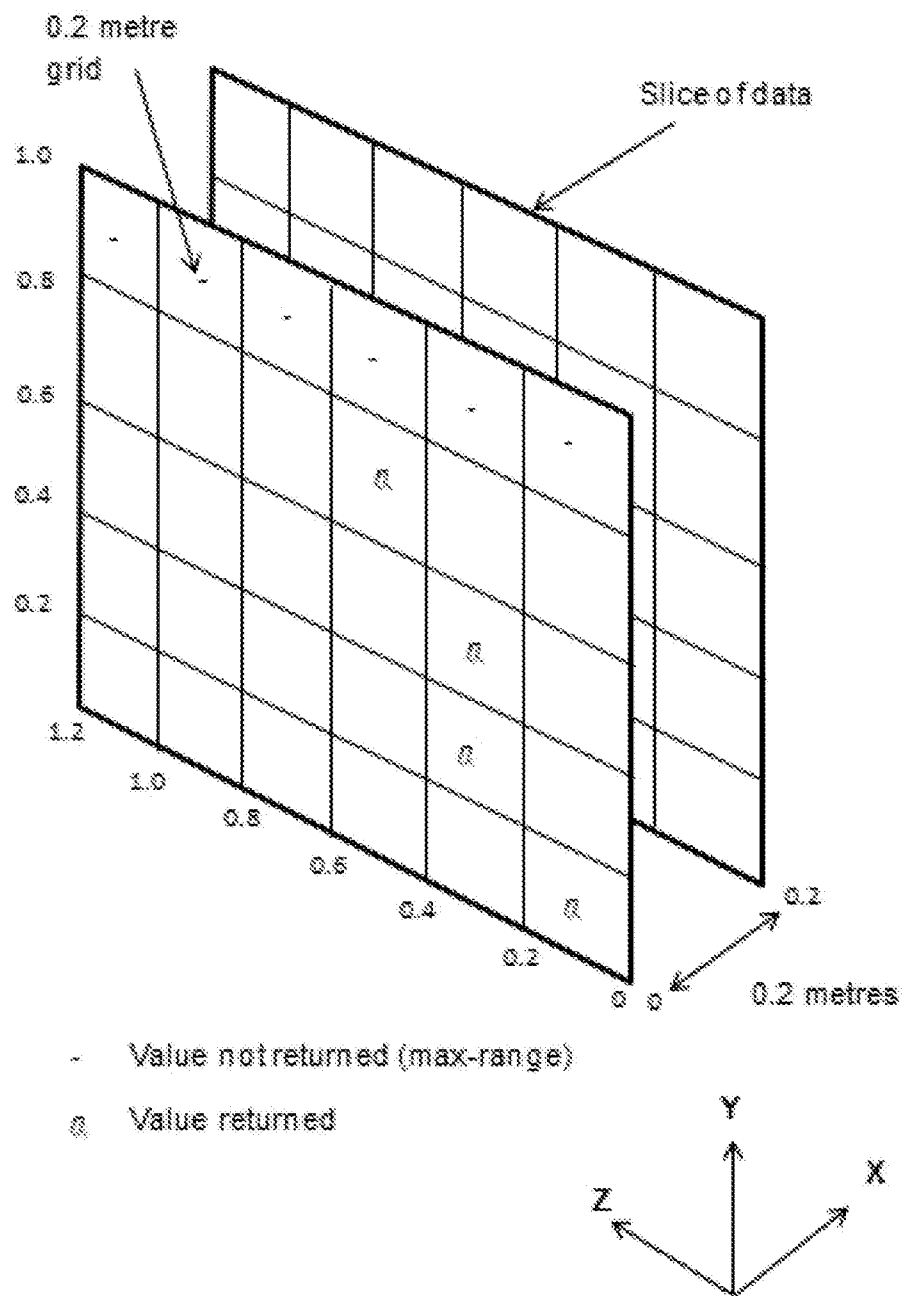


FIGURE 3

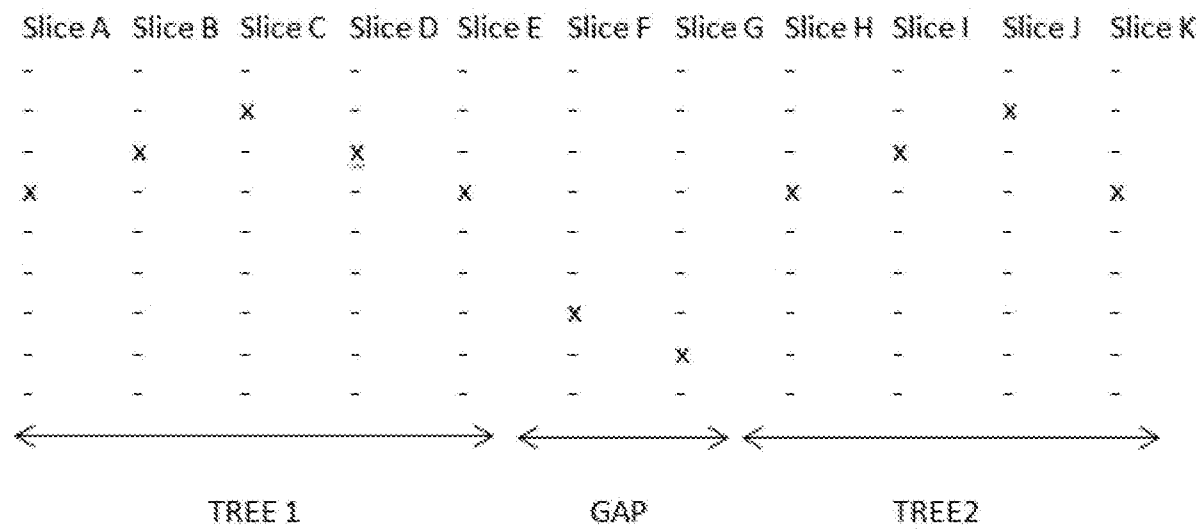


FIGURE 4

Slice A	Slice B	Slice C	Slice D	Slice E
-	-	-	-	-
-	-	X	-	-
-	X	-	X	-
X	-	-	-	X
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	-	-

FIGURE 5

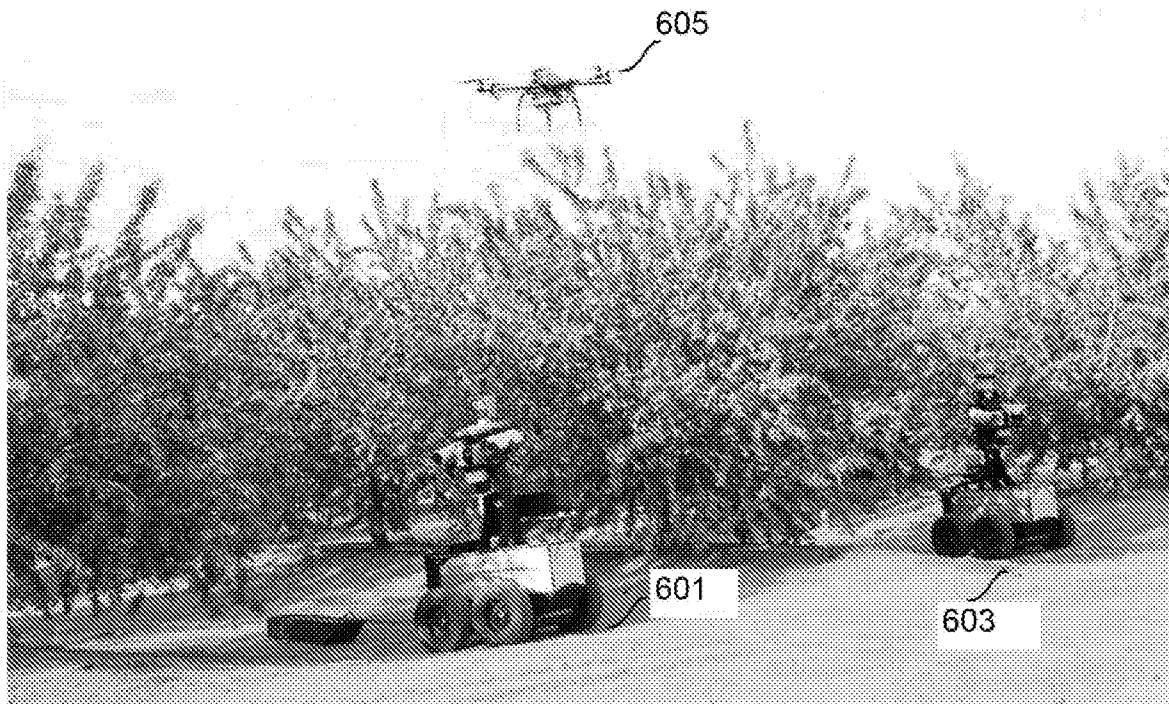


FIGURE 6

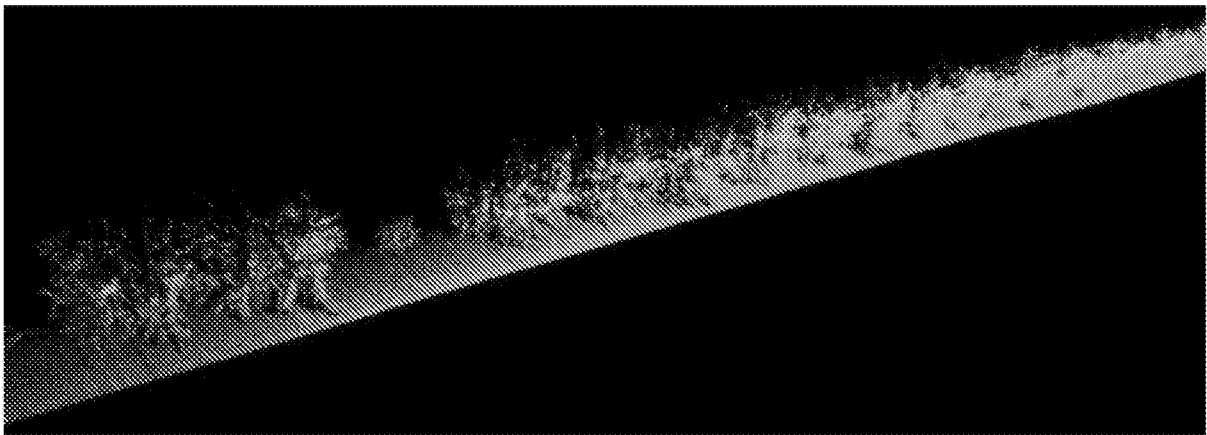


FIGURE 7

9/46

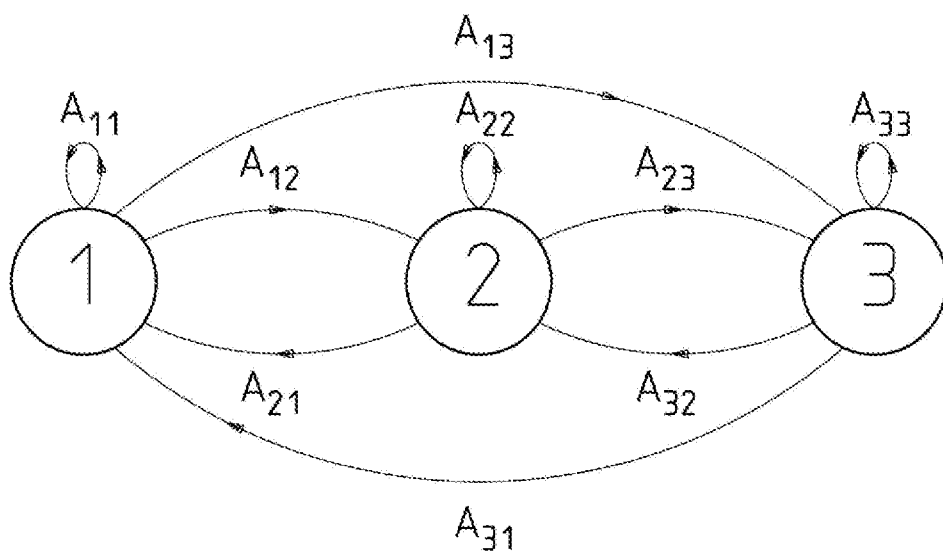


FIGURE 8

10/46

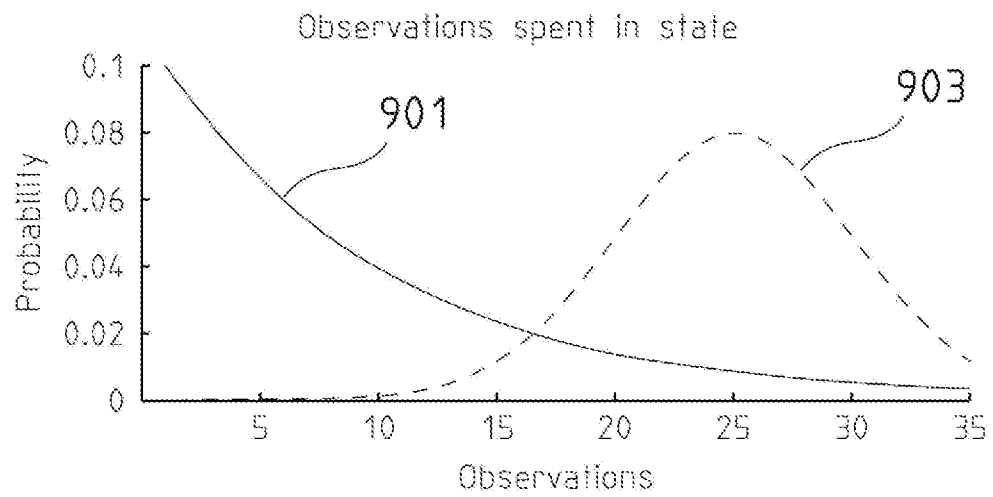


FIGURE 9

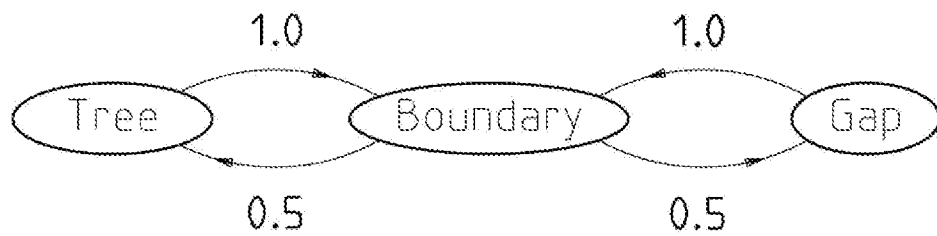


FIGURE 10

11/46

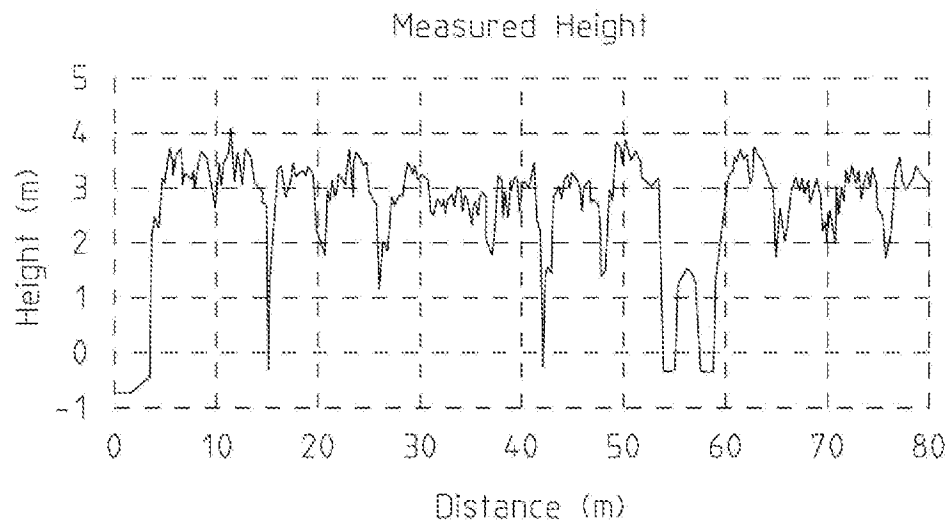


FIGURE 11A

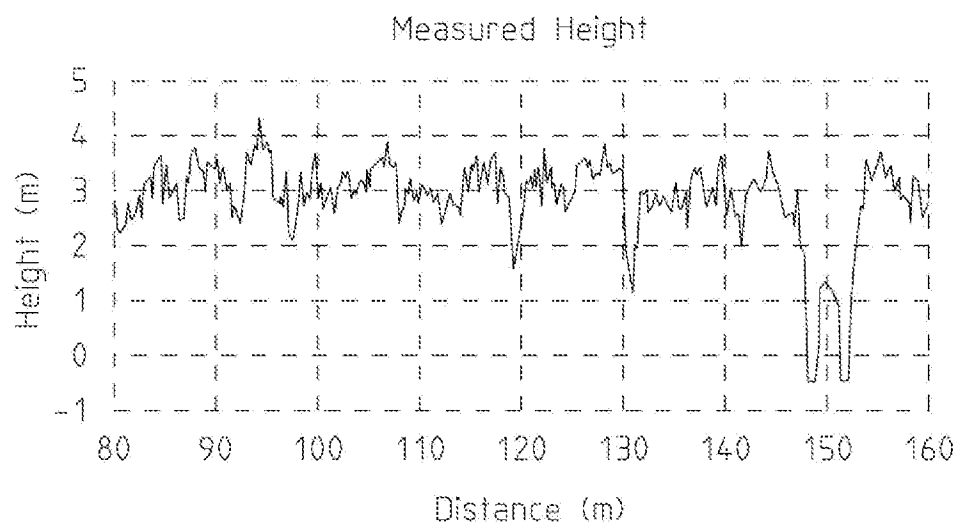


FIGURE 11B

12/46

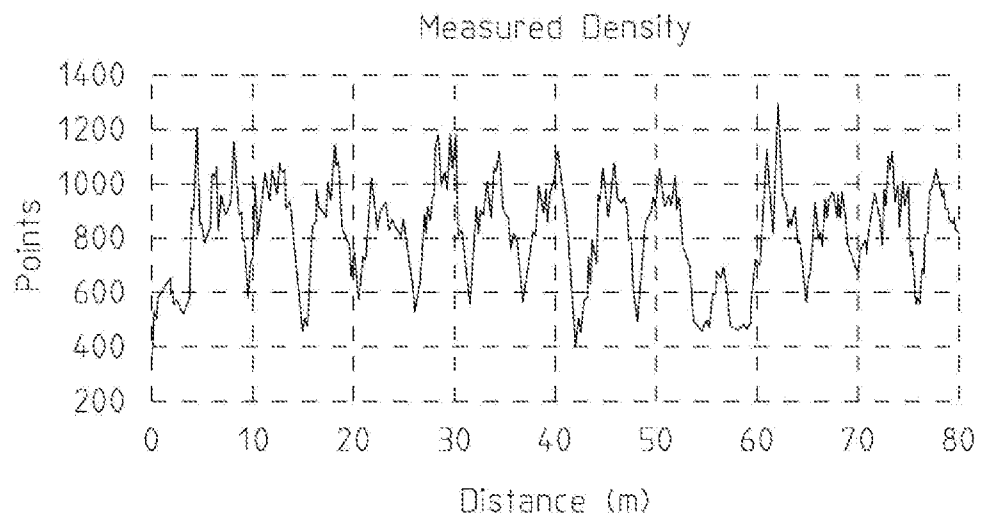


FIGURE 12A

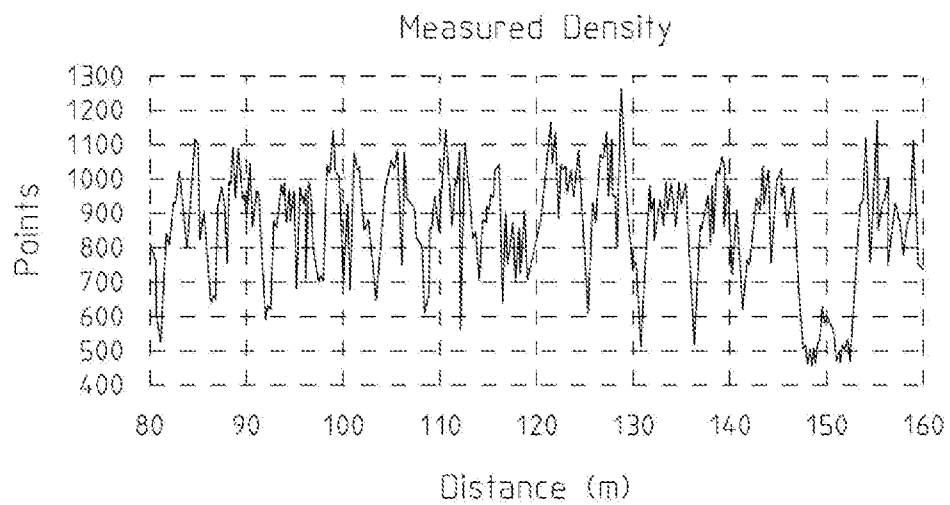


FIGURE 12B

13/46

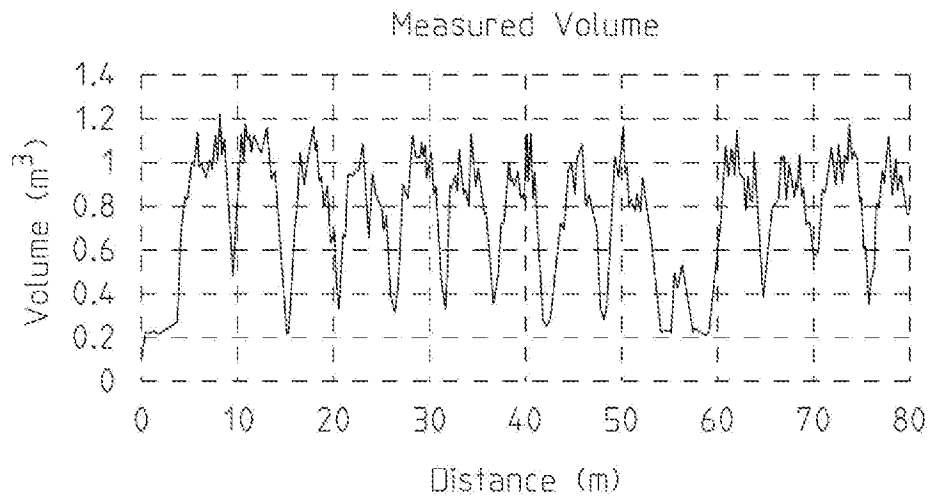


FIGURE 13A

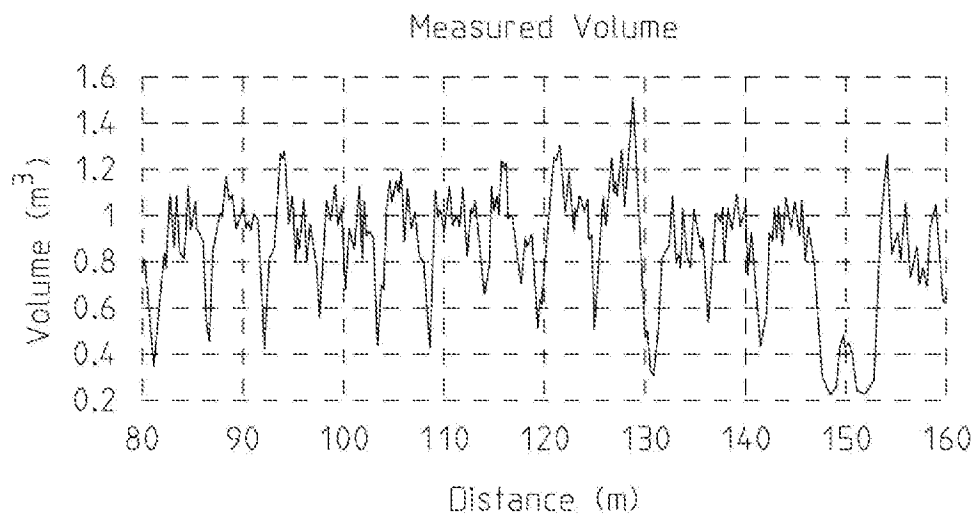


FIGURE 13B

14/46

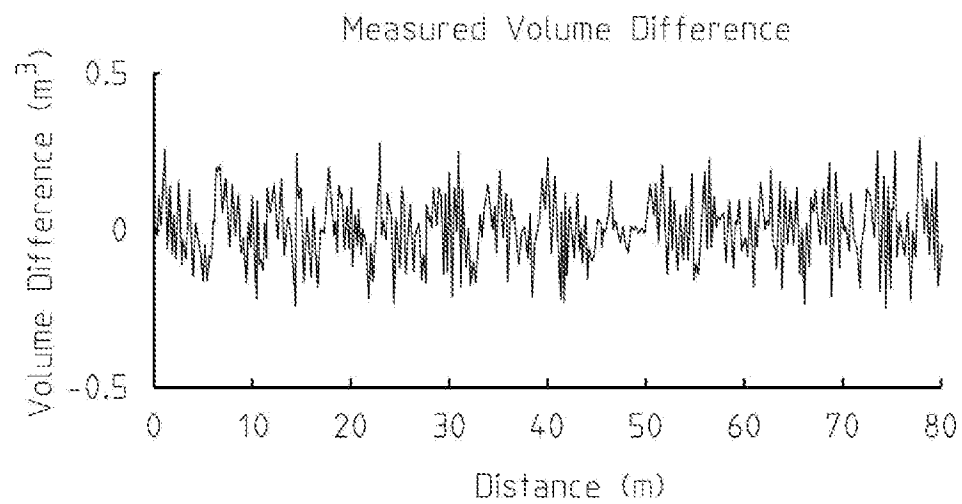


FIGURE 14

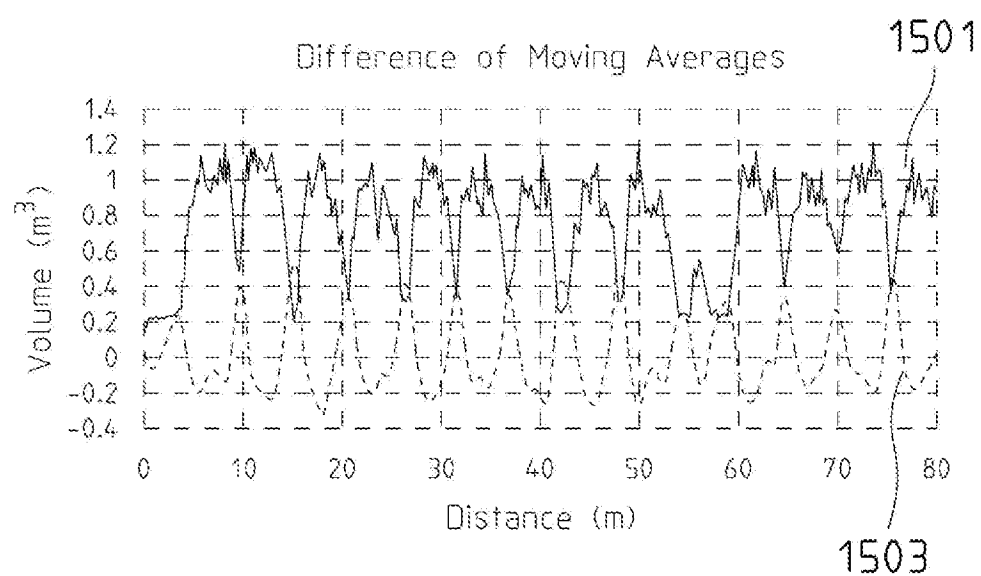


FIGURE 15

15/46

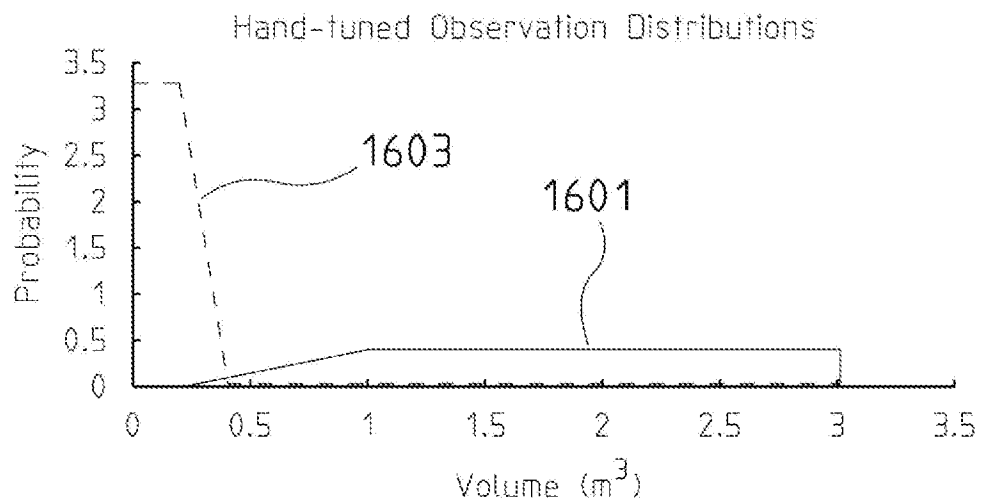


FIGURE 16A

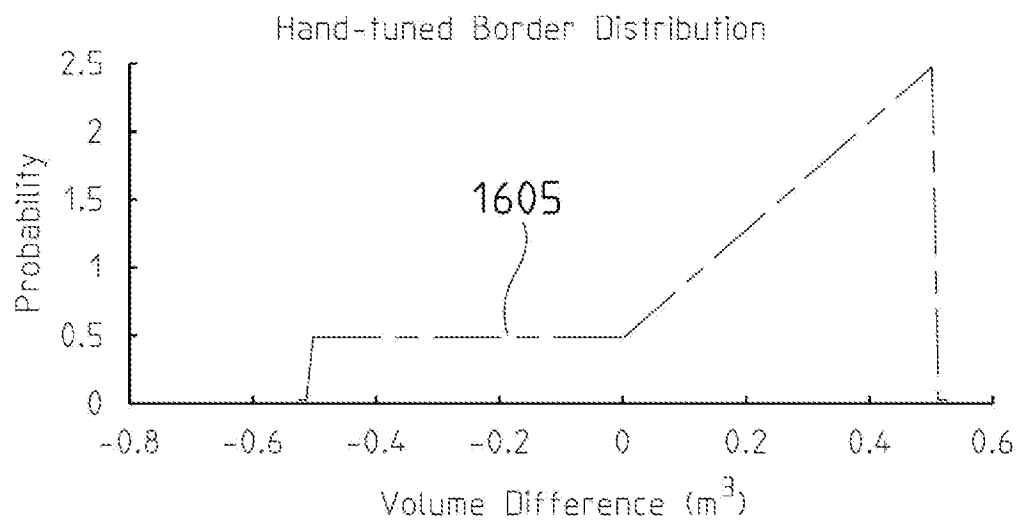


FIGURE 16B

16/46

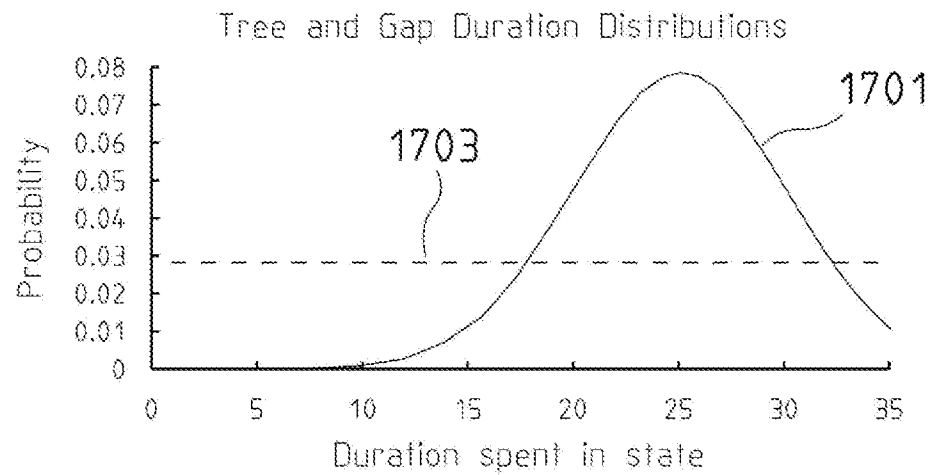


FIGURE 17

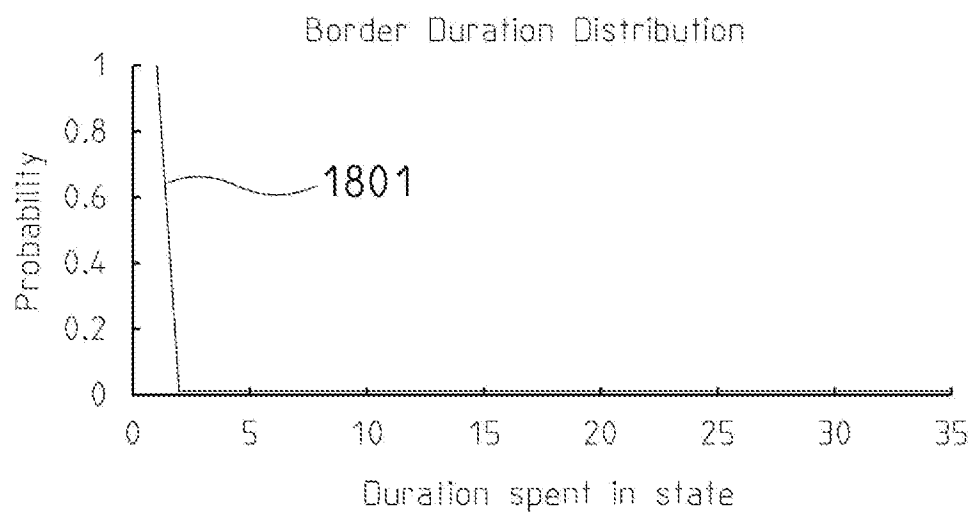


FIGURE 18

17/46

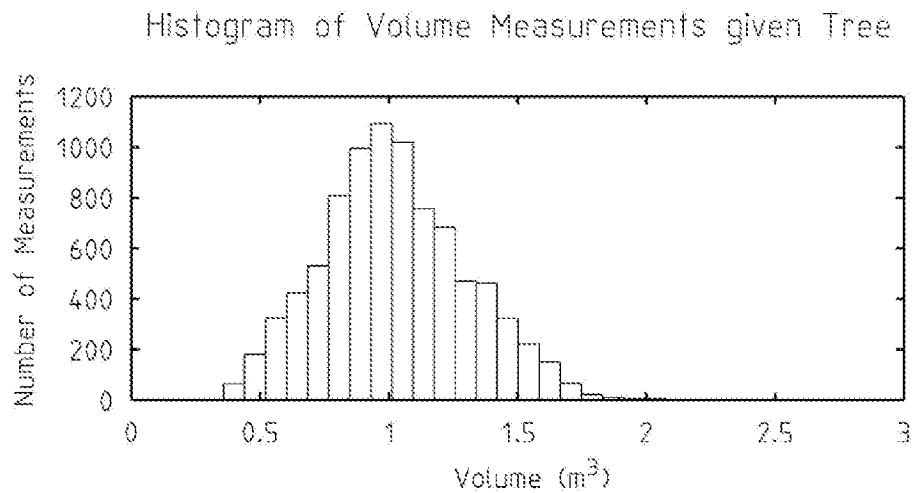


FIGURE 19A

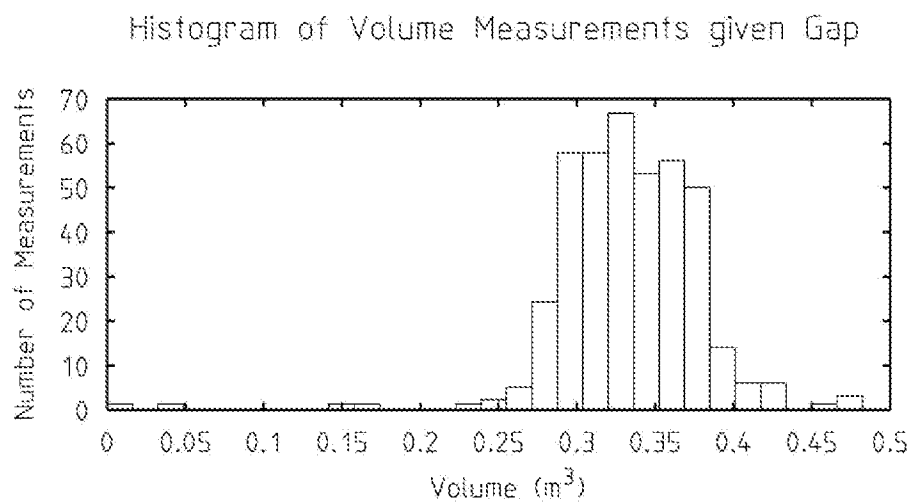


FIGURE 19B

18/46

Histogram of Volume Measurements given Border

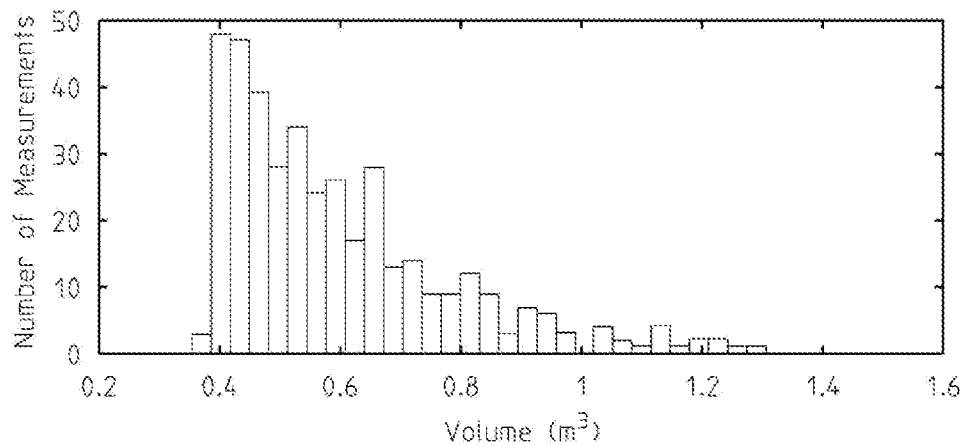


FIGURE 19C

Learnt Observation Distributions

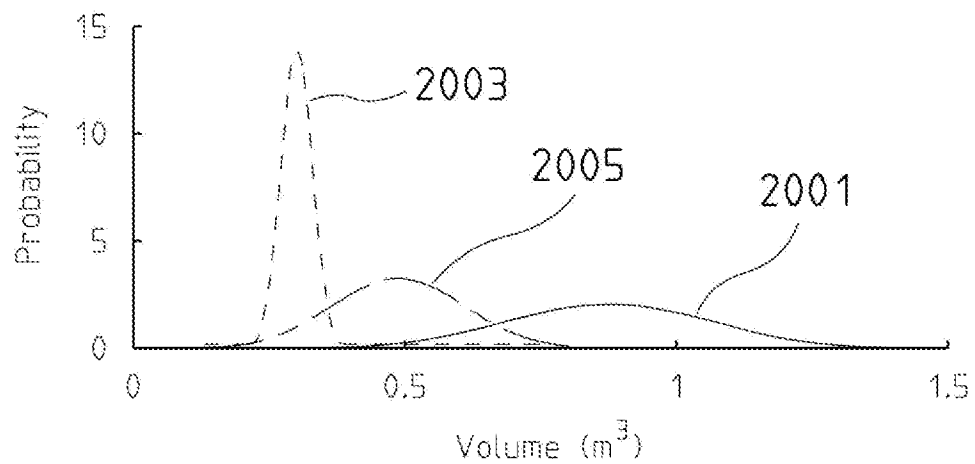


FIGURE 20

19/46

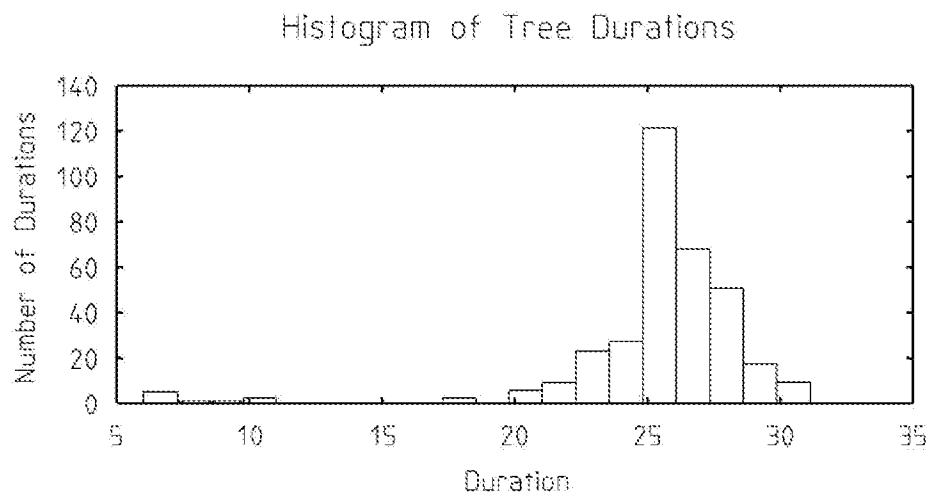


FIGURE 21

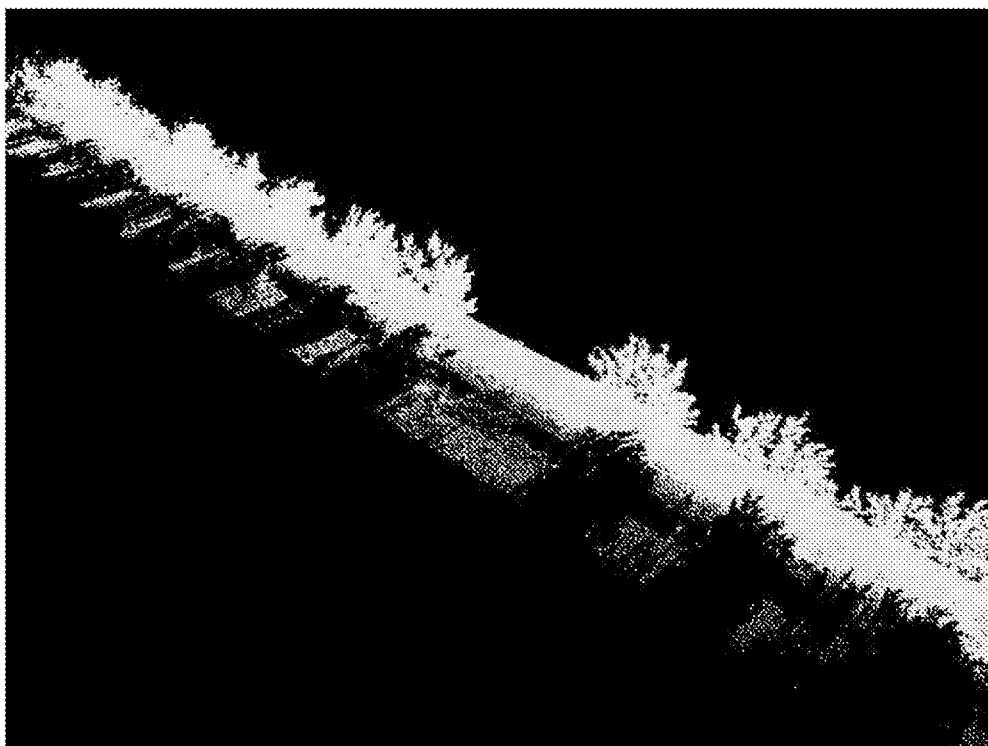


FIGURE 22A



FIGURE 22B

21/46

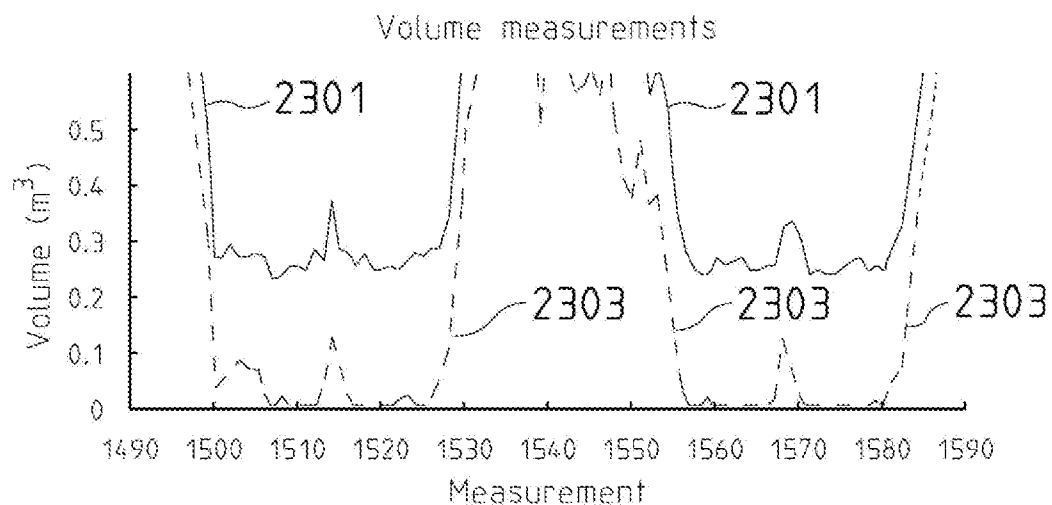


FIGURE 23

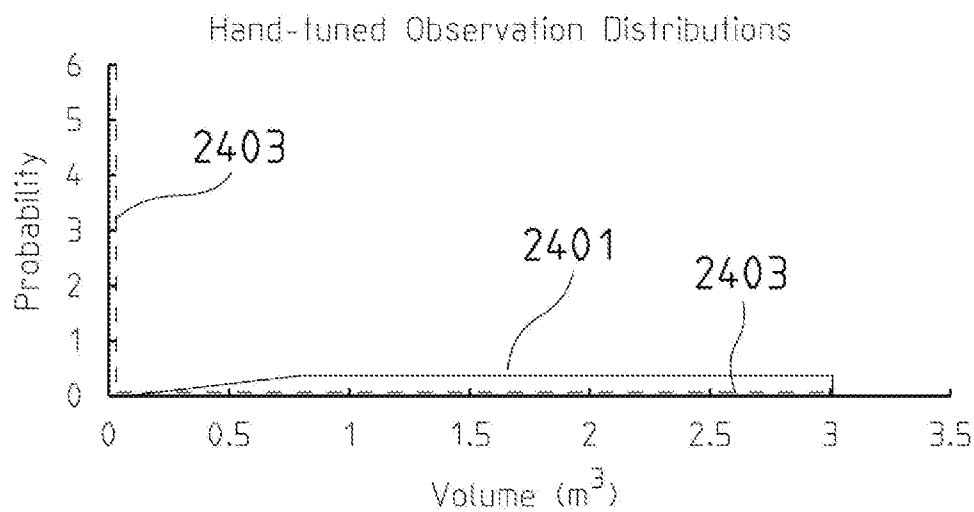


FIGURE 24

22/46

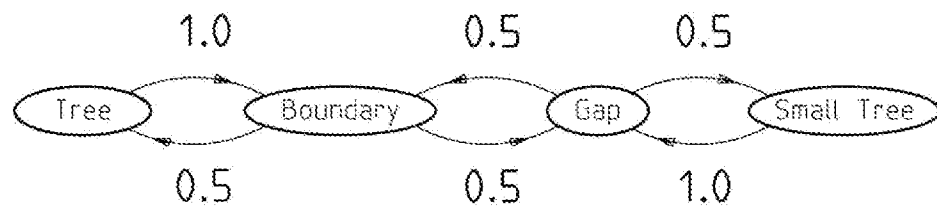


FIGURE 25

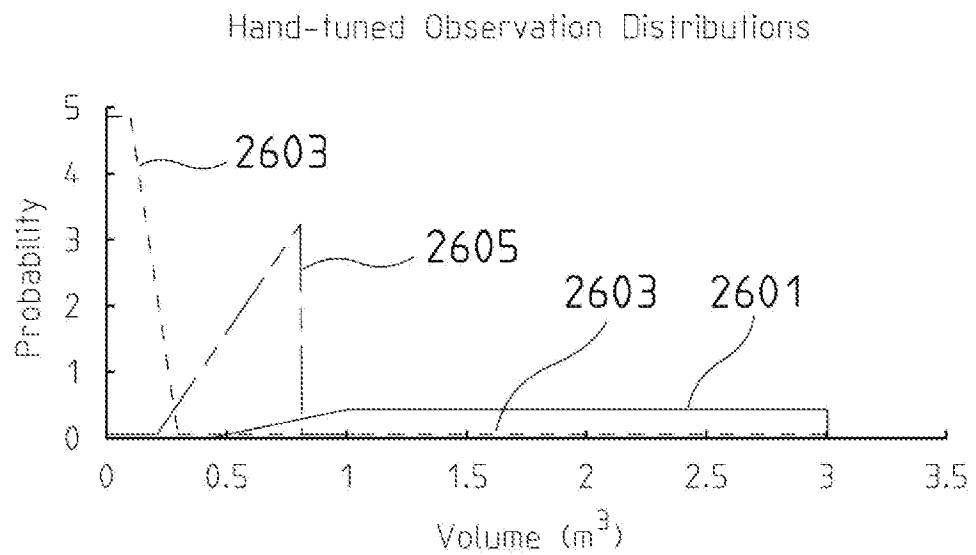


FIGURE 26A

23/46

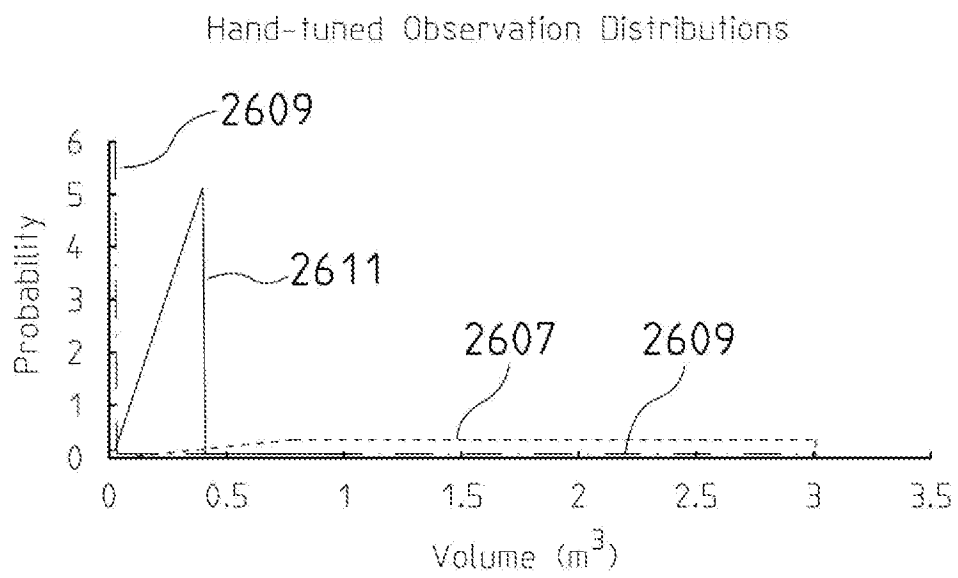


FIGURE 26B

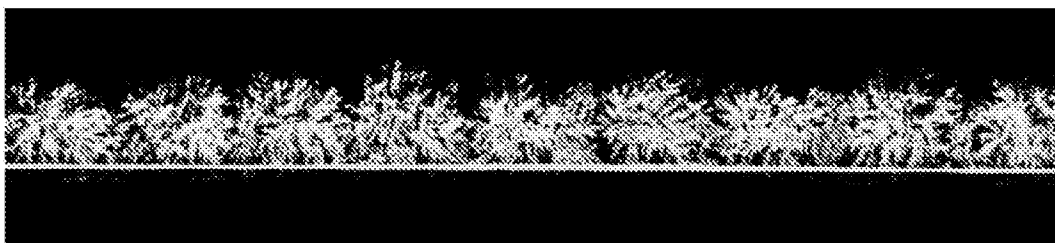


FIGURE 27

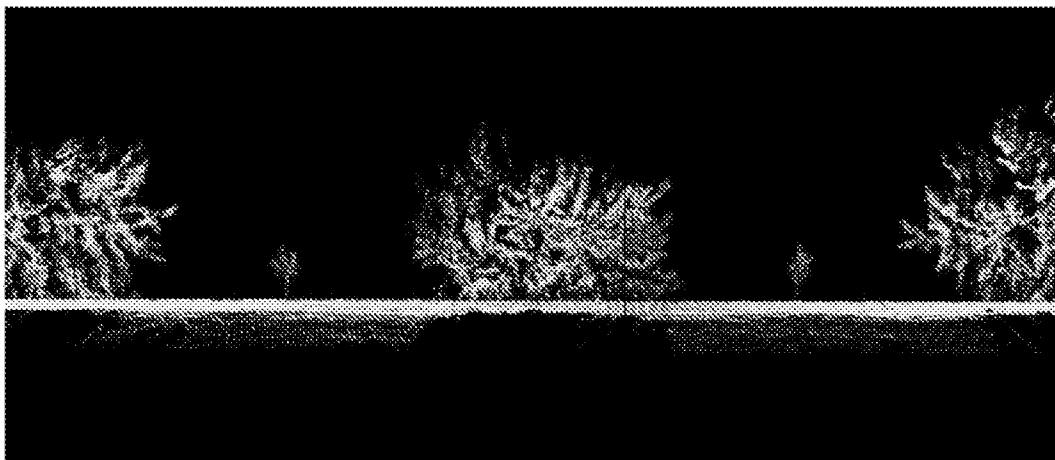


FIGURE 28

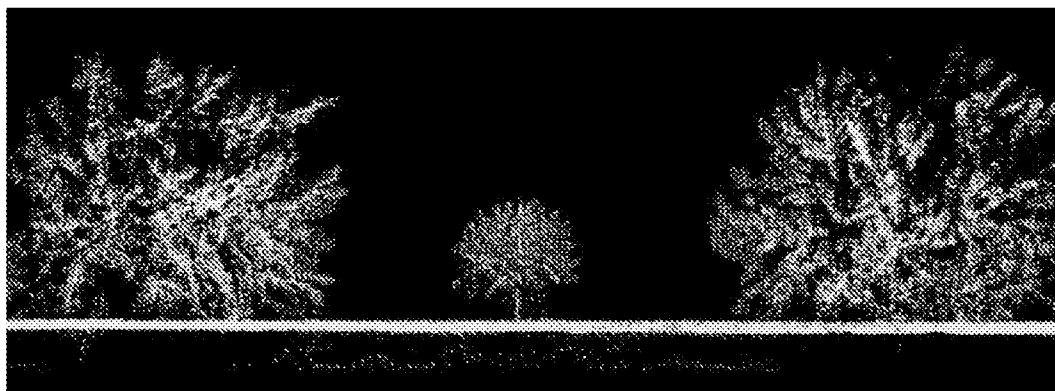


FIGURE 29

25/46



FIGURE 30

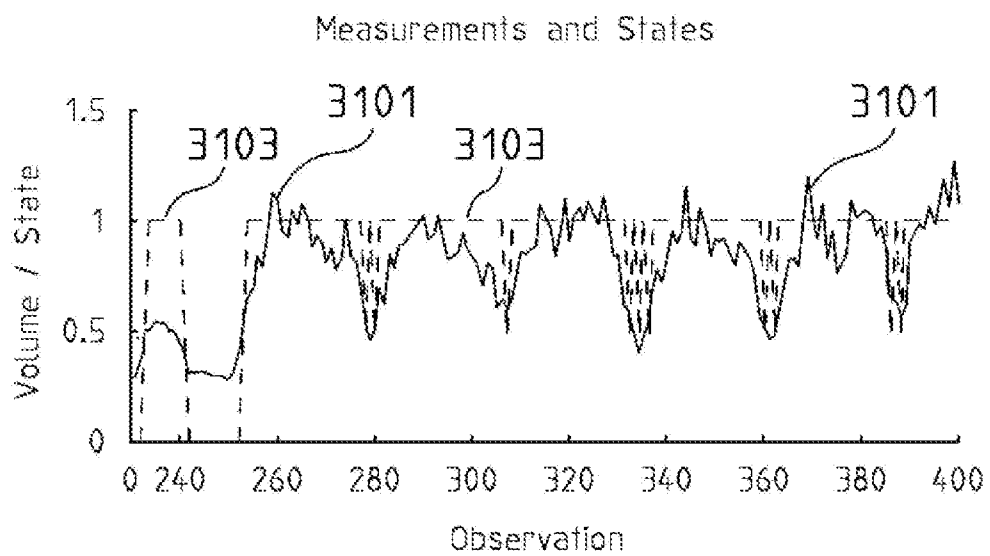


FIGURE 31

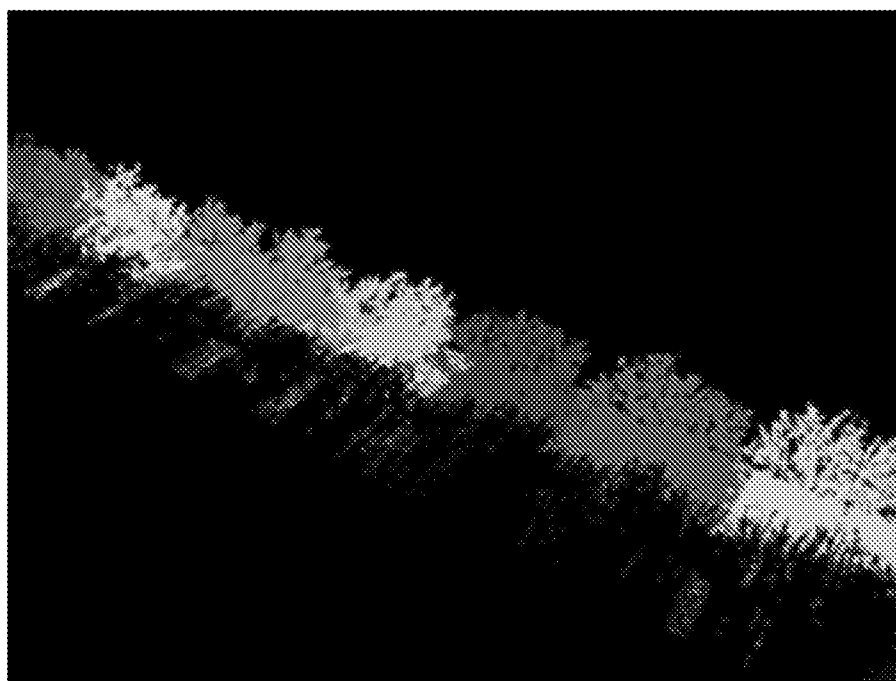


FIGURE 32

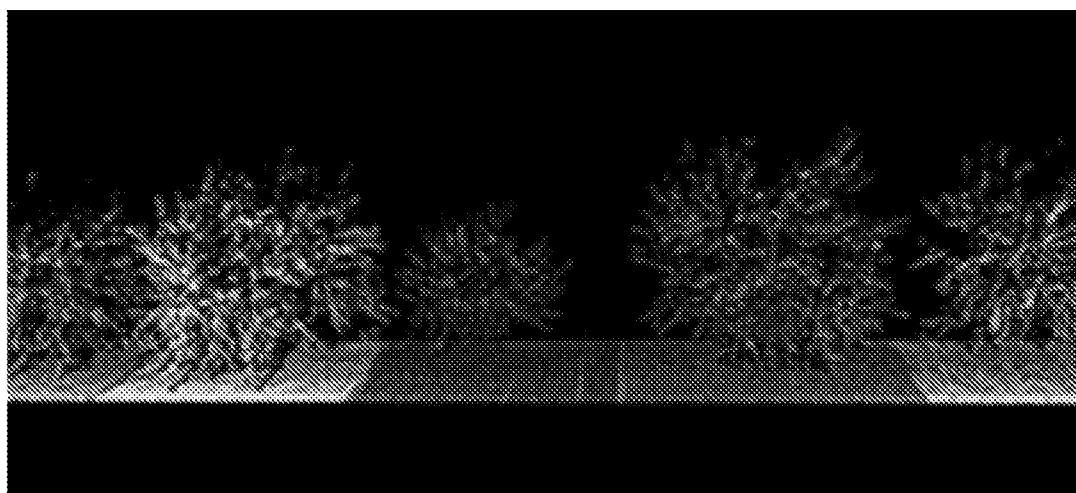


FIGURE 33

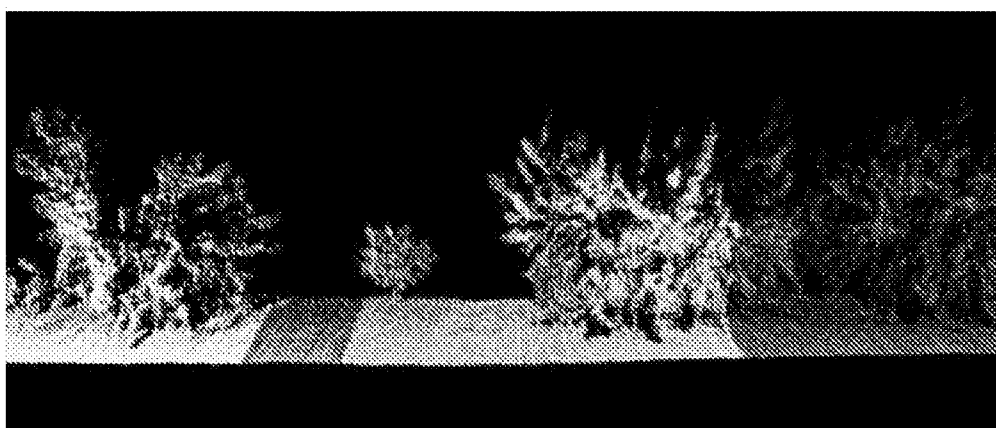


FIGURE 34

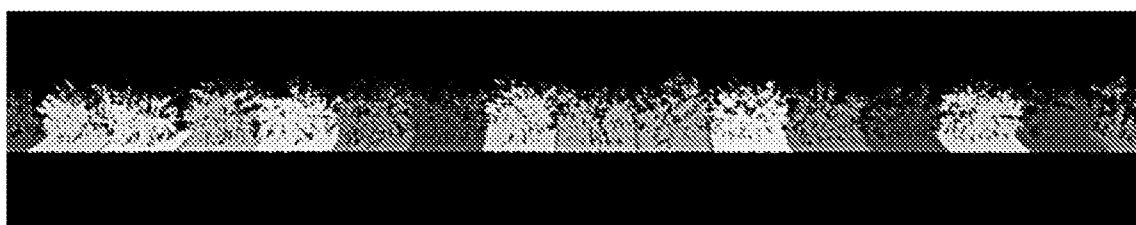


FIGURE 35

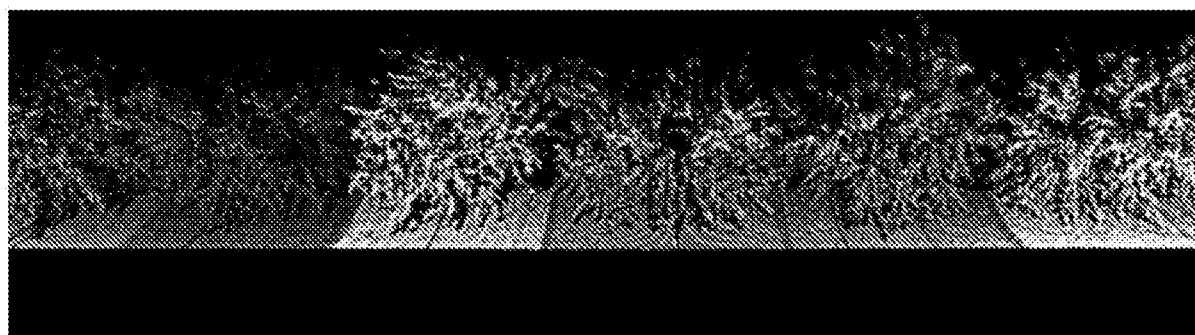


FIGURE 36

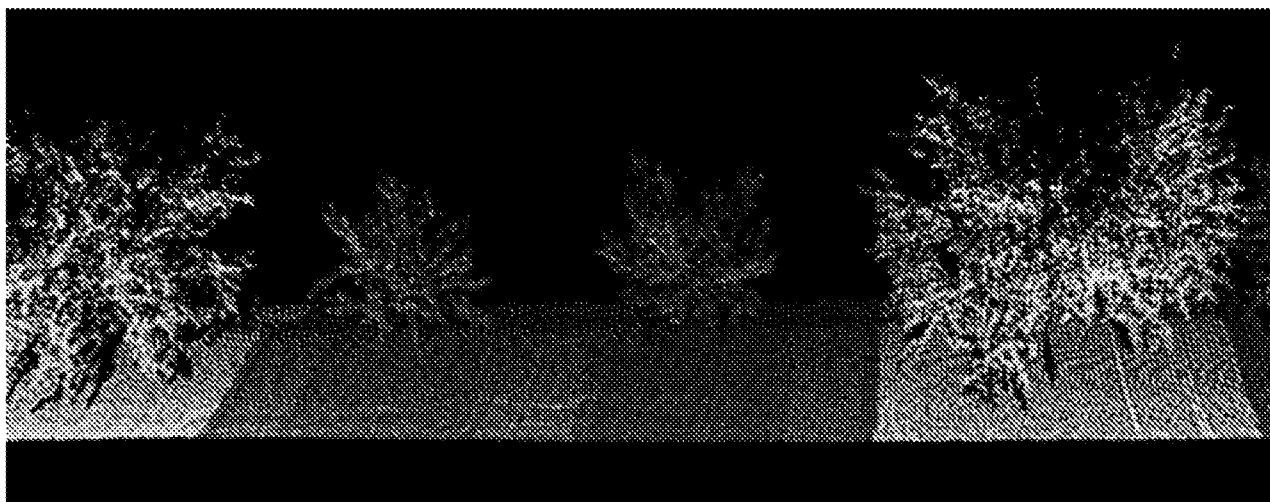


FIGURE 37

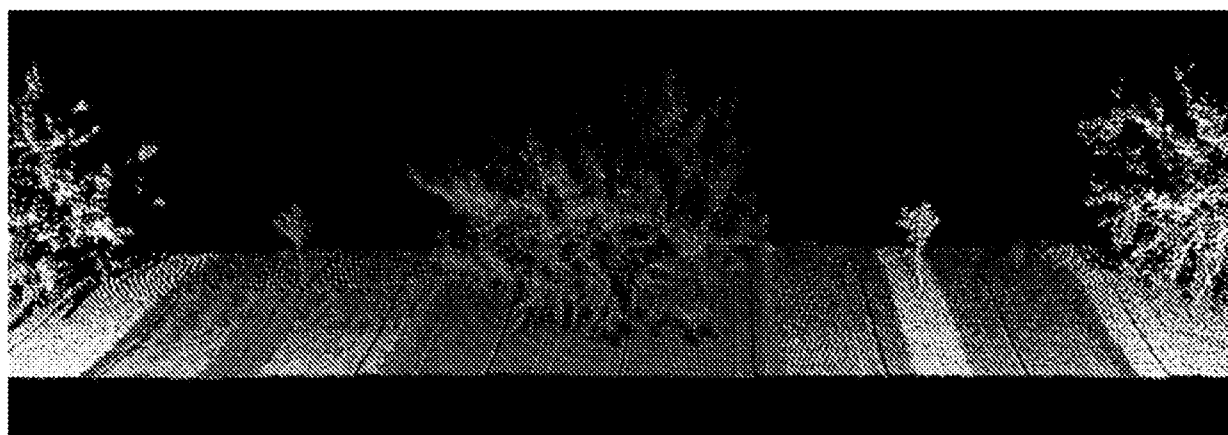


FIGURE 38

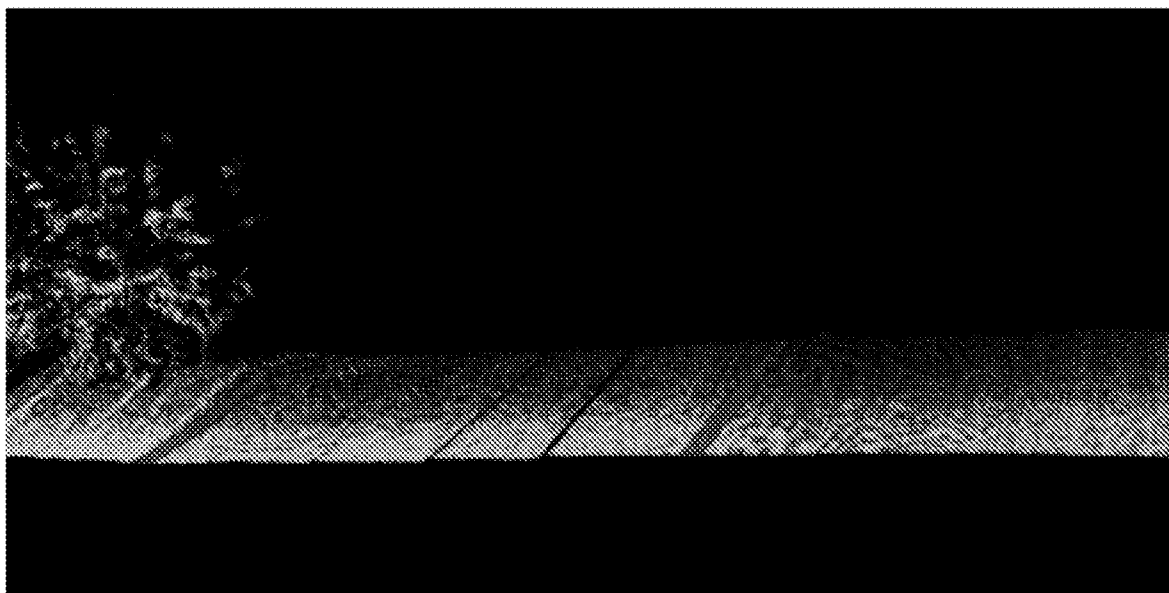


FIGURE 39

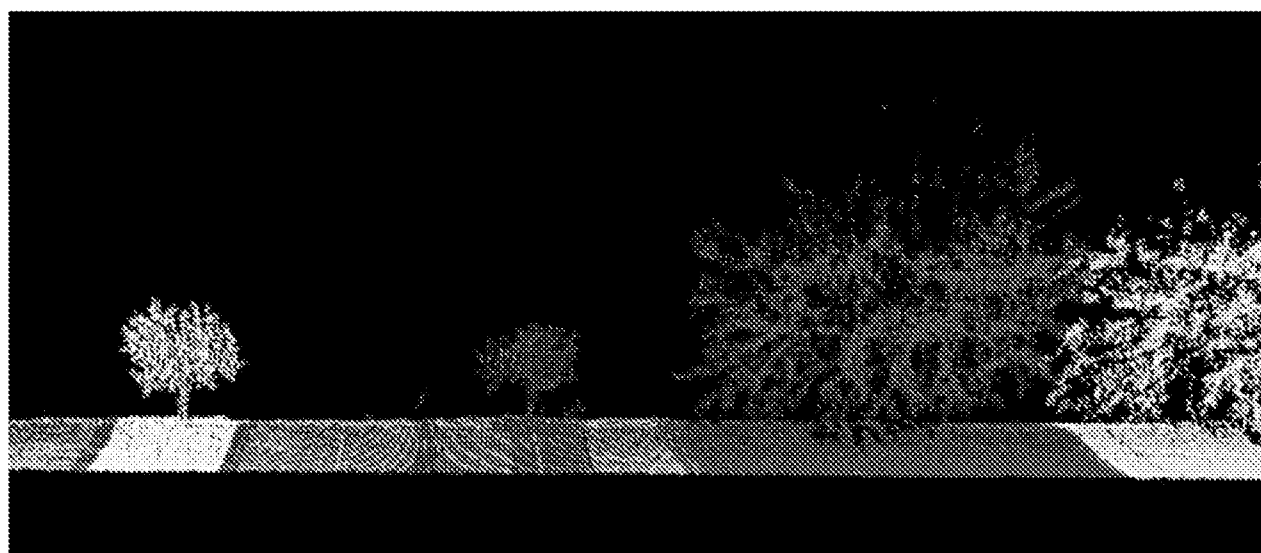


FIGURE 40

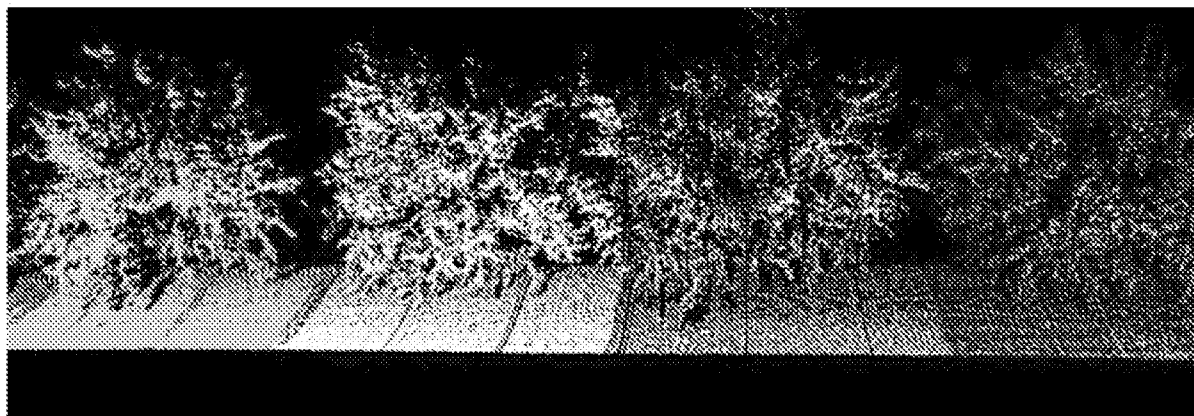


FIGURE 41

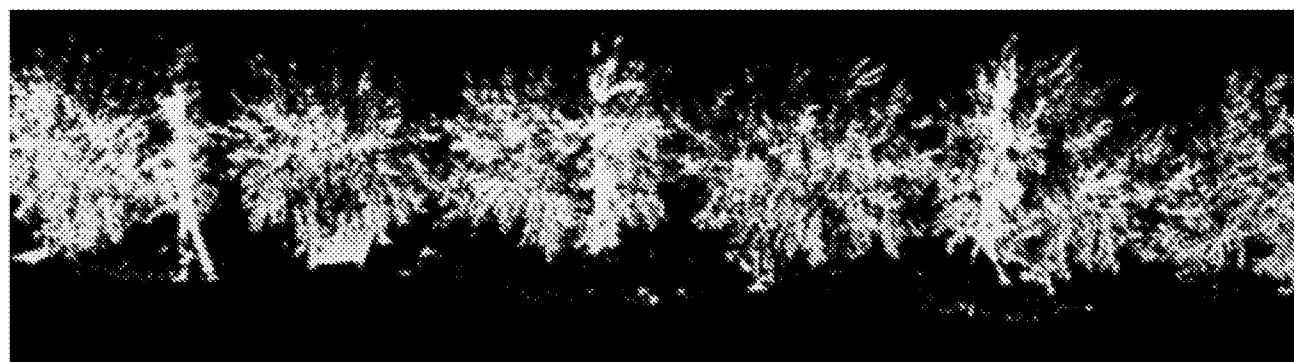


FIGURE 42

31/46

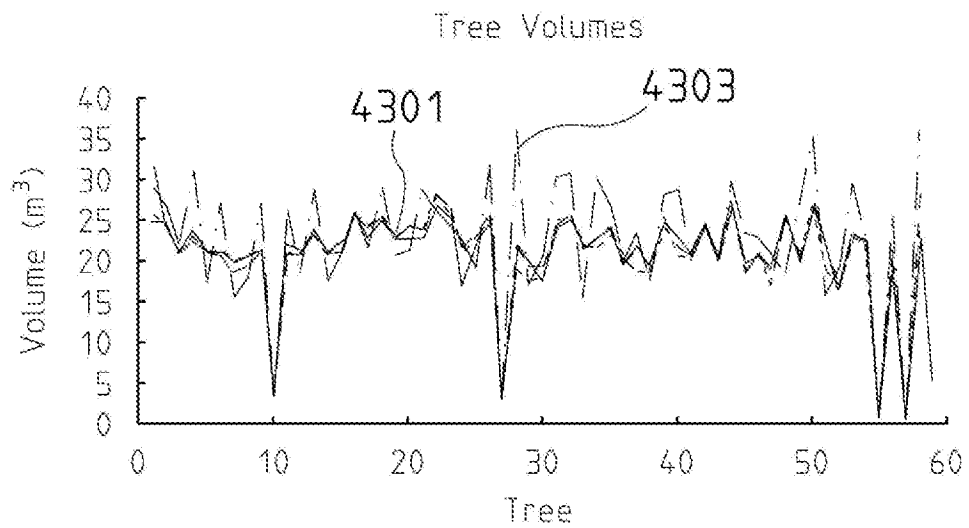


FIGURE 43

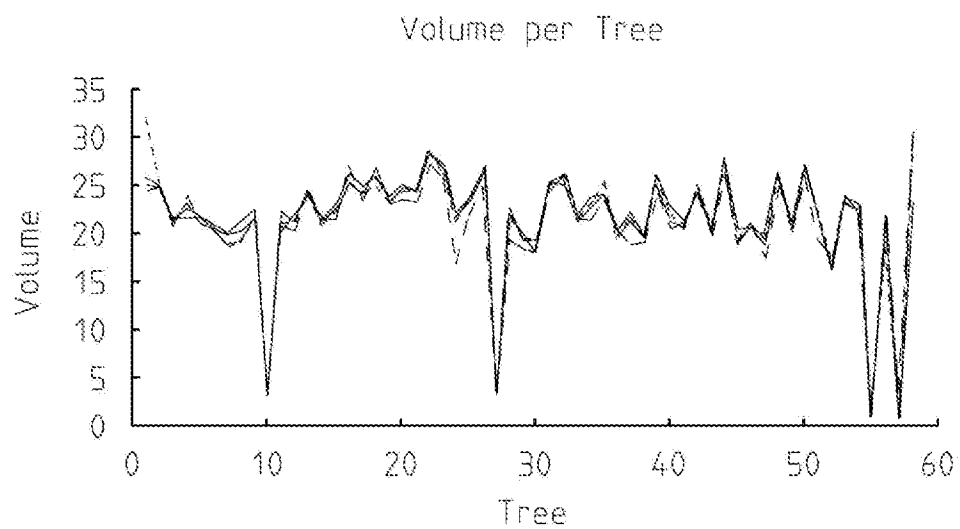


FIGURE 44

32/46

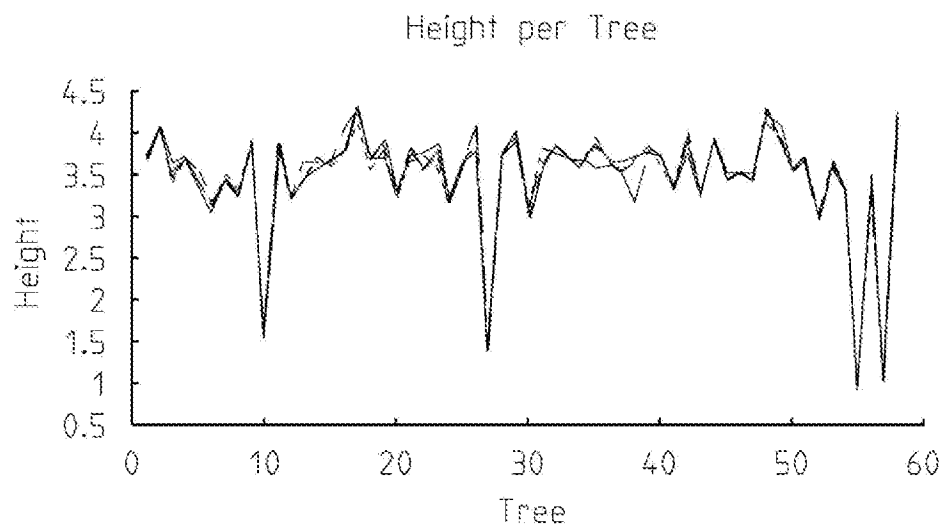


FIGURE 45

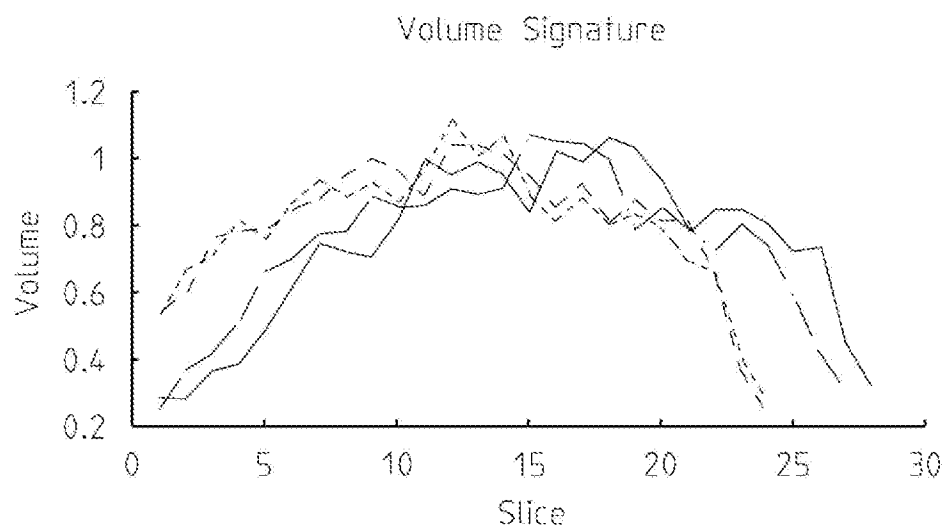


FIGURE 46

33/46

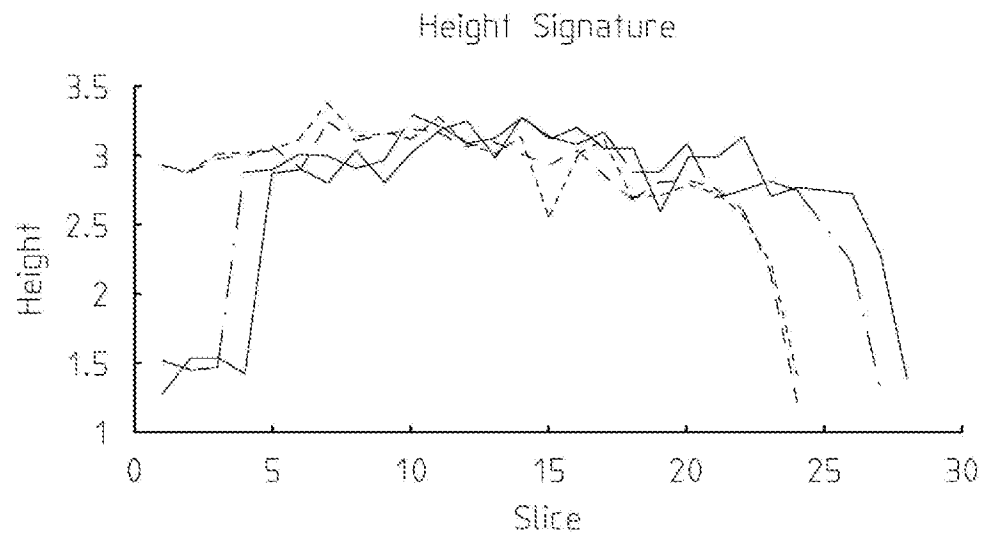


FIGURE 47

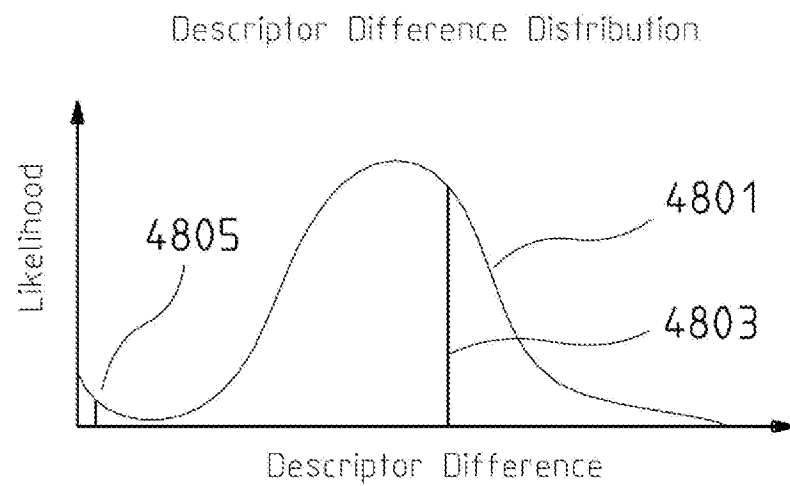


FIGURE 48

34/46

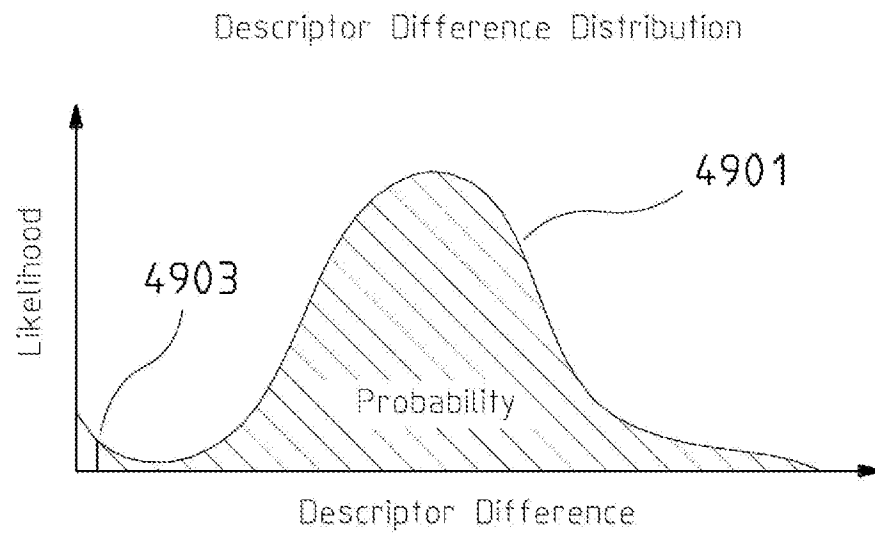


FIGURE 49

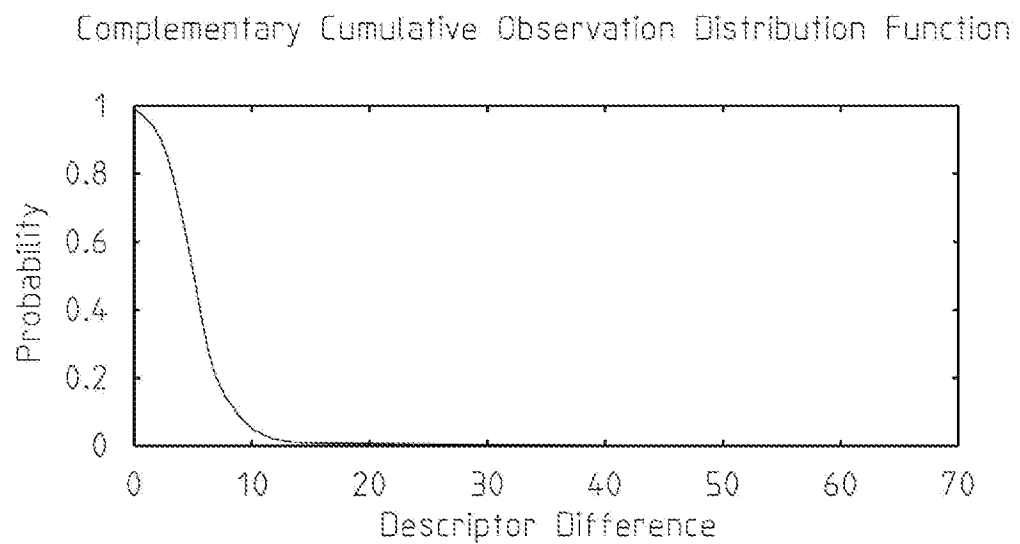


FIGURE 50

35/46

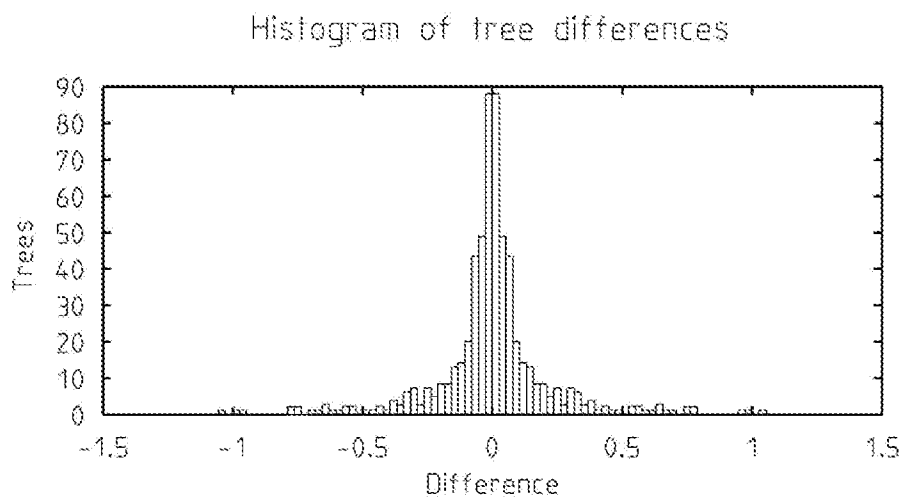


FIGURE 51

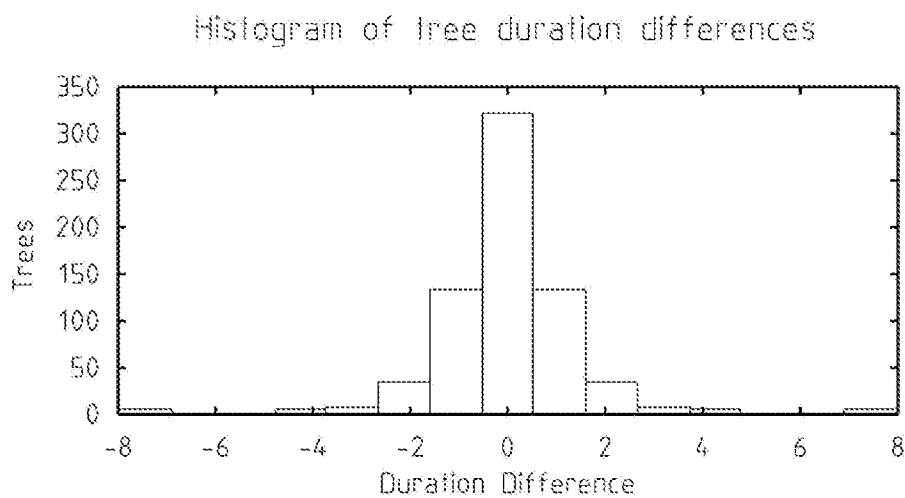


FIGURE 52

36/46

Histogram of differences between
real measurements of the same tree

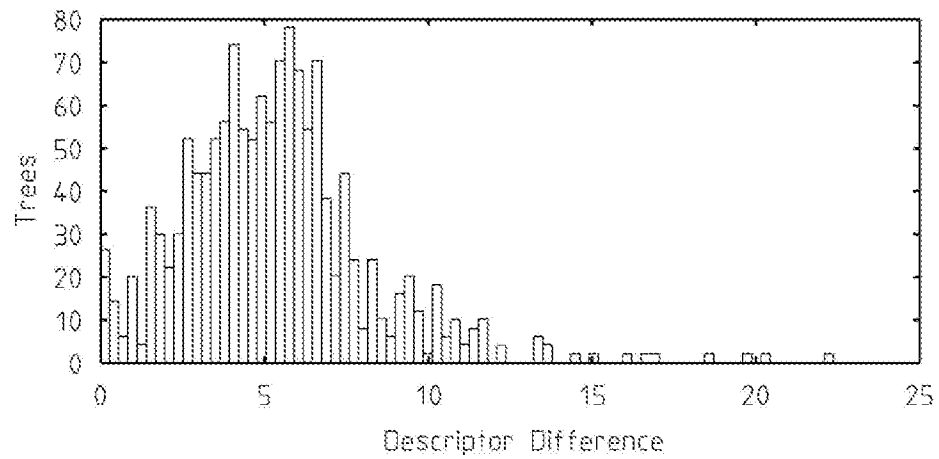


FIGURE 53

Histogram of differences between
simulated measurements of the same tree

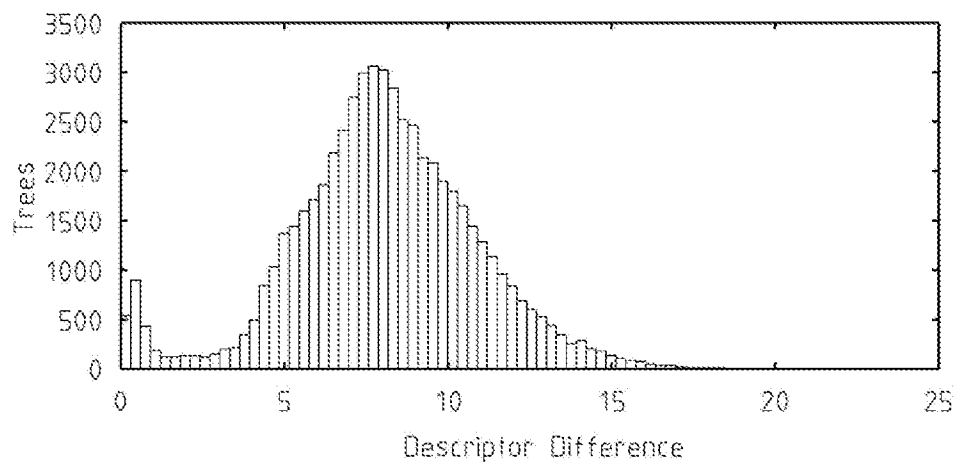


FIGURE 54

37/46

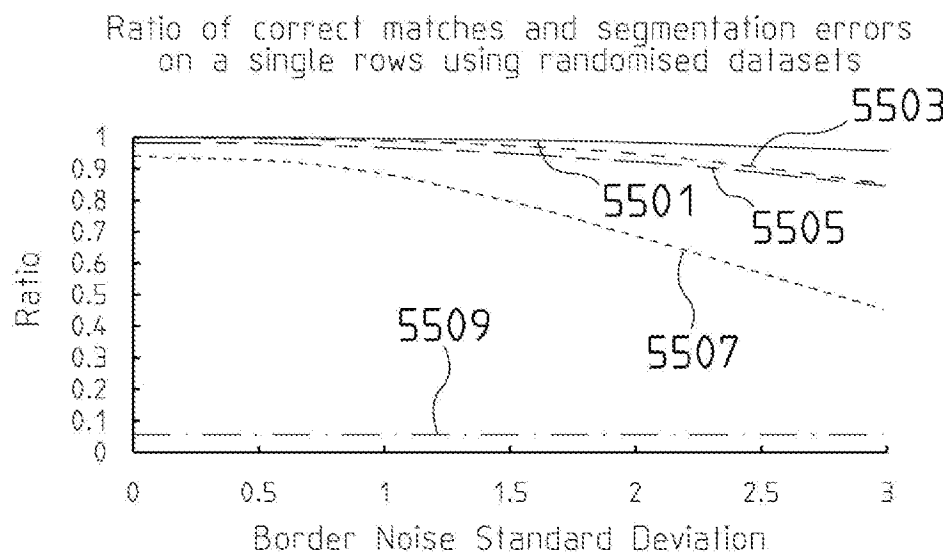


FIGURE 55

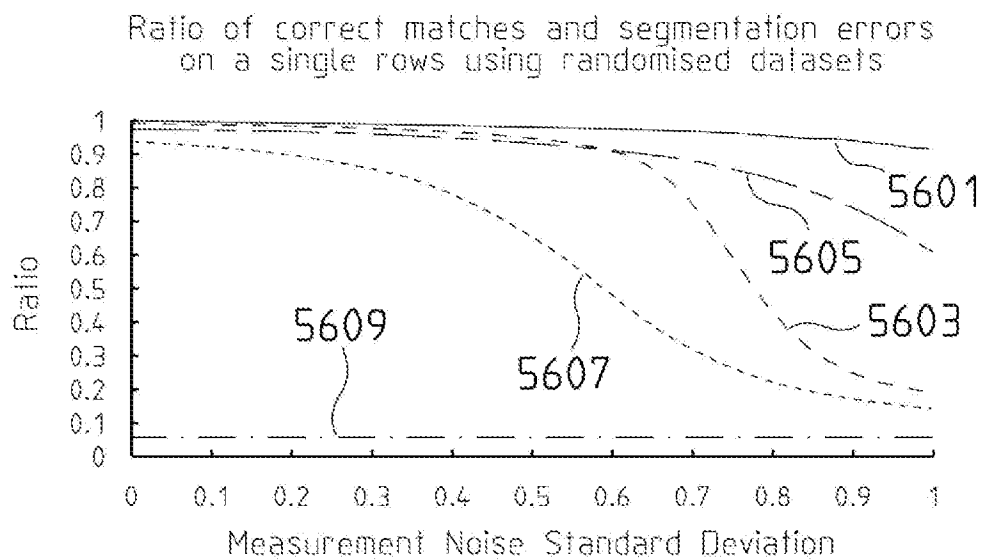


FIGURE 56

38/46

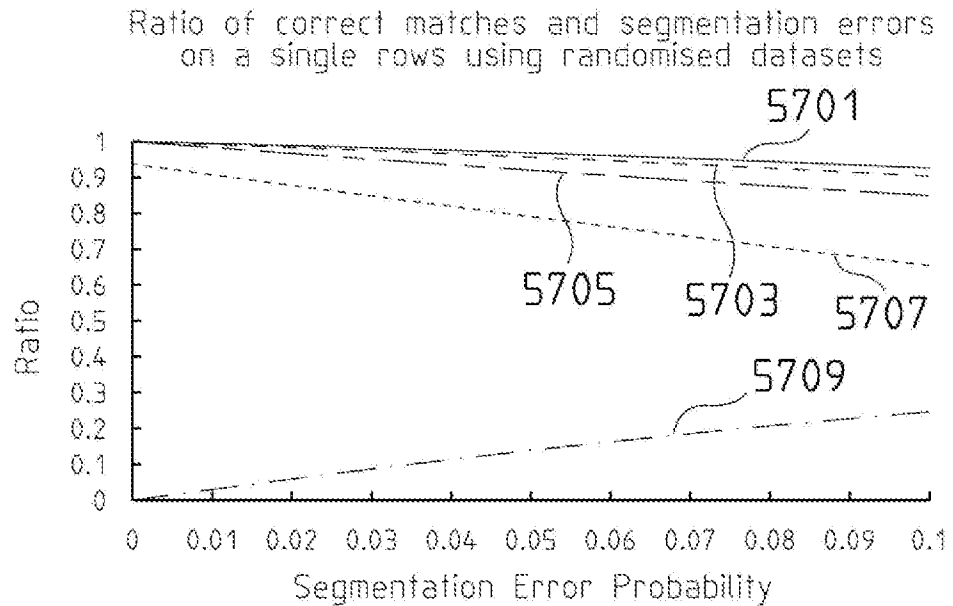


FIGURE 57

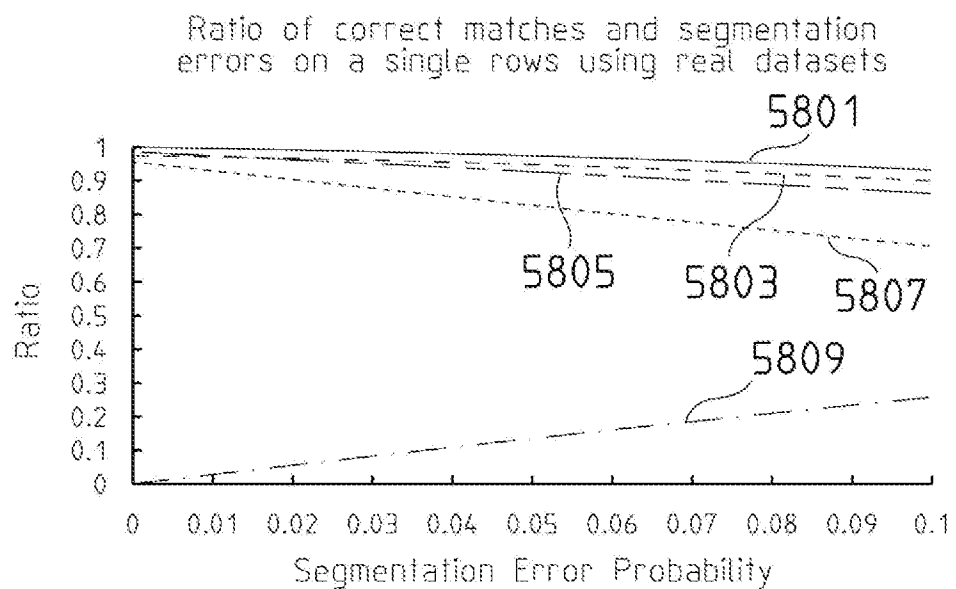


FIGURE 58

39/46

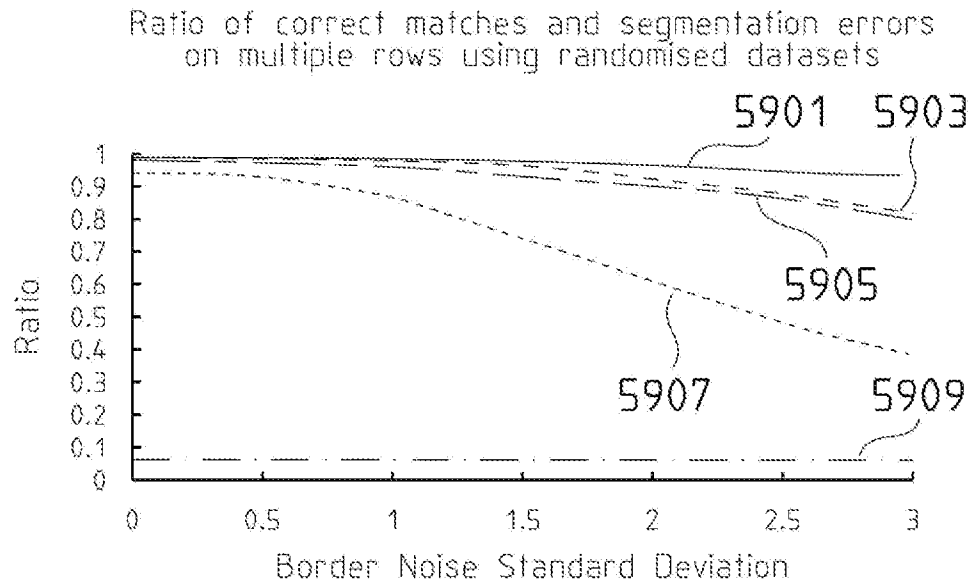


FIGURE 59

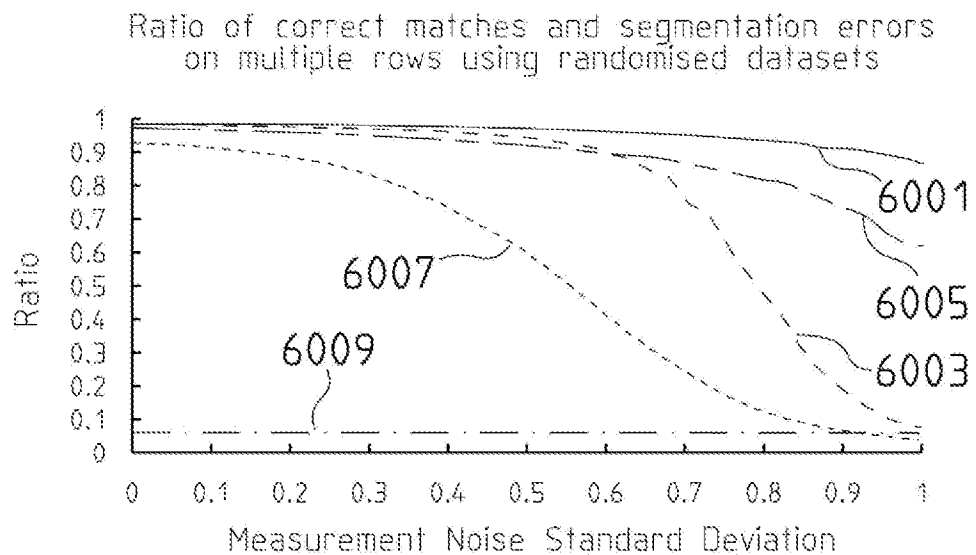


FIGURE 60

40/46

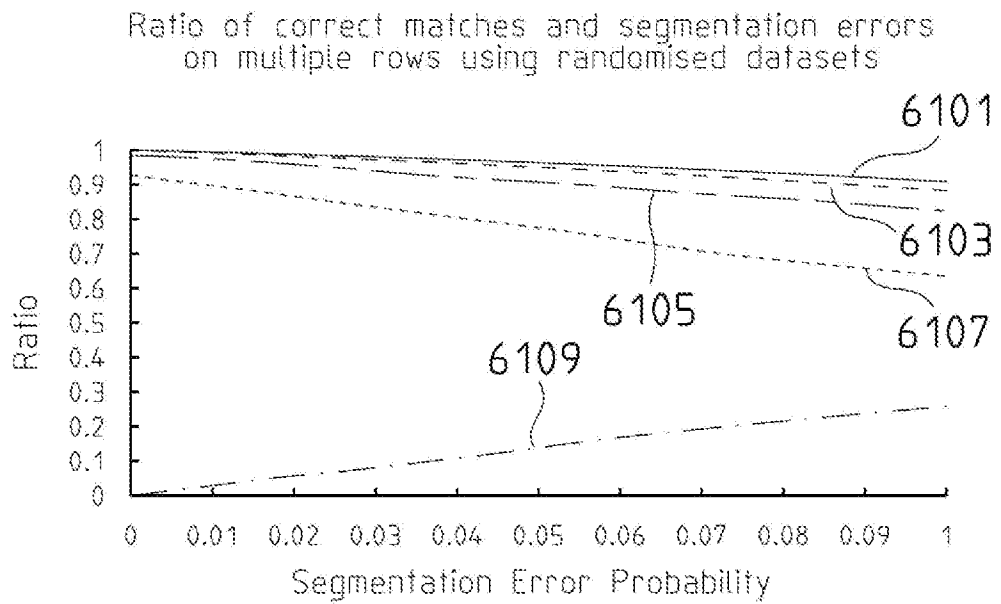


FIGURE 61

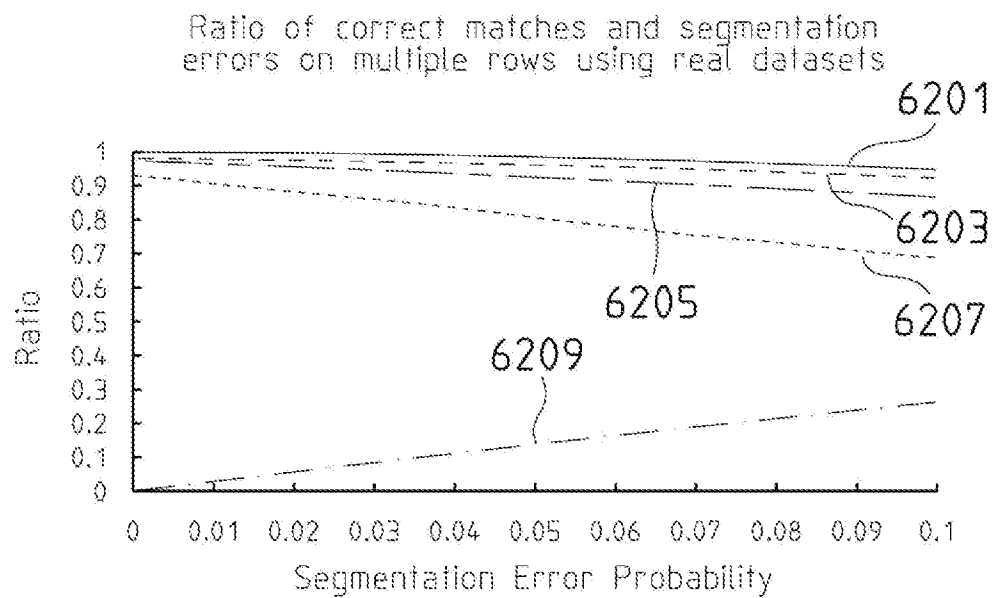


FIGURE 62

41/46

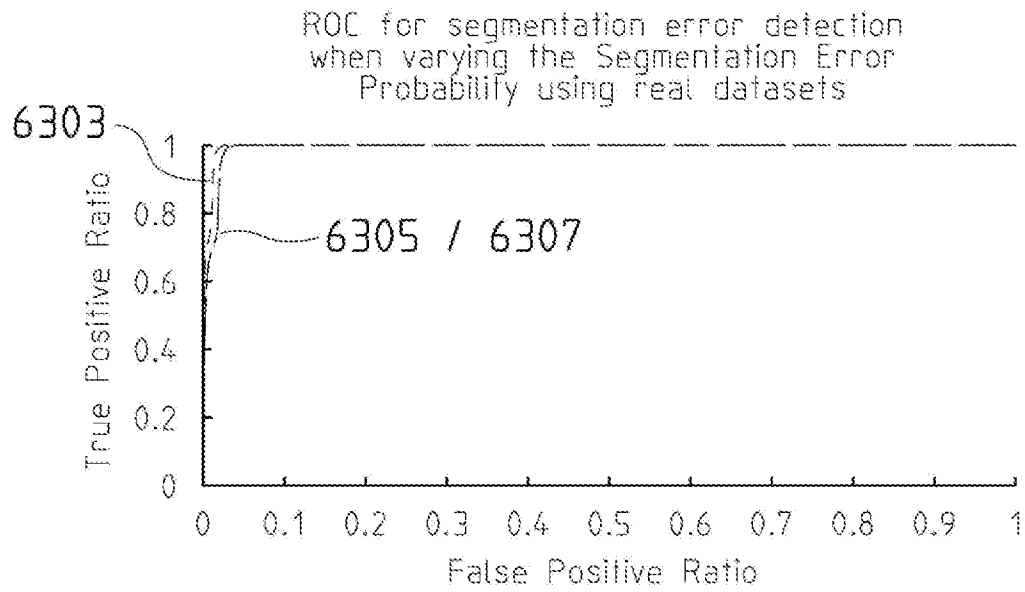


FIGURE 63

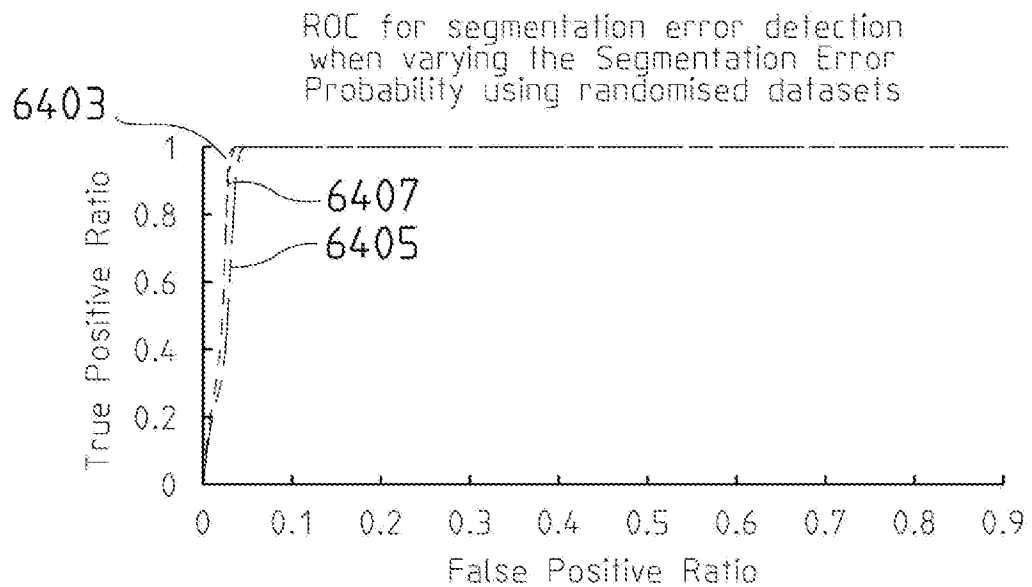


FIGURE 64

42/46

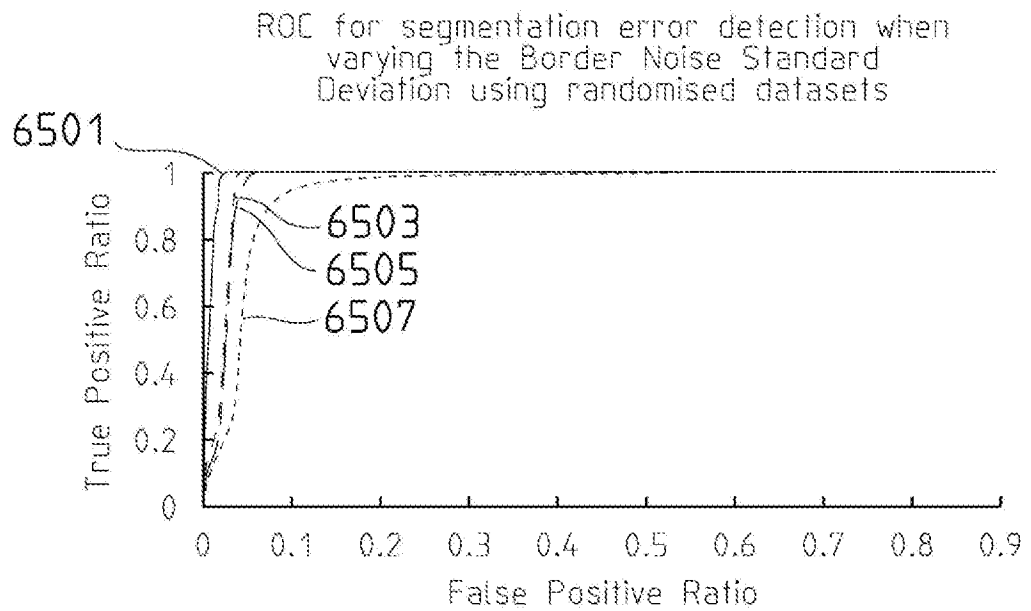


FIGURE 65

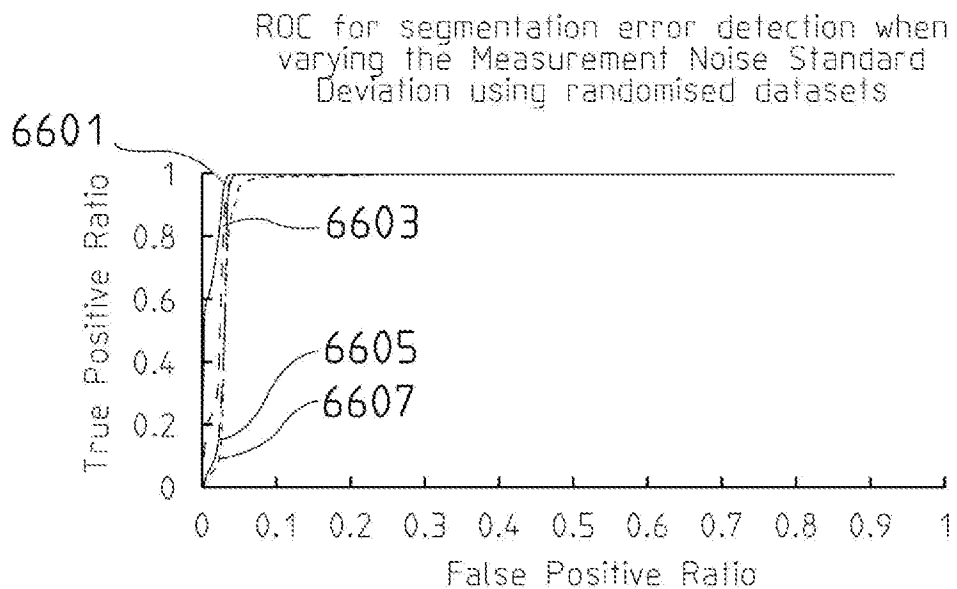


FIGURE 66

43/46

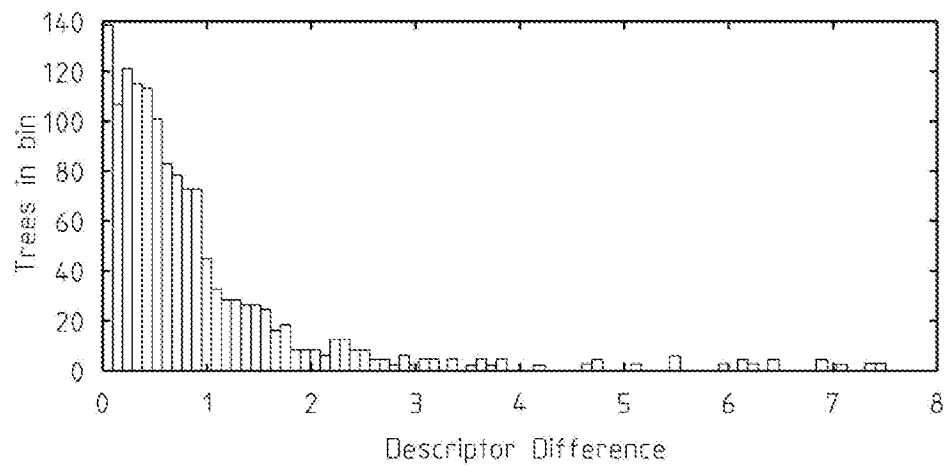
Histogram of differences between
measurements of the same tree

FIGURE 67

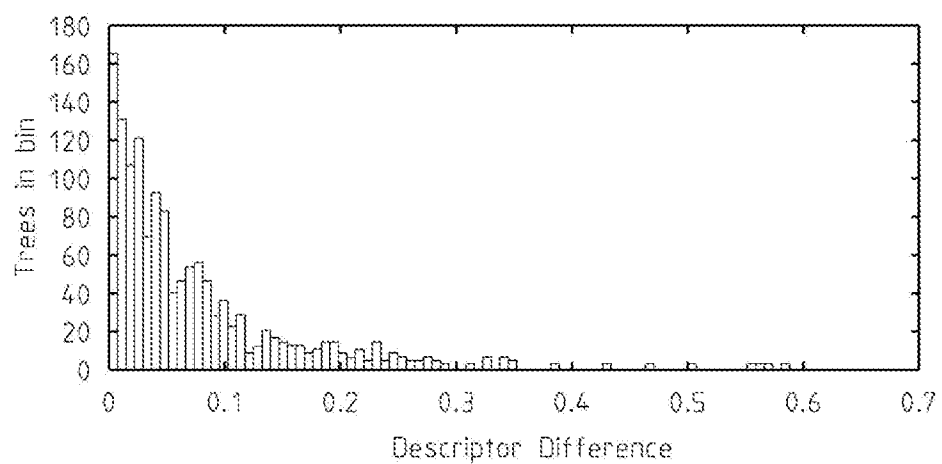
Histogram of differences between
measurements of the same tree

FIGURE 68

44/46

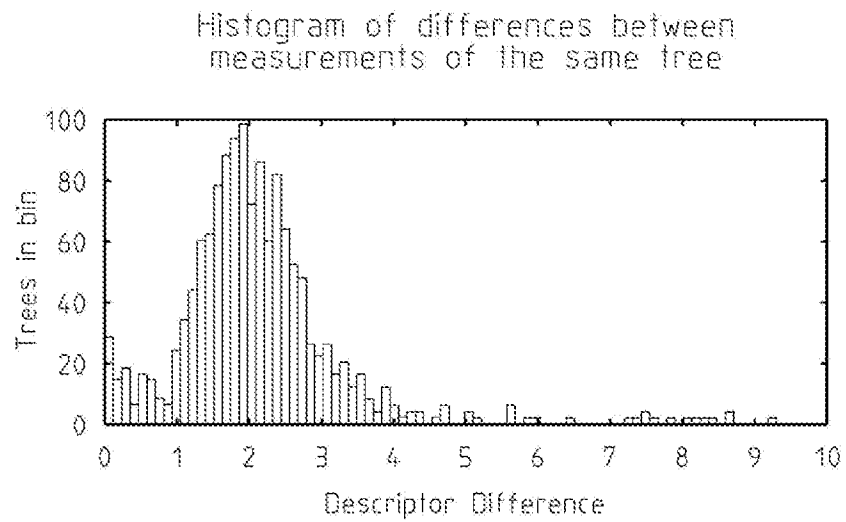


FIGURE 69

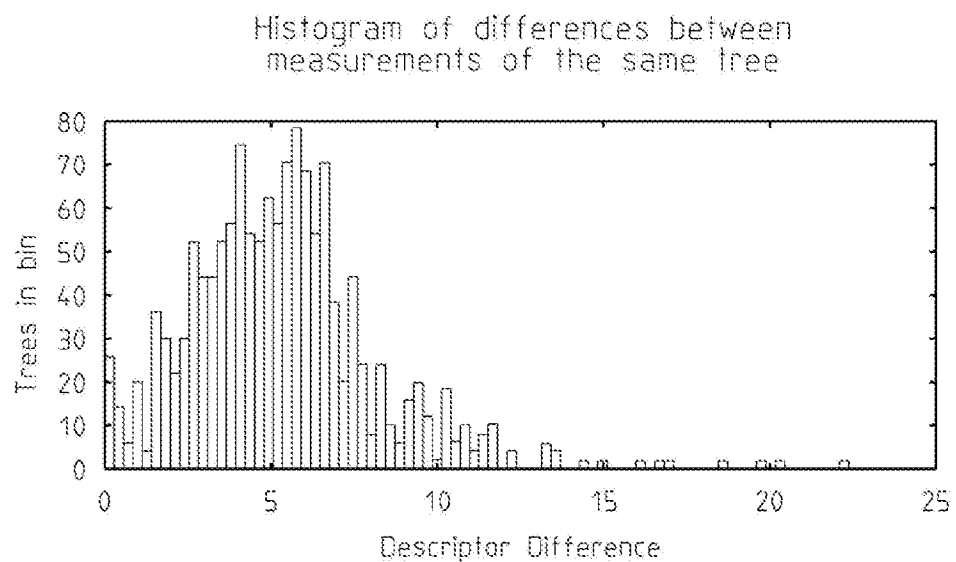


FIGURE 70

45/46

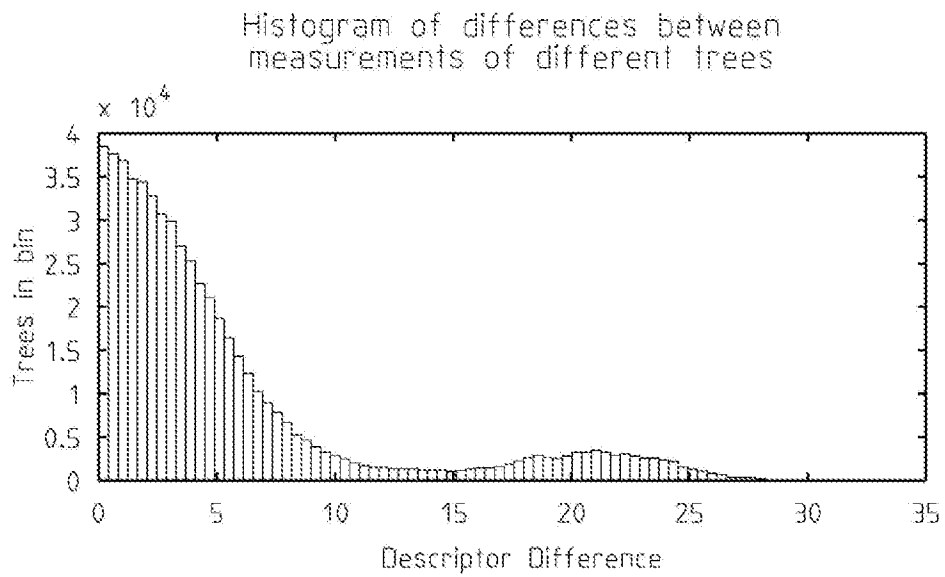


FIGURE 71

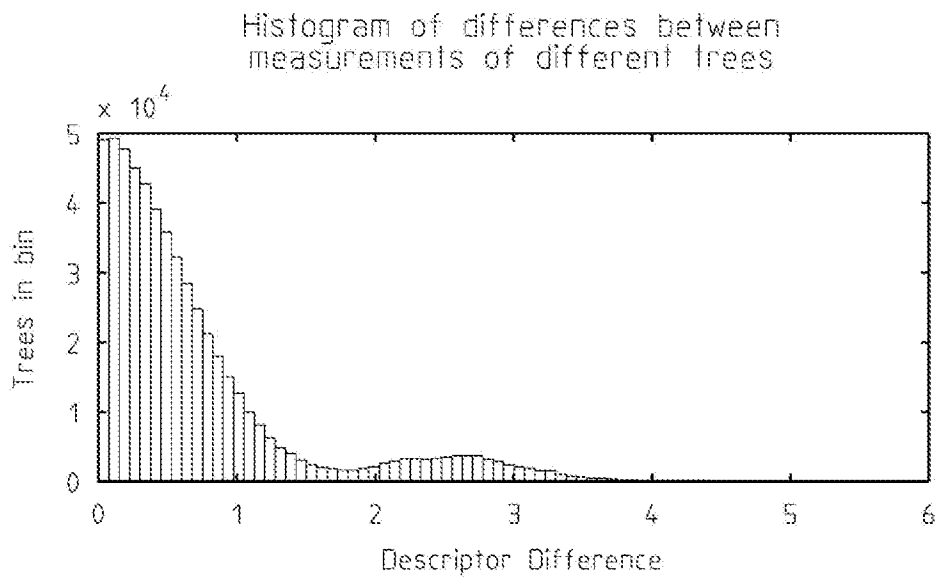


FIGURE 72

46/46

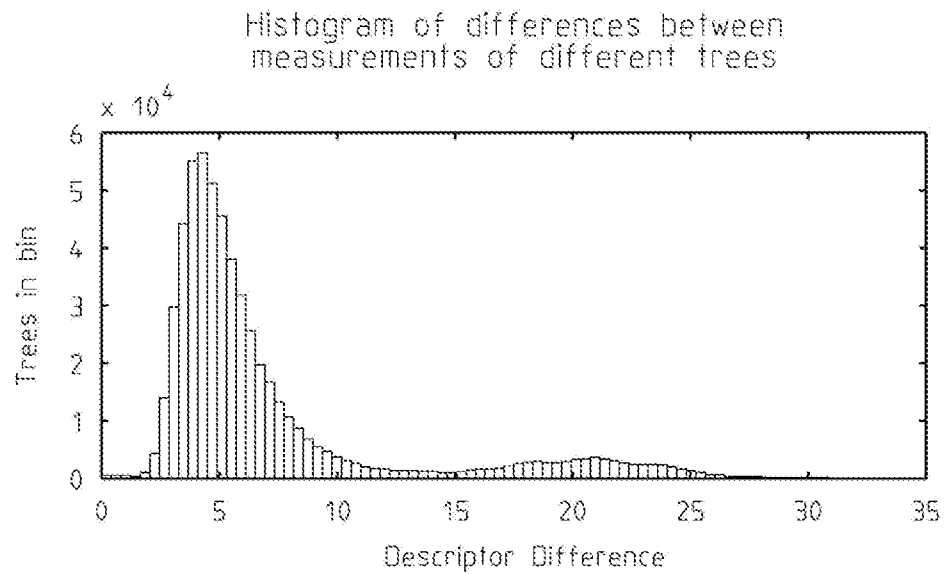


FIGURE 73

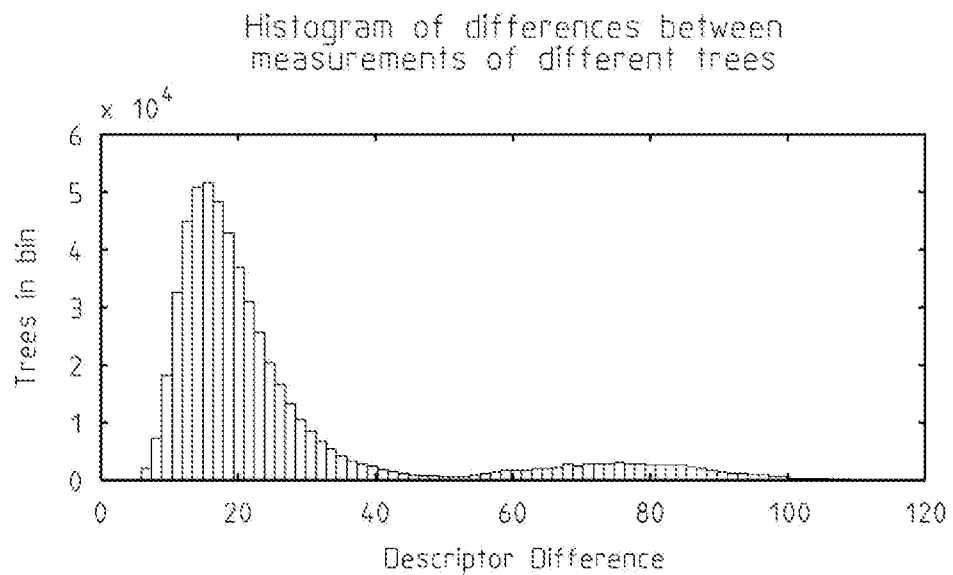


FIGURE 74

INTERNATIONAL SEARCH REPORT

International application No.

PCT/AU2014/000829**A. CLASSIFICATION OF SUBJECT MATTER****G05D 1/00 (2006.01) G05D 3/00 (2006.01) G01C 21/00 (2006.01)**

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPODOC, WPI: Keywords (AGRICULTURE+, CROP+, ORCHARD+, PLANTATION+, VEGETATION+, FOREST+, (NATURA+, ELEMENT+, ROBOT+, AUTOMAT+, VEHICLE+, OR MACHIN+, ENVIRONM+, SURROUND+, LOCAT+, LOCALI_ATION+, DETERMIN+, POSITION+, MARKOV+, SEQUENC+); GOOGLE PATENTS (Prior Art Finder) & PATENTLENS: Keywords: (G05D1, G05D3, G01C21, G01C22, G06T7, AGRICULTURE, CROP, ORCHARD, PLANTATION, VEGETATION, FOREST, NATURAL, ELEMENT, ROBOT, AUTOMATE, AUTONOMOUS, VEHICLE, MACHINE, ENVIRONMENT SURROUND, LOCATION, POSE, LOCALISATION, LOCALIZATION, DETERMINE, POSITION, KALMAN, MARKOV, HIDDEN MARKOV, PARTICLE FILTER, SEQUENCE, FEATURES, SIGNATURES, DESCRIPTORS, SEGMENTS, SEGMENTATION, CHARACTERISATION, LIDAR, ADAPTIVE, MAPPING, UNDERWOOD, NIETO, JAGBRANT, SUKKARIEH, THE UNIVERSITY OF SYDNEY, & like terms)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
	Documents are listed in the continuation of Box C	



Further documents are listed in the continuation of Box C



See patent family annex

* Special categories of cited documents:	
"A" document defining the general state of the art which is not considered to be of particular relevance	"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"E" earlier application or patent but published on or after the international filing date	"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"O" document referring to an oral disclosure, use, exhibition or other means	"&" document member of the same patent family
"P" document published prior to the international filing date but later than the priority date claimed	

Date of the actual completion of the international search
3 November 2014Date of mailing of the international search report
03 November 2014**Name and mailing address of the ISA/AU**

AUSTRALIAN PATENT OFFICE
PO BOX 200, WODEN ACT 2606, AUSTRALIA
Email address: pct@ipaustalia.gov.au

Authorised officer

Hendrik Liebenberg
AUSTRALIAN PATENT OFFICE
(ISO 9001 Quality Certified Service)
Telephone No. 0262104066

INTERNATIONAL SEARCH REPORT		International application No.
C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT		PCT/AU2014/000829
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	“Semantic place classification and mapping for autonomous agricultural robots”, U. Weiss and P. Biber. In IROS 2010 workshop: Semantic Mapping and Autonomous Knowledge Acquisition, 2010. The whole document, in particular see "I. Introduction", "III. Semantic Place Classification and Mapping", "IV. Experiments", and "V. Conclusions and Future Work".	1-34
Y	“Orchard tree modeling for advanced sprayer control and automatic tree inventory”, C. Wellington, J. Campoy, L. Khot, and R. Ehsani, in Intelligent Robots and Systems (IROS)workshop on Agricultural Robotics, IEEE/RSJ International Conference on, 2012. The whole document, in particular see “III. TREE INVENTORY”.	1-34
Y	“Markov localization for mobile robots in dynamic environments”, D. Fox, B. Wolfram, and S. Thrun, in Journal of Artificial Intelligence Research 11, pp. 391-427, 1999. The whole document.	1-34
Y	“Mobile Robot Navigation based on localisation using Hidden Markov Models”, S. Rahman and A. Zelinsky, in Australasian Conference on Robotics and Automation (ACRA), 1999. The whole document.	1-34
A	“Autonomous Robot Supervision using Fault Diagnosis and Semantic Mapping in an Orchard”, Blanke, Mogens; Blas, Morten Rufus; Hansen, Søren; Andersen, Jens Christian; Caponetti, Fabio. Fault Diagnosis in Robotic and Industrial Systems. iConcept Press Ltd, 2012. The whole document.	1-34

Form PCT/ISA/210 (fifth sheet) (July 2009)