

REMOTE DETERMINATION OF CONTAINERS IN GEOGRAPHICAL REGION

Inventors:

Adam Wigen Kraft

Boris Aleksandrovich Babenko

Alexander Bogdanov Avtanski

Daniel Michael Sammons

Jasper Lin

Jason D. Lohn

CROSS REFERENCE TO RELATED APPLICATION

[0001] This application is a continuation-in-part of co-pending U.S. Application No. 15/470,543, filed March 27, 2017, which claims the benefit of U.S. Provisional Application No. 62/320,387, filed on April 8, 2016, which is incorporated by reference in its entirety.

TECHNICAL FIELD

[0002] This disclosure relates generally to image processing of low resolution images and, in particular, to identifying remote objects in a geographical area using low resolution images captured by an aerial imaging device.

BACKGROUND

[0003] Several applications analyze aerial images to identify objects in the images, for example, various objects in aerial images captured by satellites. Analysis of high resolution images can be performed using relatively simple techniques. Obtaining high resolution aerial images typically requires use of large, expensive satellites and results. These satellites typically require a significant amount of resources. For example, such satellites carry sophisticated and

expensive equipment such as high spatial resolution cameras, expensive transponders, and advanced computers. Other factors that contribute to the cost associated with expensive imaging satellites are the launch cost and maintenance. Expensive high spatial resolution imaging satellites must be monitored from a ground facility, which requires expensive manpower. These satellites are also susceptible to damage or costly downtimes. The high launch and development costs of expensive imaging satellites leads to a slowdown in the introduction of new or upgraded satellite imagery and communication services for object detection.

[0004] Cheaper low spatial resolution imaging satellites may be used for capturing images. However, such satellites and provide unclear images. In low-resolution imagery, objects such as containers or tanks are typically not clearly identifiable and often appear as blobs containing a few adjacent pixels. In other instances, such as in infrared band imagery, the images may be completely invisible to humans.

SUMMARY

[0004a] In a first aspect, the present disclosure provides a method for processing images from an aerial imaging device, the method comprising: detecting by a machine learning model, using an image of an object of interest, an amount of cloud cover present in the image of the object of interest, the machine learning model configured to differentiate features corresponding to the cloud cover from background features and features corresponding to the object of interest; responsive to the amount of cloud cover present in the image of the object of interest being less than a cloud detection threshold, extracting, from the image of the object of interest, parameters describing at least an azimuth angle of the aerial imaging device; generating, from the parameters, idealized images of the object of interest, wherein each idealized image corresponds to a distinct filled volume of the object of interest; matching the image of the object of interest to

each idealized image to determine a filled volume of the object of interest; and transmitting information corresponding to the determined filled volume of the object of interest to a user device.

[0004b] In a second aspect, the present disclosure provides a method for processing images from an aerial imaging device, the method comprising: detecting by a machine learning model, using an image of an object of interest, an amount of cloud cover present in the image of the object of interest; responsive to the amount of cloud cover present in the image of the object of interest being less than a threshold, extracting, from the image of the object of interest, parameters describing at least an azimuth angle of the aerial imaging device; determining, from the parameters, a filled volume of the object of interest; and transmitting information corresponding to the determined filled volume of the object of interest to a user device.

[0004c] In a third aspect, the present disclosure provides a non-transitory computer-readable storage medium comprising instructions executable by a processor, the instructions when executed by the processor cause the processor to: detect by a machine learning model, using an image of an object of interest, an amount of cloud cover present in the image of the object of interest; responsive to the amount of cloud cover present in the image of the object of interest being less than a cloud detection threshold, extract, from the image of the object of interest, parameters describing at least an azimuth angle of the aerial imaging device; determine, from the parameters, a filled volume of the object of interest; and transmit information corresponding to the determined filled volume of the object of interest to a user device.

BRIEF DESCRIPTION OF THE DRAWINGS

[0005] The disclosed embodiments have advantages and features which will be more readily apparent from the detailed description, the appended claims, and the accompanying figures (or

drawings). A brief introduction of the figures is below.

[0006] Figure (FIG.) 1 illustrates a block diagram of an example system environment in which a remote container analysis system operates, in accordance with an embodiment.

[0007] FIG. 2 illustrates a block diagram of an example system architecture for the remote container analysis system, in accordance with an embodiment.

[0008] FIG. 3A illustrates an example positive training set for the remote container analysis system, in accordance with an embodiment.

[0009] FIG. 3B illustrates an example negative training set for the remote container analysis

system, in accordance with an embodiment.

[0010] FIG. 3C illustrates an example positive training set for cloud cover detection, in accordance with an embodiment.

[0011] FIG. 3D illustrates an example negative training set for cloud cover detection, in accordance with an embodiment.

[0012] FIG. 4A illustrates an example process for training a machine learning model in the remote container analysis system, in accordance with an embodiment.

[0013] FIG. 4B illustrates an example Precision vs. Recall curve for training a machine learning model to perform cloud cover detection, in accordance with an embodiment.

[0014] FIG. 5 illustrates an example process for the remote container analysis system for identifying remote objects, in accordance with an embodiment.

[0015] FIG. 6A illustrates an example process for the remote container analysis system for determining the filled volume of remote objects, in accordance with an embodiment.

[0016] FIG. 6B illustrates an example process for cloud cover detection and determining the filled volume of remote objects, in accordance with an embodiment.

[0017] FIG. 7 illustrates an example synthesis of an idealized image, in accordance with an embodiment.

[0018] FIG. 8 illustrates a set of example circle projection equations, in accordance with an embodiment.

[0019] FIG. 9 illustrates a set of example idealized images, in accordance with an embodiment.

[0020] FIG. 10A illustrates an example received image of a container, in accordance with an embodiment.

[0021] FIG. 10B illustrates an example image gradient for a received image, in accordance with an embodiment.

[0022] FIG. 10C illustrates an example outline of the top rim of a container in an idealized image, in accordance with an embodiment.

[0023] FIG. 10D illustrates an example outline of a shadow on the inner surface of a container in an idealized image, in accordance with an embodiment.

[0024] FIG. 11 is a block diagram illustrating components of an example machine able to read instructions from a machine-readable medium and execute them in a processor or controller.

DETAILED DESCRIPTION

[0025] The Figures (FIGs.) and the following description relate to preferred embodiments by way of illustration only. It should be noted that from the following discussion, alternative embodiments of the structures and methods disclosed herein will be readily recognized as viable alternatives that may be employed without departing from the principles of what is claimed.

[0026] Reference will now be made in detail to several embodiments, examples of which are illustrated in the accompanying figures. It is noted that wherever practicable similar or like reference numbers may be used in the figures and may indicate similar or like functionality. The figures depict embodiments of the disclosed system (or method) for purposes of illustration only. One skilled in the art will readily recognize from the following description that alternative embodiments of the structures and methods illustrated herein may be employed without departing from the principles described herein.

CONFIGURATION OVERVIEW

[0027] Disclosed by way of example embodiments are systems, methods and/or computer program products (e.g., a non-transitory computer readable storage media that stores instructions

executable by one or more processing units) for identifying remote objects, such as cylindrical containers or tanks with floating roof structures over large geographic regions (e.g., a country), and determining the filled volume of remote objects.

[0028] In one example embodiment, a remote container analysis system receives an image of a geographical area from an aerial imaging device, such as a satellite, drone, or other aerial configured imaging system. The image may have a low resolution, e.g., a 15 meter (m) per pixel Landsat image in the panchromatic band. The system transmits the image to a machine learning model to identify an area of interest containing an object of interest in the image, such as a cylindrical container or tank with a floating roof structure. Such tanks are typically found in clusters or “tank farms.” The system receives a second image of the geographical area. The second image has a resolution higher than the first image, e.g., 50 cm per pixel. The system may transmit the second image to the machine learning model to determine a likelihood that the area of interest contains the object of interest. If the likelihood is below a threshold, such that the area of interest does not contain the object of interest (false positive) the machine learning model filters out features corresponding to the area of interest in images having the first resolution. If the likelihood exceeds the threshold, a visual representation identifying the object of interest is sent to a user device.

[0029] In another example embodiment, the system may receive an image of the object of interest. The system extracts a parameter vector from the image. The parameter vector may include a parameter describing an elevation angle of the aerial imaging device. The system performs image analysis on the image to determine a height and a width of the object of interest. The system generates idealized image templates of the object of interest using the extracted parameter vector and the determined height and width of the object of interest. Each idealized

image corresponds to a distinct filled volume of the object of interest, such as 30%, 70%, etc.

The system matches the received image of the object of interest to each idealized image to determine the filled volume of the object of interest by performing a dot product between pixels of the received image and pixels of the idealized image. The system transmits information corresponding to the determined filled volume of the object of interest to a user device.

[0030] In another example embodiment, the system may detect by a machine learning model, using an image of an object of interest, an amount of cloud cover present in the image of the object of interest. The machine learning model is configured to differentiate features corresponding to the cloud cover from background features and features corresponding to the object of interest. Responsive to the amount of cloud cover present in the image of the object of interest being less than a cloud detection threshold, the system extracts, from the image of the object of interest, parameters describing at least an azimuth angle of an aerial imaging device. From the parameters, idealized images of the object of interest are generated. Each idealized image corresponds to a distinct filled volume of the object of interest. The image of the object of interest is matched to each idealized image to determine a filled volume of the object of interest. Information corresponding to the determined filled volume of the object of interest is transmitted to a user device.

EXAMPLE SYSTEM ENVIRONMENT

[0031] Referring now to Figure (FIG.) 1, it illustrates a block diagram of an example system environment in which a remote container analysis system 101 operates, in accordance with an embodiment. The example system environment shown in FIG. 1 may include an aerial imaging device 110, the remote container analysis system 101, and a user device 120.

[0032] The aerial imaging device 110 shown in FIG. 1 may be a satellite, drone, or other

aerial configured imaging system, capable of capturing low resolution images. The images may correspond to the same spectral band or different spectral bands, where a spectral band corresponds to a range of wavelengths of light. Example spectral bands include the red spectral band, the green spectral band, the blue spectral band, the infrared spectral band, and the panchromatic spectral band. It is noted that low resolution images have resolution significantly less (e.g., 15 m per pixel) than high resolution images (e.g., 50 cm per pixel).

[0033] The remote container analysis system 101 may contain an image store 102, an optional feature extraction module 104, a machine learning model 106, a container analysis module 107, a parameter extraction module 105, and a template generation module 103. The image store 102 shown in FIG. 1 may store images received from the aerial imaging device 110, as illustrated and described below with reference to FIG. 2. The feature extraction module 104 may optionally extract feature vectors from the images received from the aerial imaging device 110. For example, a feature vector may include aggregate values based on pixel attributes of pixels in the images, as illustrated and described below with reference to FIG. 4A. The remote container analysis system 101 optionally transmits the feature vector to the machine learning model 106 to identify objects of interest in the images as illustrated and described below with reference to FIG. 5 or to perform cloud cover detection as illustrated and described below with reference to FIG. 6B. The container analysis module 107 may analyze a pattern related to the identified objects of interest in the images, for example, times of capture of images, counts of the images, filled volumes of the images.

[0034] The parameter extraction module 105 may extract parameters from the image, for example, a parameter describing an azimuth angle of the aerial imaging device 110, a parameter describing an elevation angle of the sun, and a parameter describing an azimuth angle of the sun.

The parameters are used by the template generation module 103 to generate idealized image templates of the object of interest using the extracted parameters, as illustrated and described below with reference to FIG. 9. Each idealized image corresponds to a distinct filled volume of the object of interest, for example, 35%.

[0035] The remote container analysis system 101 may interact with the user device 120 shown in FIG. 1. The user device 120 may be a computing device capable of receiving user input as well as transmitting and/or receiving data via a network. In one example embodiment, a user device 120 may be a conventional computer system, such as a desktop or laptop computer. Alternatively, a user device 120 may be a device having computer functionality, such as a personal digital assistant (PDA), a mobile telephone, a smartphone, a tablet, or another suitable device. The remote container analysis system 101 may transmit a visual representation of the object of interest to the user device 120 or output a visual representation of the object of interest to a user interface, for example, through graphical icons, graphical overlays, and other visual indicators.

EXAMPLE SYSTEM ARCHITECTURE

[0036] Turning now to FIG. 2, it illustrates a block diagram of an example system architecture for the remote container analysis system 101, in accordance with an embodiment. The system architecture shown in FIG. 2 may include an external system interface 201, the image store 102, the parameter extraction module 105, the template generation module 103, the optional feature extraction module 104, an optional feature store 202, the machine learning model 106, a machine learning training engine 203, an image analysis module 204, a template matching module 205, the container analysis module 107, and a container pattern store 206.

[0037] The external system interface 201 shown in FIG. 2 may be a dedicated hardware or

software networking device that receives data packets representing images from the aerial imaging device 110. The external system interface 201 may forward data packets representing visual representation of the objects of interest or information corresponding to the determined filled volume of the objects of interest from the remote container analysis system 101 via a network to user devices 120. In one example, the external system interface 201 forwards data packets at high speed along the optical fiber lines of the Internet backbone. In another example, the external system interface 201 exchanges routing information using the Border Gateway Protocol (BGP) and may be an edge router, a border router, or a core router.

[0038] The image store 102 shown in FIG. 2 may store images received from the aerial imaging device 110. To make scanning over a large area practical and efficient, lower resolution imagery is used for the first phase of the system. An example of such imagery is 15 m / pixel Landsat imagery (in the panchromatic band). The first phase of the system is designed to have high recall at the cost of lower precision. In a second phase, higher resolution imagery, e.g., 50 cm / pixel may be used to train the machine learning model 106 to filter out false alarms, as described below with reference to FIG. 4A. The parameter extraction module 105, the optional feature extraction module 104, and the image analysis module 204 may retrieve images stored by the image store 102 for processing. The image store 102 may be organized as a database or table of images stored on one or more of removable or non-removable memory cards, tape cassettes, zip cassettes, and computer hard drives. In one embodiment, the image store 102 may include multiple data fields, each describing one or more attributes of the images. In one example, the image store 102 contains, for a single image, the time of capture, spectral band information, geographical area coordinates, etc.

[0039] The optional feature extraction module 104 may extract feature vectors from the

images in the image store 102. The feature vector may include aggregate values based on pixel attributes of pixels in the images. In an embodiment, the feature extraction module 104 may optionally identify clusters of adjacent pixels using pixel clustering. Within an identified cluster, adjacent pixels may match each other based on a pixel attribute. For example, for a grayscale image, the pixel attribute may be a single number that represents the brightness of the pixel. In this example, the pixel attribute is a byte stored as an 8-bit integer giving a range of possible values from 0 to 255. Zero represents black and 255 represents white. Values in between 0 and 255 make up the different shades of gray. In another example of color images, separate red, green and blue components are specified for each pixel. In this example, the pixel attribute is a vector of three numbers.

[0040] The optional feature extraction module 104 shown in FIG. 2 may identify pixel clusters in the images from the image store 102 by initializing each pixel in an image as a region with the attribute of the pixel. The feature extraction module 104 identifies two adjacent regions having the most similar attribute value. These two regions are merged to form a new region containing all the pixels of the two regions and having the attribute value as the average of the attribute values of the two regions. The feature extraction module 104 repeats the process until there are no similar regions left.

[0041] Other embodiments of the feature extraction module 104 shown in FIG. 2 may use one or a combination of the following: (a) edge / corner detection methods, such as Harris Corner or Canny edge, which find edges or corners in the image to use as candidate features; (b) image gradients, which extract edge strength information; (c) oriented filters, which identify specific shapes; (d) thresholding methods, which use local or global threshold values to extract features; (e) image patch descriptors such as Scale-Invariant Feature Transform (SIFT), Speeded

Up Robust Features (SURF), Features from Accelerated Segment Test (FAST), Binary Robust Independent Elementary Features (BRIEF), Fast Retina Keypoint (FREAK), and Histogram of Oriented Gradients (HOG), which calculate orientation and edge description features at a given image patch.

[0042] The feature extraction module 104 shown in FIG. 2 may perform edge analysis in the received images to identify pixels in images corresponding to the object of interest. The feature extraction module 104 may operate on each pixel location (i, j) in an image. Here, i represents the row value of a pixel location in the image and j represents the column value of the pixel in the image. In one example embodiment, S represents an image and M represents the corresponding object map image output. The function $M(i, j)$ is defined to be 1 whenever location (i, j) in image S corresponds to an object pixel and 0 otherwise. The feature extraction module 104 may identify points in an image at which the pixel attributes change sharply. The points at which pixel attributes change sharply may be organized into a set of curved line segments termed edges. The feature extraction module 104 may perform three steps in the edge analysis process to identify pairs of edges: filtering, enhancement, and detection. The filtering step reduces noise, for example, salt and pepper noise, impulse noise and Gaussian noise in the images. The enhancement emphasizes pixels at locations (i, j) where there is a significant change in the pixel attribute value. In one example, the feature extraction module 104 performs enhancement by computing the gradient magnitude of the image at various pixel locations (i, j) . The detection searches for pixel locations (i, j) that have a gradient value higher than a threshold to detect edge pixels.

[0043] In alternative embodiments, the feature extraction module 104 shown in FIG. 2 may analyze an image to create a probabilistic heat map or blocked image containing an area of

interest where objects of interest are expected. The feature extraction module 104 may be further configured to incorporate other mapping sources, which contain geometric information (e.g., points, lines, and polygons). The feature extraction module 104 may use the geometric information to directly create the probabilistic heat map or in conjunction with other image processing operations, for example, as a line finding algorithm or random forest algorithm, or using machine learning methods such as Support Vector Machines (SVM), neural network, or convolutional neural network (CNN), which requires no feature extraction.

[0044] Referring back to FIG. 2, the feature extraction module 104 reduces the redundancy in images, e.g., repetitive pixel values, to transform an image into a reduced set of features (features vector). The feature vector contains the relevant information from the images, such that objects of interest can be identified by the machine learning model 106 by using this reduced representation instead of the complete initial image. Example features extracted by the feature extraction module 104 are illustrated and described in FIG. 4A. In some example embodiments, the following dimensionality reduction techniques may be used by the feature extraction module 104: independent component analysis, Isomap, Kernel PCA, latent semantic analysis, partial least squares, principal component analysis, multifactor dimensionality reduction, nonlinear dimensionality reduction, Multilinear Principal Component Analysis, multilinear subspace learning, semidefinite embedding, Autoencoder, and deep feature synthesis.

[0045] In alternate embodiments, the machine learning training engine 203 may perform deep learning (also known as deep structured learning or hierarchical learning) to learn data representations, as opposed to using task-specific algorithms. In deep learning, no explicit feature extraction is performed; the features may be implicitly extracted by the machine learning training engine 203. For example, the machine learning training engine 203 and machine

learning model 106 may use a cascade of multiple layers of nonlinear processing units for implicit feature extraction and transformation. Each successive layer uses the output from the previous layer as input. The machine learning model 106 may thus learn in supervised (e.g., classification) and/or unsupervised (e.g., pattern analysis) modes. The machine learning model 106 may learn multiple levels of representations that correspond to different levels of abstraction, wherein the different levels form a hierarchy of concepts. In this manner, the machine learning model 106 may be configured to differentiate features corresponding to cloud cover in images from background features and features corresponding to an object of interest.

[0046] The feature store 202 shown in FIG. 2 stores features extracted from received images by the feature extraction module 104. The remote container analysis system 101 retrieves the stored features for training the machine learning model 106. The feature store 202 may be organized as a database or table of images stored on one or more of removable or non-removable memory cards, tape cassettes, zip cassettes, and computer hard drives.

[0047] The remote container analysis system 101 may train the machine learning model 106 using training sets and data from the feature store 202. In one example embodiment, the machine learning model 106 may receive training sets including labeled clusters of pixels corresponding to objects of interest, as illustrated and described below with reference to FIGs. 3A and 3B. The machine learning training engine 203 shown in FIG. 2 may train the machine learning model 106 using training sets to determine scores for clusters of pixels. The score is indicative of a likelihood that the clusters corresponds to objects of interest based on the feature vector. The process followed by the machine learning training engine 203 is illustrated in FIG. 4A. The remote container analysis system 101 may select clusters based on whether the score exceeds a threshold and associates the clusters with objects of interest.

[0048] In alternative example embodiments, the machine learning model 106 shown in FIG. 2 (in the form of a convolutional neural network) may generate an output, without the need for feature extraction, directly from the images. A CNN is a type of feed-forward artificial neural network in which the connectivity pattern between its neurons is inspired by the organization of a visual cortex. Individual cortical neurons respond to stimuli in a restricted region of space known as the receptive field. The receptive fields of different neurons partially overlap such that they tile the visual field. The response of an individual neuron to stimuli within its receptive field can be approximated mathematically by a convolution operation. CNNs are based on biological processes and are variations of multilayer perceptrons designed to use minimal amounts of preprocessing. Advantages of CNNs include the obviation of feature extraction and the use of shared weight in convolutional layers, which means that the same filter (weights bank) is used for each pixel in the layer; this both reduces memory footprint and improves performance.

[0049] The machine learning model 106 may be a CNN that consists of both convolutional layers and max pooling layers. The architecture of the machine learning model 106 may be “fully convolutional,” which means that variable sized input images can be fed into it. The input to the machine learning model 106 may be a panchromatic Landsat image, and the output of the machine learning model 106 may be a per-pixel probability map (i.e., for each pixel in the input image, the machine learning model 106 considers a patch around that pixel and returns the probability that that pixel is part of a tank farm). All but the last convolutional layer in the machine learning model 106 may be followed by in-place rectified linear unit activation. For all convolutional layers, the machine learning model 106 may specify the kernel size, the stride of the convolution, and the amount of zero padding applied to the input of that layer. For the pooling layers the model 106 may specify the kernel size and stride of the pooling.

[0050] The output of the machine learning model 106 (in the form of a CNN) may optionally include pixel clusters, where each pixel cluster includes one or more adjacent pixels in a distinct image of the images, where the adjacent pixels match each other based on a pixel attribute. The output may include a score indicative of a likelihood that the pixel clusters correspond to an object of interest. The output may include one or more pixels locations corresponding to an object of interest. The output may include the number of pixels in each an object of interest. The output may include an association between the pixel clusters and objects of interest.

[0051] The parameter extraction module 105 shown in FIG. 2 may extract a parameter vector from metadata in an image received from the aerial imaging device 110. The parameter vector may include example parameters describing the elevation angle of the aerial imaging device 110. The satellite elevation angle refers to the angle between a line pointing directly towards the satellite and the local horizontal plane. A parameter may describe the time of capture of the received image. A parameter may describe the azimuth angle of the aerial imaging device 110. The azimuth angle is an angular measurement in a spherical coordinate system, which refers to the angle between the line pointing directly towards the satellite and a reference vector pointing North on the reference plane.

[0052] A parameter extracted by the parameter extraction module 105 may describe the elevation angle of the sun. The elevation angle of the sun refers to the angle between a line pointing directly towards the sun and the local horizontal plane. A parameter may describe the azimuth angle of the sun. The azimuth angle of the sun refers to the angle between the line pointing directly towards the sun and a reference vector pointing North on the reference plane. A parameter may describe the geographical location of the center of the bottom of an object of interest in the image. The remote container analysis system 101 operates under the assumption

that some parameters may be inaccurate. Specifically, the system assumes that the location of the object and the satellite angles may not be accurate, but may be processed as described herein.

[0053] The image analysis module 204 retrieves images from the image store 102. The image analysis module 204 may perform image analysis on an image to determine a height and a width of an object of interest in the image. For example, the image analysis module 204 may receive a pixel resolution r of the image of the object of interest. The image analysis module 204 may determine a number h of pixels associated with the height of the object of interest. The image analysis module 204 may determine the height of the object of interest based on the pixel resolution r and the number h of pixels associated with the height of the object of interest as $\text{height} = r \times h$. The image analysis module 204 may determine the number of pixels w associated with the width of the object of interest. The image analysis module 204 may determine the width of the object of interest based on the pixel resolution r and the number of pixels w associated with the width of the object of interest as $\text{width} = r \times w$.

[0054] The image analysis module 204 may crop the received image to position the center of the object of interest in the center of the received image. In embodiments, the image analysis module 204 may automatically remove the outer parts of the image to improve framing, accentuate the object of interest, or change the aspect ratio. The image analysis module 204 may rescale the received image of the object of interest by setting pixels corresponding to shadows and inner surfaces of the object of interest to negative values, e.g., -1, and setting pixels corresponding to the roof of the object of interest to positive values, e.g., +1.

[0055] The template generation module 103 shown in FIG. 2 uses the parameter vector extracted by the parameter extraction module 105 and synthesizes idealized image templates based on trigonometric projection for the geometry of the object of interest for different filled

volume percentages, e.g., 10%, 30%, etc. A set of templates is generated by varying the filled volume percentage. FIG. 9 illustrates a set of idealized images 900 for a cylindrical tank container corresponding to different filled volume percentages. The template generation module 103 generates the idealized images of the object of interest using the extracted parameter vector, the determined height, and the determined width of the object of interest. To allow for inaccuracies in the satellite view angles (elevation and azimuth), the template generation module 103 may perform a sweep over a range of angle values around the angle values extracted by the feature extraction module 104.

[0056] The template generation module 103 assumes that the received image has the object of interest in the center, although some error in the precise location is allowed for by the synthesis process. The template generation module 103 also assumes that the object, including its roof, is light-colored. It assumes that shadows cast by the roof of the object and its top rim, and the inner walls of the object are dark-colored. Idealized image templates are constructed from the position of circles, as illustrated and described below with reference to FIG. 7. The circles correspond to the top rim of the object of interest, the bottom of the object of interest, the arc of the shadow on the inner surface of the object of interest, and the roof of the object of interest. In embodiments illustrated in FIG. 8, the idealized images may be constructed from only the position of the circles corresponding to the top rim of the object of interest, the arc of the shadow on the inner surface of the object of interest, and the roof of the object of interest. The template generation module 103 uses the object's height and width, desired floating-roof height, the two satellite angles and the two sun angles. Using that information and the trigonometric equations shown in FIG. 8, the template generation module 103 creates 2D projections of where the circles lie. The template generation module 103 may also crop each

idealized image to position the center of the object of interest in the center of each idealized image.

[0057] Once the circle positions are generated, the template generation module 103 synthesizes the idealized images illustrated in FIG. 9 for different filled volumes of the object by performing a convolution on the circle corresponding to the top rim, the circle corresponding to the arc of the shadow, and the circle corresponding to the roof. The template generation module 103 performs the convolution by performing unions and intersections between the three circles to generate the “eyeball” shapes (dark and shadow regions) for the idealized object images shown in FIG. 9. The template generation module 103 may rescale each idealized image of the object of interest by setting pixels corresponding to shadows and inner surfaces of the object of interest to negative values such as -1, setting pixels corresponding to the roof of the object of interest to positive values such as +1, and setting all other pixels to 0.

[0058] Unions and intersections between the three circles may be performed by the template generation module 103, e.g., using morphological image processing. Morphological image processing refers to non-linear operations related to the shape or morphology of features in an image. Morphological image processing operations rely only on the relative ordering of pixel values, not on their numerical values, and therefore are suited to the rescaled idealized images. The intersection of two images A and B , written $A \cap B$, is the binary image which is 1 at all pixels p which are 1 in both A and B . The union of A and B , written $A \cup B$ is the binary image which is 1 at all pixels p which are 1 in A or 1 in B (or in both).

[0059] The template matching module 205 shown in FIG. 2 matches the received image of the object of interest to each idealized image synthesized by the template generation module 103 to determine a filled volume of the object of interest. The matching may be performed by

performing a dot product between pixels of the received image and pixels of the idealized image. Because the received image and the idealized image are rescaled such that its pixel values range from -1 to +1, i.e., dark pixels (shadows, inner wall, etc.) are negative and light pixels (roof, etc.) are positive, performing a dot product between the received image and the idealized image results in a large positive number if the received image and the idealized image look similar. This is because positive pixels in the received image line up with positive pixels in the idealized image, and negative pixels in the received image line up with negative pixels in the idealized image.

[0060] Performing the dot product by the template matching module 205 shown in FIG. 2 is an algebraic operation that takes pixels of the received image and pixels of the idealized image and returns a single number. Algebraically, the dot product is the sum of the products of the corresponding pixel values of the pixels of the received image and pixels of the idealized image. For example, the dot product of the two images $A = [a_1, a_2, \dots, a_n]$ and $B = [b_1, b_2, \dots, b_n]$, where A is the received image and B is the idealized image template may be determined as $A \cdot B = \sum_i a_i b_i = a_1 b_1 + a_2 b_2 + \dots + a_n b_n$. Further details of the convolutions performed by the image analysis module 204 and the template matching module 205 to avoid false positive matches are illustrated and described below with reference to FIG. 10A.

[0061] To allow for inaccuracies in geo-referenced imagery, and the fact that an object may not be precisely in the location expected, the template matching module 205 performs a sweep over the received image to account for a number of possible locations of the object. The template matching module 205 performs the sweep by using 2D convolution between the received image and each template. Once the template matching module 205 has found a template match for the received image of the object of interest, it determines the filled volume of

the object of interest as the filled volume corresponding to the matching idealized image template.

[0062] The container analysis module 107 may analyze an object of interest pattern including one or more of the time of capture of the received image, the count of one or more objects of interest in the received image, and the determined filled volume of each of one or more objects of interest in the received image. The container analysis module 107 may send information to the user device 120 if the analyzed object of interest pattern exceeds a threshold. For example, the container analysis module 107 may send information to the user device 120 if the count of the objects of interest in the received image exceeds a threshold or the determined filled volume of a threshold number of objects exceeds a threshold.

[0063] The container pattern store 206 shown in FIG. 2 may store patterns received from the container analysis module 107. The container pattern store 206 may be organized as a database or table stored on one or more of removable or non-removable memory cards, tape cassettes, zip cassettes, and computer hard drives. In one embodiment, the container pattern store 206 stores multiple data fields, each describing one or more attributes of an object. In one example, the container pattern store 206 stores, for a single object, the time of capture of images, geographical region coordinates the height of the object, and/or the width of the object.

EXAMPLE MACHINE LEARNING TRAINING SETS

[0064] FIG. 3A illustrates an example positive training set 300 for the remote container analysis system 101, in accordance with an embodiment. As part of the training of the machine learning model 106, the machine learning training engine 203 forms a training set of features and training labels, e.g., container 305, by identifying a positive training set of features that have been determined to have the property in question (presence of containers), and, in some

embodiments, forms a negative training set of features that lack the property in question, as described below in detail with reference to FIG. 3B. For example, each training set may include labeled pixel clusters corresponding to containers, e.g., containers 303. To collect a training set, polygons may be marked around known tank farms around the world and downloaded Landsat 8 imagery may be intersected with these polygons. Randomly sampled imagery may also be collected for a set of negative examples (i.e., images that contain no oil tank farms). Once trained, the machine learning model 106 may be run on all imagery in a region of interest (e.g., the United States). The final output of the remote container analysis system 101 is a set of areas of interest (geometry polygons) where the machine learning model 106 returned a high output score.

[0065] The positive training set 300 shown in FIG. 3A contains features that have been determined to have the presence of containers. The positive training set 300 may include labeled pixel clusters corresponding to container 305, containers 303, and container 304. The positive training set 300 also contains labels for background regions water 301 and land 302. The example training set 300 may correspond to a port where the land 302 meets water 301. In the positive training set 300, container 305 is in the area labeled water, while containers 303 and container 304 are in the area labeled land.

[0066] FIG. 3B illustrates an example negative training set 350 for the remote container analysis system 101, in accordance with an example embodiment. The negative training set 350 shown in FIG. 3B contains features that have been determined to lack the presence of containers. The negative training set 350 includes a false positive cluster of pixels 354 that is located partly in the water 351 and partly on the land 352. The negative training set 350 also includes a false positive cluster of pixels 353 related to two intersecting clusters of pixels. Since two containers

cannot intersect each other, these two intersecting clusters of pixels are a false positive and labeled as such (353).

EXAMPLE CLOUD COVER DETECTION TRAINING SETS

[0067] FIG. 3C illustrates an example positive training set 370 for cloud cover detection, in accordance with an example embodiment. Cloud cover within aerial images may obscure or distort containers or other objects of interest in the images and remote sensing image interpretation. Therefore, traditional image processing methods are configured to remove aerial images having even a small amount of cloud cover before image recognition is performed. However, even though a particular image might have an amount of cloud cover, the image can still be useful as long as the cloud cover is less than a cloud detection threshold and specific areas of interest are not obscured. The remote container analysis system 101 therefore determines the presence and absence of clouds in aerial images as well as the amount of the cloud cover to determine whether to use the results of the computer vision methods described herein for container fill measurement.

[0068] As part of the training of the machine learning model 106, the machine learning training engine 203 forms a training set of cloud features (e.g., 375), water features (e.g., 371), land features (e.g., 372), and training labels by identifying a positive training set of features that have been determined to have the property in question (presence of clouds), and, in some embodiments, forms a negative training set of features that lack the property in question, as described below in detail with reference to FIG. 3D. For example, each training set may include labeled pixel clusters corresponding to clouds, e.g., 373, 374, and 375. To collect a training set, polygons may be marked around known clouds and downloaded Landsat 8 imagery may be intersected with these polygons. Randomly sampled imagery may also be collected for a set of

negative examples (e.g., features related to snow fields that contain no clouds). Once trained, the machine learning model 106 may be run on all imagery in a region of interest (e.g., a region in Europe). An output of the remote container analysis system 101 is a set of areas of interest (geometry polygons) where the machine learning model 106 returned a high output score. This is used to determine an amount of cloud cover present in an image of an object of interest.

[0069] The positive training set 370 shown in FIG. 3C contains features that have been determined to have the presence of clouds (e.g., 375). The positive training set 300 may include labeled pixel clusters corresponding to clouds 373, 374, and 375. The positive training set 370 also contains labels for background features corresponding to water 371 and land 372. The example training set 370 may correspond to a port where the land 372 meets water 371. In the positive training set 370, clouds 375 are in the area labeled water, while clouds 373 and 374 are in the area labeled land.

[0070] FIG. 3D illustrates an example negative training set 380 for cloud cover detection, in accordance with an example embodiment. The negative training set 380 shown in FIG. 3D contains features (false positives) that have been determined to lack the presence of clouds. The negative training set 380 includes a false positive cluster of pixels 384 and a false positive cluster of pixels 383. The cluster of pixels 384 represents a false positive because it captures a snow field on the edge between water 381 and land 382, and not a cloud. The cluster of pixels 383 represents a false positive because it captures a snow field upon a mountain top, and not a cloud.

[0071] In some example embodiments, the training sets 370 and 380 may be created by manually labeling pixel clusters that represent high scores and pixel clusters that represent low scores. In other embodiments, the machine learning training engine 203 may extract training sets from stored images obtained from the image store 102. For example, if a stored image contains

cloud features located on land, e.g., 373, the machine learning training engine 203 may use the pixel features as a positive training set.

EXAMPLE MACHINE LEARNING TRAINING PROCESS

[0072] Referring now to FIG. 4A, it illustrates an example training process for the machine learning training engine 203 for the machine learning model 106 in the remote container analysis system 101. The process may use the image analysis module 204, the feature extraction module 104, and the machine learning model 106. FIG. 4A and the other figures use like reference numerals to identify like elements. A letter after a reference numeral, such as “410a,” indicates that the text refers specifically to the element having that particular reference numeral plus letter suffix. A reference numeral in the text without a following letter, such as “410,” refers to any or all of the elements in the figures bearing that reference numeral, e.g., “410” in the text refer to reference numerals “410a” and/or “410b” in the figures.

[0073] The image analysis module 204 may perform edge analysis in the training images 401 to identify pixels in the training images 401 corresponding to the objects of interest. The optional feature extraction module 104 shown in FIG. 4A may extract features 410 from the training images 401. The features 410 corresponding to the training images 401 are used for training the machine learning model 106 based on training labels 402. In one example embodiment, a feature 410a may represent aggregate values based on pixel attributes of pixels in the images 401. Extracting the feature 410a from the training images 401 may include performing pixel clustering to identify clusters of adjacent pixels in the training images 401. The adjacent pixels in the training images 401 match each other based on a pixel attribute. An example feature 410b may represent whether two clusters of adjacent pixels in an image intersect each other; this feature teaches the machine learning model 106 that the two pixel clusters may

not represent a container because containers cannot intersect.

[0074] An example feature 410c may represent whether a cluster of pixels is located partly on land and partly on water; this feature teaches the machine learning model 106 that the pixel cluster may not represent a container because containers cannot be located partly on land 302 and partly on water 301. A feature 410d may represent an association between pixel locations and a pixel attribute. For example, the feature 410d may represent the brightness value of a pixel relative to pixels located on its right in an image; this feature teaches the machine learning model 106 that the pixel may be part of a pixel cluster representing a container because the pixel is brighter than surrounding pixels. A feature 410e may represent the brightness of a pixel relative to the average brightness of pixels located on the same row in an image; this feature teaches the machine learning model 106 that the pixel may be part of an image blob representing a container because the pixel is brighter (e.g., greater illumination) than surrounding pixels.

[0075] The machine learning training engine 203 may train the machine learning model 106 shown in FIG. 4A using the feature vector 410 and training labels 402. In one embodiment, the machine learning model 106 is thereby configured to determine a score for each pixel location in an image, the score indicative of a likelihood that the pixel location corresponds to a container. In another embodiment, the machine learning model 106 is configured to determine a score for pixel clusters, the score indicative of a likelihood that the pixel clusters correspond to containers. In alternative embodiments, the machine learning model 106 is configured to generate an output including pixel clusters and a score indicative of a likelihood that the pixel clusters correspond to containers. In an embodiment, the machine learning model 106 is configured to generate an output including one or more pixel locations corresponding to a pixel cluster and a score indicative of a likelihood that the pixel locations correspond to a pixel cluster. In an

embodiment, the machine learning model 106 is configured to generate an output including a number of pixels in each identified pixel cluster. In an embodiment, the machine learning model 106 is configured to generate an output including an association between the identified pixel clusters and containers.

[0076] The machine learning model training engine 203 may apply machine learning techniques to train the machine learning model 106 that when applied to features outputs indications of whether the features have an associated property or properties, e.g., that when applied to features of received images outputs estimates of whether there are containers present, such as probabilities that the features have a particular Boolean property, or an estimated value of a scalar property. The machine learning training engine 203 may apply dimensionality reduction (e.g., via linear discriminant analysis (LDA), principle component analysis (PCA), or the like) to reduce the amount of data in the feature vector 410 to a smaller, more representative set of data.

[0077] The machine learning training engine 203 may use supervised machine learning to train the machine learning model 106 shown in FIG. 4A, with the feature vectors 410 of the positive training set 300 and the negative training set 350 serving as the inputs. In other embodiments, different machine learning techniques, such as linear support vector machine (linear SVM), boosting for other algorithms (e.g., AdaBoost), logistic regression, naïve Bayes, memory-based learning, random forests, bagged trees, decision trees, boosted trees, boosted stumps, neural networks, CNNs, etc., may be used. The machine learning model 106, when applied to the feature vector 410 extracted from a set of received images, outputs an indication of whether a pixel cluster has the property in question, such as a Boolean yes/no estimate, or a scalar value representing a probability.

[0078] In some example embodiments, a validation set is formed of additional features, other than those in the training sets, which have already been determined to have or to lack the property in question. The machine learning training engine 203 applies the trained machine learning model 106 shown in FIG. 4A to the features of the validation set to quantify the accuracy of the machine learning model 106. Common metrics applied in accuracy measurement include: Precision = $TP / (TP + FP)$ and Recall = $TP / (TP + FN)$, where Precision is how many results (e.g., containers or other objects) the machine learning model 106 correctly predicted (TP or true positives) out of the total it predicted (TP + FP or false positives), and Recall is how many results the machine learning model 106 correctly predicted (TP) out of the total number of features that did have the property in question (TP + FN or false negatives). The F score ($F\text{-score} = 2 \times PR / (P + R)$) unifies Precision and Recall into a single measure. In one embodiment, the machine learning training engine 203 iteratively re-trains the machine learning model 106 until the occurrence of a stopping condition, such as the accuracy measurement indication that the machine learning model 106 is sufficiently accurate, or a number of training rounds having taken place.

[0079] In alternative embodiments, the machine learning model 106 may be a CNN that learns useful representations (features) such as which pixel clusters correspond to containers directly from training sets without explicit feature extraction. For example, the machine learning model 106 may be an end-to-end recognition system (a non-linear map) that takes raw pixels from the training images 401 directly to internal labels. The machine learning model 106 shown in FIG. 4A (in the form of a CNN) may generate an output directly from the training images 401, without the need for feature extraction, edge analysis or pixel cluster identification.

CLOUD COVER DETECTION

[0080] The machine learning training engine 203 may similarly train the machine learning model 106 to perform cloud cover detection. For example, as described above with respect to FIG. 3C, training sets (e.g., 370) may be created including labeled images 401 of a geographical area. The labeled images 401 of the geographical area may include distinct amounts of cloud cover, e.g., 5 %, 10 %, 35 %, 75 %, etc. The machine learning model 106 is configured, based on the training sets 370 and 380, to detect an amount of cloud cover present in an unlabeled image by differentiating features in the unlabeled image corresponding to the cloud cover present in the unlabeled image (e.g., 375) from background features (e.g., 371) present in the unlabeled image.

[0081] To perform cloud cover detection using the trained machine learning model 106 on an unlabeled image of an object of interest, a feature vector 410 may be optionally extracted from the unlabeled image of the object of interest. The feature vector 410 may include features representing attributes of pixels in the image of the object of interest. For example, features analogous to the attributes of pixels described above with respect to FIG. 4A may be used. Detecting the amount of cloud cover present in the image of the object of interest may include transmitting the feature vector 410 to the trained machine learning model 106.

[0082] In one embodiment, cloud regions during the training phase may include small, medium, and large size clouds. The backgrounds may include mountains, buildings, roads, agriculture, and rivers. The machine learning model 106 may be used to classify clouds (e.g., 375) in different situations with different cloud types, cloud sizes, densities, and different underlying landscape environments. Higher resolution images may be used for texture information, clearer geometry shape attributes, and a more distinct spatial distribution of features. For example, features may change little between adjacent pixels of clouds and water,

but the textures of buildings and mountains may be very different. Thus, uniform cloud texture may be used to distinguish clouds from buildings and mountains.

[0083] In one embodiment, geometry shape attributes may be used to differentiate features corresponding to the cloud cover (e.g., 373) from background features (e.g., 372) and features corresponding to the object of interest (e.g., 305). For example, cloud features are more natural and irregular, while road features are relatively narrow and linear, and building features have relatively, regular shape outlines. Therefore, features related to both spectral characteristics and texture may be used.

[0084] In one embodiment, “feature fusion” may be used. Feature fusion involves normalizing a variety of features (e.g., 373 and 371) and then merging the normalized features into a single feature set. This allows selection of features that are related to categories and to exclude redundant features. Multiple sets of feature vectors 410 may be created to classify the target features. Multiple sets of features may also be combined by complex vectors and used to classify the target features.

[0085] The machine learning model 106 uses the image of the object of interest to detect an amount of cloud cover present in the image of the object of interest by differentiating features corresponding to the cloud cover (e.g., 373) from background features (e.g., 372) and features corresponding to the object of interest (e.g., 305).

[0086] The benefits and advantages of this method are that a larger number of images may now be used for remote container volume detection. Even if some geographical regions have persistent cloud cover at certain times in the year, accurate volume detection can be performed for containers. The amount of data transmitted is reduced since there are less wasted images. The flexible use of cloud detection thresholds enables detection of objects and the filled volume

of objects even in the presence of cloud cover.

EXAMPLE PRECISION VS. RECALL CURVE

[0087] FIG. 4B illustrates an example Precision vs. Recall curve for training a machine learning model to perform cloud cover detection, in accordance with an embodiment. The machine learning training engine 203 uses Precision (also called positive predictive value) to denote a fraction of relevant cloud features among all features. The Recall (also known as sensitivity) is used to denote the fraction of relevant cloud features that have been retrieved over the total amount of relevant cloud features.

[0088] For example, the machine learning model 106 may identify seven clouds in an aerial image having eleven clouds and some land features. Of the seven clouds identified, four actually are clouds (true positives), while the rest are buildings (false positives). The machine learning model 106's Precision may be determined as $4/7$ while its Recall may be determined as $4/11$. When the machine learning model 106 returns 29 clouds, only 19 of which were true clouds while failing to return 39 additional clouds, its Precision may be determined as $19/29$, while its Recall is $19/58$.

EXAMPLE PROCESS FOR IDENTIFYING REMOTE OBJECTS

[0089] FIG. 5 illustrates an example process for the remote container analysis system 101 for identifying remote objects, in accordance with an embodiment. In some example embodiments, the process may have different and/or additional steps than those described in conjunction with FIG. 5. Steps of the process may be performed in different orders than the order described in conjunction with FIG. 5. Some steps may be executed in parallel. Alternatively, some of the steps may be executed in parallel and some steps executed sequentially. Alternatively, some steps may execute in a pipelined fashion such that execution of a step is started before the

execution of a previous step.

[0090] The remote container analysis system 101 receives 500 a first image of a geographical area, where the first image has a first resolution. The first image is of a large geographic region. The large geographic region may be predefined, for example, based on area. This area may be, for example, an entire country, e.g., the United States, or a smaller portion such as a state/province or city, e.g., Texas or Houston. To make scanning over a large area practical and efficient, lower resolution imagery is used for the first image. An example of such imagery is 15 m / pixel Landsat imagery (in the panchromatic band). The feature extraction module 104 extracts 504 a first feature vector from the first image. The first feature vector may include aggregate values based on pixel attributes of pixels in the first image, as described above with reference to FIG. 2. The remote container analysis system 101 transmits 508 the first feature vector to the machine learning model 106 to identify an area of interest containing an object of interest in the first image. Identifying the area of interest containing the object of interest in the first image includes, for each pixel in the first image, determining a likelihood that the pixel corresponds to the object of interest, as described above with reference to FIGs. 2 and 4. The machine learning model 106 is trained to have high Recall at the price of lower Precision (e.g., a higher false positive rate).

[0091] The remote container analysis system 101 receives 512 a second image of the geographical area. The second image has a second resolution higher than the first resolution. The processing of the low resolution first image is followed by a cleanup phase on the second image. To filter out the false positives, a second pass is performed over all areas of interest returned by the first pass. This time higher resolution imagery is used where individual containers can be seen more clearly (e.g., using 50 cm per pixel imagery). The feature extraction

module 104 extracts 516 a second feature vector from the second image. The second feature vector includes aggregate values based on pixel attributes of pixels in the area of interest, as described above with reference to FIG. 2.

[0092] The remote container analysis system 101 transmits 520 the second feature vector to the machine learning model 106 to determine a likelihood that the area of interest contains the object of interest. Determining the likelihood that the area of interest contains the object of interest includes, for each pixel in the area of interest, determining a likelihood that the pixel corresponds to the object of interest, as described above with reference to FIGs. 2 and 4. If the likelihood is below a threshold, the remote container analysis system 101 trains 524 the machine learning model to filter out features corresponding to the area of interest in images having the first resolution. To improve the accuracy of the machine learning model 106 a procedure may be performed that is referred to as “bootstrapping” or “hard negative mining.” The clean-up is restricted to a reasonably small set of high scoring areas of interest. Areas of interest receiving a high score but containing no objects are added back into the negative training sets, and the machine learning model 106 is trained again. This procedure ensures that the training set contains “difficult” negative examples and can improve precision and reduce the number of false positives.

[0093] In one example embodiment, training 524 the machine learning model 106 to filter out the features corresponding to the area of interest includes extracting a feature vector corresponding to the area of interest from the first image. The remote container analysis system 101 creates a training set including the feature vector and a label corresponding to a lack of objects of interest in the first image. The remote container analysis system 101 configures the machine learning model 106, based on the training set, to identify the lack of objects of interest

in the first image. In another example embodiment, training 524 the machine learning model 106 to filter out the features corresponding to the area of interest includes extracting a feature vector corresponding to the area of interest from the first image and configuring the machine learning model 106, based on the extracted feature vector, to report a lack of objects of interest in the first image.

[0094] If the likelihood that the area of interest in the second image contains the object of interest exceeds a threshold, the remote container analysis system 101 transmits 528 a visual representation of the object of interest to a user device, as described in FIG. 2.

EXAMPLE PROCESS FOR DETERMINING THE FILLED VOLUME OF REMOTE OBJECTS

[0095] FIG. 6A illustrates an example process for the remote container analysis system 101 for determining the filled volume of remote objects, in accordance with an embodiment. In some embodiments, the process may have different and/or additional steps than those described in conjunction with FIG. 6A. Steps of the process may be performed in different orders than the order described in conjunction with FIG. 6A. Some steps may be executed in parallel.

Alternatively, some of the steps may be executed in parallel and some steps executed sequentially. Alternatively, some steps may execute in a pipelined fashion such that execution of a step is started before the execution of a previous step.

[0096] The remote container analysis system 101 processes satellite imagery to search for intersections of imagery and known floating-roof container locations. The container image is received 600 and cropped such that the center of the container is in the center of the image. Using the cropped image, the task is to determine the filled volume of the container (i.e., determine how far down the roof is). In an example embodiment, the system is configured so that the containers are assumed to be light colored, and the inner walls of each container are dark

colored. The remote container analysis system 101 extracts 604 a parameter vector from the image. The parameter vector may include parameters describing the latitude and longitude of the container, an image timestamp, the satellite elevation and azimuth angles, the sun elevation and azimuth angles, and the tank height and width (or diameter).

[0097] In an example embodiment, the remote container analysis system 101 may perform 608 image analysis on the image to determine the height and width of the object of interest (container), as described above with reference to FIG. 2. The remote container analysis system 101 generates 612 idealized images of the object of interest using the extracted parameter vector, the determined height, and the determined width of the object of interest, as described above with reference to FIG. 2 and illustrated below with reference to FIG. 7. Each idealized image corresponds to a distinct filled volume of the object of interest, as illustrated and described below with reference to FIG. 9.

[0098] The remote container analysis system 101 matches 616 the received image of the object of interest to each idealized image to determine the filled volume of the object of interest. The matching includes performing a dot product between pixels of the received image and pixels of the idealized image, as described above with reference to FIG. 2 and further illustrated below with reference to FIG. 9. The remote container analysis system 101 transmits 620 information corresponding to the determined filled volume of the object of interest to a user device 120, as described above with reference to FIG. 2.

EXAMPLE PROCESS FOR CLOUD COVER DETECTION AND FILLED VOLUME DETERMINATION

[0099] FIG. 6B illustrates an example process for cloud cover detection and determining the filled volume of remote objects, in accordance with an embodiment. In some embodiments, the process may have different and/or additional steps than those described in conjunction with FIG.

6B. Steps of the process may be performed in different orders than the order described in conjunction with FIG. 6B. Some steps may be executed in parallel. Alternatively, some of the steps may be executed in parallel and some steps executed sequentially. Alternatively, some steps may execute in a pipelined fashion such that execution of a step is started before the execution of a previous step.

[00100] The remote container analysis system 101 receives 630 an image of an object of interest (e.g., container) from the aerial imaging device 110, which may be a satellite, drone, etc. The image of the object of interest may be characterized by its spatial resolution (pixel size of an image representing the size of the surface area being measured on the ground), spectral resolution (wavelength interval size and number of intervals measured), temporal resolution (amount of time passed between image collection periods), and radiometric resolution (ability of the device 110 to record different levels of brightness).

[00101] The remote container analysis system 101 selects 634 a cloud detection threshold. The cloud detection threshold is a fraction or percentage value of cloud cover in an image. The cloud detection threshold may be selected by comparing container detection or filled volume detection results using cloudy vs. non-cloudy images. The cloud detection threshold may thus be selected as the amount of cloud cover at which the remote container analysis system 101 can accurately identify containers and determine the filled volume of the identified containers. For example, the cloud detection threshold may be 15 %. A margin may also be added to the cloud detection threshold in order to mitigate the effects of false negatives during cloud cover detection. False negatives may lead the remote container analysis system 101 to believe that there is less cloud cover (e.g., 15 %) than actually present (e.g., 16 %). In this event, the volume detection may be impacted. Therefore, a margin (e.g., 1 %) added to the cloud detection

threshold improves the accuracy of container detection and volume detection. The cloud detection threshold and/or margin may also be altered according to the weather, geographic region, and type of object of interest under analysis.

[00102] The remote container analysis system 101 detects 638 by the machine learning model 106, using the image of the object of interest, an amount of cloud cover present in the image of the object of interest. The machine learning model 106 is configured to differentiate features corresponding to the cloud cover from background features and features corresponding to the object of interest, as described and illustrated in detail above with respect to FIGs. 3C, 3D, 4A, and 4B.

[00103] Responsive to the amount of cloud cover present in the image of the object of interest being less than the cloud detection threshold, the remote container analysis system 101 extracts 642, from the image of the object of interest, parameters describing at least an azimuth angle of the aerial imaging device. Other parameters that may be extracted are described in detail with respect to FIGs. 2, 6A, and FIG. 8.

[00104] The remote container analysis system 101 generates 646, from the parameters, idealized images of the object of interest. Each idealized image correspond to a distinct filled volume of the object of interest. The idealized images are generated as described and illustrated with respect to FIGs. 2, 6A, 7, 8, and 9. For example, the generating of each idealized image may include generating a first circle corresponding to a top rim of the object, generating a second circle corresponding to an arc of a shadow on an inner surface of the object, generating a third circle corresponding to a roof of the object, and synthesizing the idealized image by performing a convolution on the three circles.

[00105] The remote container analysis system 101 matches 650 the image of the object of

interest to each idealized image to determine a filled volume of the object of interest, as described and illustrated with respect to FIGs. 2, 6A, and 10A-10D. For example, the matching of the image of the object of interest to each idealized image may include superimposing the idealized image on different locations of the image of the object of interest and performing a dot product between pixels of the image of the object of interest and pixels of the idealized image at each of the different locations to determine a maximum value of the dot product at each of the different locations.

[00106] The remote container analysis system 101 transmits 654 information corresponding to the determined filled volume of the object of interest to a user device 120.

EXAMPLE SYNTHESIS OF IDEALIZED IMAGE

[00107] FIG. 7 illustrates an example synthesis 700 of an idealized image, in accordance with an embodiment. The template generation module 103 assumes that the container, including its roof, is white or light-colored. It also assumes that shadows and the inner container surfaces are black. The template generation module 103 generates idealized image templates from the positions of circles: a top circle 704 (the top rim of the object), a bottom circle 720 (the bottom of the object, where it contacts the ground), a roof height circle 708 (that represents the roof of the object), and an internal shadow circle (generated from the arc of the internal shadow on the inner surface 712 of the object). In embodiments, only the top circle 704, roof height circle 708, and internal shadow circle 712 may be used. To generate the idealized image templates, the template generation module 103 uses the following information: object height and width, desired roof height, the two satellite angles, and the two sun angles. Based on the information above and the trigonometric equations shown in FIG. 8, the template generation module 103 creates 2D projections of where the circles lie.

[00108] The template generation module 103 generates an idealized image for a given filled volume of the object by generating the circle 704 corresponding to the top rim of the object of interest using the parameter vector as shown in FIG. 8. The template generation module 103 generates a circle corresponding to an arc of a shadow on an inner surface 712 of the object of interest using the parameter vector. The template generation module 103 generates the circle 708 corresponding to the roof of the object of interest using the parameter vector. The template generation module 103 uses the shadow 716 on the roof to create a template corresponding to the desired roof height as shown in FIG. 8. The template generation module 103 may synthesize the idealized image by performing a convolution on the circle 704, the circle 720, the circle corresponding to the arc of the internal shadow 712, and the circle 708.

[00109] Once the circle positions are known, the template generation module 103 computes unions and intersections to generate the “eyeball” shape (dark and shadow regions) template shown in FIG. 7, as described above with reference to FIG. 2. In the final templates, internal shadow pixels and interior wall pixels are set to -1, roof pixels are set to +1, and all other pixels are set to 0. This is done so that dark pixels (e.g., shadows and inner surfaces) are negative and light pixels (e.g., roof) are positive. A dot product performed between the input image and an idealized image will then result in a large positive number if the template and image are similar because positive pixels in the image will line up with positive pixels in the idealized image, and negative pixels in the image will line up with negative pixels in the idealized image.

EXAMPLE CIRCLE PROJECTION EQUATIONS

[00110] FIG. 8 illustrates a set of example circle projection equations, in accordance with an embodiment. The template generation module 103 generates idealized image templates from the positions of the circles illustrated and described above with reference to FIG. 7 based on the

extracted parameters and the trigonometric equations shown in FIG. 8.

[00111] In one embodiment, the template generation module 103 may create projections, based on the trigonometric equations shown in FIG. 8 to map the parameter vector onto the circles. The template generation module 103 may project the shadows cast by the top rim onto the roof and the inner surface onto a plane as follows. The projection of a point is its shadow on the plane. The shadow of a point on the plane is the point itself. For example, the projection from a point onto a plane may be performed as follows. If C is a point, called the center of projection, then the projection of a point P different from C onto a plane that does not contain C is the intersection of the line CP with the plane. The points P , such that the line CP is parallel to the plane do not have any image by the projection. However, they are regarded as projecting to a point at infinity of the plane. The projection of the point C itself is not defined. In another example, the projection may be performed parallel to a direction D , onto a plane as follows. The image of a point P is the intersection with the plane of the line parallel to D passing through P .

[00112] In alternative embodiments, the template generation module 103 may define a projective space $P(V)$ of dimension n over a field K as the set of the lines in a K -vector space of dimension $n + 1$. If a basis of V has been fixed, a point of V may be represented by a point $(x_0, \dots, x_n)$ of K^{n+1} . A point of $P(V)$, being a line in V , may thus be represented by the coordinates of any nonzero point of this line. Given two projective spaces $P(V)$ and $P(W)$ of the same dimension, the template generation module 103 may generate an homography as a mapping from $P(V)$ to $P(W)$, which is induced by an isomorphism of vector spaces $f: V \rightarrow W$. Such an isomorphism induces a bijection from $P(V)$ to $P(W)$, because of the linearity of f . Two such isomorphisms, f and g , may define the same homography if and only if there is a nonzero element a of K such that $g = af$.

EXAMPLE IDEALIZED IMAGES

[00113] FIG. 9 illustrates a set of example idealized images 900, in accordance with an embodiment. The idealized images 900 are generated by the template generation module 103 by varying the filled volume percentage of the container of interest from 0% filled (image 904) to 100% filled (image 924). In image 908, the filled volume percentage of the container is 20%. In image 912, the filled volume percentage of the container is 40%. The shadow 936 in image 912 cast by the top rim of the container on the roof 932 and the inner surface of the container is smaller than the shadow in image 908.

[00114] In image 916, the filled volume percentage of the container is 60%. The shadow in image 916 cast by the top rim of the container on the roof and the inner surface of the container is smaller than the shadow 936 in image 912. In image 920, the filled volume percentage of the container is 80%. The shadow in image 920 cast by the top rim of the container on the roof and the inner surface of the container is smaller than the shadow in image 916. In image 924, the filled volume percentage of the container is 100%. There is no shadow in image 924.

[00115] For a given set of inputs, the remote container analysis system 101 determines which idealized template among the images 900 matches the received image best, and then returns the corresponding filled volume percentage. In one example embodiment, the template matching module 205 determines the filled volume of the container based on the received image, the satellite and sun angles, and the container dimensions as follows. The template matching module 205 sets the variable “best_score” to a large negative number. The template matching module 205 sets the variable “best_fill_percentage” to -1. The template matching module 205 performs the following steps for different filled volume percentages from 0% to 100%. The template matching module 205 determines the score from matching the received image to each template. If the score is higher than “best_score,” the template matching module 205 sets the value of

“best_score” to the score and the value of “best_fill_percentage” to the filled volume percentage. At the end of the process, the template matching module 205 returns the value of “best_fill_percentage.”

EXAMPLE IMAGE GRADIENTS AND OUTLINES OF REMOTE OBJECTS

[00116] Referring now to FIG. 10A, it illustrates an example received image 1000 of a container, in accordance with an embodiment. The container has a roof 1008 having a shadow 1004. When the roof 1008 of the floating-roof container is all the way up (a full container), the matching idealized image is a white circle (where all the pixels have value 1) surrounded by gray pixels 1012 (pixels with a value of 0), illustrated above as image 924 in FIG. 9. This template will match any white region with the same score. To avoid false positives, gradient information from the received image 1000 may be incorporated by the template matching module 205.

[00117] FIG. 10B illustrates an example image gradient 1020 for the received image 1000 of FIG. 10A, in accordance with an embodiment. The image analysis module 204 may perform edge analysis, as described above with reference to FIG. 2, on the received image 1000 to obtain the image gradient 1020 of the received image 1000. The image gradient 1020 represents the directional change in the intensity or color in the image 1000. The image analysis module 204 may derive the image gradient 1020 as a single value at each pixel. At each image point, the gradient denotes the largest possible intensity increase. The edge 1024 in FIG. 10B represents the change in the intensity or color in the image 1000 from the background 1012 to the shadow 1004 in FIG. 10A. The edge 1028 in FIG. 10B represents the change in the intensity or color in the image 1000 from the shadow 1004 to the roof 1008 in FIG. 10A. The edge 1032 in FIG. 10B represents the change in the intensity or color in the image 1000 from the roof 1008 to the background 1012 in FIG. 10A.

[00118] FIG. 10C illustrates an example outline 1040 of a top rim 1044 of the object of interest (container) in an idealized image template, in accordance with an embodiment. The image analysis module 204 may perform edge analysis on the idealized image to obtain the outline 1040 of the top rim 1044 of the container in the idealized image. For example, the image analysis module 204 may perform edge thinning to remove the unwanted spurious points on the edge 1044 in the outline 1040. The image analysis module 204 may perform edge thinning after the idealized image has been filtered for noise (e.g., using median, Gaussian filters etc.), the edge operator has been applied (as described above with reference to FIG. 2) to detect the edge 1044, and after the edge 1044 has been smoothed using an appropriate threshold value. This removes all the unwanted points and results in one-pixel-thick edge elements in an embodiment.

[00119] The template matching module 205 may perform a dot product between pixels of the image gradient 1020 and pixels of the outline 1040 of the top rim 1044 in order to determine the filled volume of the container in the received image 1000. The benefits and advantages of this process are that sharp and thin edges lead to greater efficiency in template matching. Using Hough transforms to detect arcs (of shadows) and circles (e.g., the top rim) results in greater accuracy.

[00120] FIG. 10D illustrates an example outline 1060 of the shadow on the inner surface of a container in an idealized image template, in accordance with an embodiment. The image analysis module 204 may perform edge analysis on the idealized image to obtain the outline 1060 of the shadow on the inner surface of the container. In FIG. 10D, edge 1064 represents the change in the intensity or color in the idealized image from the background to the shadow on the inner surface. Edge 1068 represents the change in the intensity or color in the idealized image from the shadow on the inner surface to the roof. The template matching module 205 may

perform a dot product between pixels of the image gradient 1020 and pixels of the outline of the shadow 1060 in order to determine the filled volume of the container in the received image 1000.

[00121] In some example embodiments, three convolutions may be performed and added up to form the response map. The first convolution is between the received image 1000 and the idealized image template, e.g., image 912 in FIG. 9. The second convolution is between the image gradient 1020 and the outline of the top rim 1040. The third convolution is between the image gradient 1020 and the outline of the shadow 1060. The three resulting response maps may be summed, and the location with the maximal response within a specified radius of the center of the image may be determined as the final template match score. The above procedure may be generalized to any situation where the geometry of the object of interest is known and characterized by a small number of parameters, most of which are known. The unknown parameters can then be determined by sweeping over possible values, generating templates, and matching them to the input image.

EXAMPLE MACHINE ARCHITECTURE

[00122] FIG. 11 is a block diagram illustrating components of an example machine able to read instructions described as processes herein from a machine-readable medium and execute them in at least one processor (or controller). Specifically, FIG. 11 shows a diagrammatic representation of a machine in the example form of a computer system 1100. The computer system 1100 can be used to execute instructions 1124 (e.g., program code or software) for causing the machine to perform any one or more of the methodologies (or processes) described herein. In alternative embodiments, the machine operates as a standalone device or a connected (e.g., networked) device that connects to other machines. In a networked deployment, the machine may operate in the capacity of a server machine or a client machine in a server-client

network environment, or as a peer machine in a peer-to-peer (or distributed) network environment. It is noted the instructions correspond to the functionality of components and/or processes described herein, for example, with respect to FIGS. 1, 2, and 4-6. The instructions also may correspond to the processes associated with driving to the results shown in FIGs. 3A-3B, 7, 9, and 10A-10D.

[00123] The machine may be a server computer, a client computer, a personal computer (PC), a tablet PC, a set-top box (STB), a smartphone, an internet of things (IoT) appliance, a network router, switch or bridge, or any machine capable of executing instructions 1124 (sequential or otherwise) that specify actions to be taken by that machine. Further, while only a single machine is illustrated, the term “machine” shall also be taken to include any collection of machines that individually or jointly execute instructions 1124 to perform any one or more of the methodologies discussed herein.

[00124] The example computer system 1100 includes one or more processing units (generally processor 1102). The processor 1102 is, for example, a central processing unit (CPU), a graphics processing unit (GPU), a digital signal processor (DSP), a controller, a state machine, one or more application specific integrated circuits (ASICs), one or more radio-frequency integrated circuits (RFICs), or any combination of these. The computer system 1100 also includes a main memory 1104. The computer system may include a storage unit 1116. The processor 1102, memory 1104 and the storage unit 1116 communicate via a bus 1108.

[00125] In addition, the computer system 1100 can include a static memory 1106, a display driver 1110 (e.g., to drive a plasma display panel (PDP), a liquid crystal display (LCD), or a projector). The computer system 1100 may also include alphanumeric input device 1112 (e.g., a keyboard), a cursor control device 1114 (e.g., a mouse, a trackball, a joystick, a motion sensor, or

other pointing instrument), a signal generation device 1118 (e.g., a speaker), and a network interface device 1120, which also are configured to communicate via the bus 1108.

[00126] The storage unit 1116 includes a machine-readable medium 1122 on which is stored instructions 1124 (e.g., software) embodying any one or more of the methodologies or functions described herein. The instructions 1124 may also reside, completely or at least partially, within the main memory 1104 or within the processor 1102 (e.g., within a processor's cache memory) during execution thereof by the computer system 1100, the main memory 1104 and the processor 1102 also constituting machine-readable media. The instructions 1124 may be transmitted or received over a network 1126 via the network interface device 1120.

[00127] While machine-readable medium 1122 is shown in an example embodiment to be a single medium, the term "machine-readable medium" should be taken to include a single medium or multiple media (e.g., a centralized or distributed database, or associated caches and servers) able to store the instructions 1124. The term "machine-readable medium" shall also be taken to include any medium that is capable of storing instructions 1124 for execution by the machine and that cause the machine to perform any one or more of the methodologies disclosed herein. The term "machine-readable medium" includes, but not be limited to, data repositories in the form of solid-state memories, optical media, and magnetic media. It is noted that in some example embodiments, the core components of the computer system may disregard components except for the processor 1102, memory 1104, and bus 1108 and may in other embodiments also include the storage unit 1116 and/or the network interface device 1120.

ADDITIONAL CONSIDERATIONS

[00128] The remote container analysis system as disclosed provides benefits and advantages that include the transformation of clusters of pixels into a digital representation of remote

containers, and for each remote container, the digital representation of the roof, inner surfaces, and the filled volume of the remote container. Other advantages of the system include faster processing of the aerial images, less power consumption, lower latency in remote container detection, less data transmitted over the network, etc.

[00129] Throughout this specification, plural instances may implement components, operations, or structures described as a single instance. Although individual operations of one or more methods are illustrated and described as separate operations, one or more of the individual operations may be performed concurrently, and nothing requires that the operations be performed in the order illustrated. Structures and functionality presented as separate components in example configurations may be implemented as a combined structure or component. Similarly, structures and functionality presented as a single component may be implemented as separate components. These and other variations, modifications, additions, and improvements fall within the scope of the subject matter herein.

[00130] Certain embodiments are described herein as including logic or a number of components, modules, or mechanisms, for example, as illustrated and described with FIGs. 1, 2, 4, 5, 6, and 11. Modules may constitute either software modules (e.g., code embodied on a machine-readable medium) or hardware modules. A hardware module is tangible unit capable of performing certain operations and may be configured or arranged in a certain manner. In example embodiments, one or more computer systems (e.g., a standalone, client or server computer system) or one or more hardware modules of a computer system (e.g., a processor or a group of processors) may be configured by software (e.g., an application or application portion) as a hardware module that operates to perform certain operations as described herein.

[00131] In various embodiments, a hardware module may be implemented mechanically or

electronically. For example, a hardware module may include dedicated circuitry or logic that is permanently configured (e.g., as a special-purpose processor, such as a field programmable gate array (FPGA) or an application-specific integrated circuit (ASIC)) to perform certain operations. A hardware module may also include programmable logic or circuitry (e.g., as encompassed within a general-purpose processor or other programmable processor) that is temporarily configured by software to perform certain operations. It will be appreciated that the decision to implement a hardware module mechanically, in dedicated and permanently configured circuitry, or in temporarily configured circuitry (e.g., configured by software) may be driven by cost and time considerations.

[00132] The various operations of example methods described herein may be performed, at least partially, by one or more processors, e.g., processor 1102, that are temporarily configured (e.g., by software) or permanently configured to perform the relevant operations. Whether temporarily or permanently configured, such processors may constitute processor-implemented modules that operate to perform one or more operations or functions. The modules referred to herein may, in some example embodiments, include processor-implemented modules.

[00133] The one or more processors may also operate to support performance of the relevant operations in a “cloud computing” environment or as a “software as a service” (SaaS). For example, at least some of the operations may be performed by a group of computers (as examples of machines including processors), these operations being accessible via a network (e.g., the Internet) and via one or more appropriate interfaces (e.g., application program interfaces (APIs).)

[00134] The performance of certain of the operations may be distributed among the one or more processors, not only residing within a single machine, but deployed across a number of

machines. In some example embodiments, the one or more processors or processor-implemented modules may be located in a single geographic location (e.g., within a home environment, an office environment, or a server farm). In other example embodiments, the one or more processors or processor-implemented modules may be distributed across a number of geographic locations.

[00135] Some portions of this specification are presented in terms of algorithms or symbolic representations of operations on data stored as bits or binary digital signals within a machine memory (e.g., a computer memory). These algorithms or symbolic representations are examples of techniques used by those of ordinary skill in the data processing arts to convey the substance of their work to others skilled in the art. As used herein, an “algorithm” is a self-consistent sequence of operations or similar processing leading to a desired result. In this context, algorithms and operations involve physical manipulation of physical quantities. Typically, but not necessarily, such quantities may take the form of electrical, magnetic, or optical signals capable of being stored, accessed, transferred, combined, compared, or otherwise manipulated by a machine. It is convenient at times, principally for reasons of common usage, to refer to such signals using words such as “data,” “content,” “bits,” “values,” “elements,” “symbols,” “characters,” “terms,” “numbers,” “numerals,” or the like. These words, however, are merely convenient labels and are to be associated with appropriate physical quantities.

[00136] Unless specifically stated otherwise, discussions herein using words such as “processing,” “computing,” “calculating,” “determining,” “presenting,” “displaying,” or the like may refer to actions or processes of a machine (e.g., a computer) that manipulates or transforms data represented as physical (e.g., electronic, magnetic, or optical) quantities within one or more memories (e.g., volatile memory, non-volatile memory, or a combination thereof), registers, or

other machine components that receive, store, transmit, or display information.

[00137] As used herein any reference to “one embodiment” or “an embodiment” means that a particular element, feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment. The appearances of the phrase “in one embodiment” in various places in the specification are not necessarily all referring to the same embodiment.

[00138] Some embodiments may be described using the expression “coupled” and “connected” along with their derivatives. For example, some embodiments may be described using the term “coupled” to indicate that two or more elements are in direct physical or electrical contact. The term “coupled,” however, may also mean that two or more elements are not in direct contact with each other, but yet still co-operate or interact with each other. The embodiments are not limited in this context.

[00139] As used herein, the terms “comprises,” “comprising,” “includes,” “including,” “has,” “having” or any other variation thereof, are intended to cover a non-exclusive inclusion. For example, a process, method, article, or apparatus that includes a list of elements is not necessarily limited to only those elements but may include other elements not expressly listed or inherent to such process, method, article, or apparatus. Further, unless expressly stated to the contrary, “or” refers to an inclusive or and not to an exclusive or. For example, a condition A or B is satisfied by any one of the following: A is true (or present) and B is false (or not present), A is false (or not present) and B is true (or present), and both A and B are true (or present).

[00140] In addition, use of the “a” or “an” are employed to describe elements and components of the embodiments herein. This is done merely for convenience and to give a general sense of the claimed invention. This description should be read to include one or at least one and the

singular also includes the plural unless it is obvious that it is meant otherwise.

[00141] Upon reading this disclosure, those of skill in the art will appreciate still additional alternative structural and functional designs for a system and a process for identifying and determining the filled volume of remote containers from low resolution imagery through the disclosed principles herein. Thus, while particular embodiments and applications have been illustrated and described, it is to be understood that the disclosed embodiments are not limited to the precise construction and components disclosed herein. Various modifications, changes and variations, which will be apparent to those skilled in the art, may be made in the arrangement, operation and details of the method and apparatus disclosed herein without departing from the spirit and scope defined in the appended claims.

CLAIMS

1. A method for processing images from an aerial imaging device, the method comprising:
detecting by a machine learning model, using an image of an object of interest, an amount of cloud cover present in the image of the object of interest, the machine learning model configured to differentiate features corresponding to the cloud cover from background features and features corresponding to the object of interest;
responsive to the amount of cloud cover present in the image of the object of interest being less than a cloud detection threshold, extracting, from the image of the object of interest, parameters describing at least an azimuth angle of the aerial imaging device;
generating, from the parameters, idealized images of the object of interest, wherein each idealized image corresponds to a distinct filled volume of the object of interest;
matching the image of the object of interest to each idealized image to determine a filled volume of the object of interest; and
transmitting information corresponding to the determined filled volume of the object of interest to a user device.
2. The method of claim 1, further comprising:
creating training sets comprising labeled images of a geographical area, the labeled images of the geographical area comprising distinct amounts of cloud cover; and
configuring the machine learning model, based on the training sets, to detect an amount of cloud cover present in an unlabeled image by differentiating features in the

unlabeled image corresponding to the cloud cover present in the unlabeled image from background features present in the unlabeled image.

3. The method of claim 1, further comprising extracting a feature vector from the image of the object of interest, the feature vector comprising features representing attributes of pixels in the image of the object of interest, wherein the detecting of the amount of cloud cover present in the image of the object of interest comprises transmitting the feature vector to the machine learning model.
4. A method for processing images from an aerial imaging device, the method comprising:
detecting by a machine learning model, using an image of an object of interest, an amount of cloud cover present in the image of the object of interest;
responsive to the amount of cloud cover present in the image of the object of interest being less than a threshold, extracting, from the image of the object of interest, parameters describing at least an azimuth angle of the aerial imaging device;
determining, from the parameters, a filled volume of the object of interest; and
transmitting information corresponding to the determined filled volume of the object of interest to a user device.
5. The method of claim 4, further comprising:
creating training sets comprising labeled images of a geographical area, the labeled images of the geographical area comprising distinct amounts of cloud cover; and
configuring the machine learning model, based on the training sets, to detect an amount of cloud cover present in an unlabeled image by differentiating features in the

unlabeled image corresponding to the cloud cover present in the unlabeled image from background features present in the unlabeled image.

6. The method of claim 4, further comprising extracting a feature vector from the image of the object of interest, the feature vector comprising features representing attributes of pixels in the image of the object of interest, wherein the detecting of the amount of cloud cover present in the image of the object of interest comprises transmitting the feature vector to the machine learning model.
7. The method of claim 4, wherein the parameters further describe one or more of:
a time of capture of the received image;
an elevation angle of the sun; and
an azimuth angle of the sun.
8. The method of claim 4, wherein the determining of the filled volume of the object of interest comprises:
generating, from the parameters, idealized images of the object of interest, wherein each
idealized image corresponds to a distinct filled volume of the object of interest;
and
matching the image of the object of interest to each idealized image to determine the
filled volume of the object of interest.
9. The method of claim 8, further comprising rescaling each idealized image by:

setting pixels corresponding to shadows and inner surfaces of the object of interest in the idealized image to negative values; and
setting pixels corresponding to a roof of the object of interest in the idealized image to positive values.

10. The method of claim 8, wherein the generating of each idealized image comprises:
generating a first circle corresponding to a top rim of the object of interest using the parameters;
generating a second circle corresponding to an arc of a shadow on an inner surface of the object of interest using the parameters;
generating a third circle corresponding to a roof of the object of interest using the parameters; and
synthesizing the idealized image by performing a convolution on the three circles.
11. The method of claim 10, wherein the performing of the convolution comprises
performing unions and intersections between the first circle corresponding to the top rim, the second circle corresponding to the arc of the shadow, and the third circle corresponding to the roof.
12. The method of claim 8, wherein the matching of the image of the object of interest to each idealized image comprises:
superimposing the idealized image on different locations of the image of the object of interest;

performing a dot product between pixels of the image of the object of interest and pixels of the idealized image at each of the different locations to determine a maximum value of the dot product at each of the different locations.

13. The method of claim 8, wherein the matching of image of the object of interest to each idealized image comprises:

performing edge analysis on the idealized image to obtain an outline of a top rim of the object of interest in the idealized image; and

performing a dot product between pixels of an image gradient of the image of the object of interest and pixels of the outline of the top rim.

14. The method of claim 8, wherein the matching of the image of the object of interest to each idealized image comprises:

performing edge analysis on the idealized image to obtain an outline of a shadow on an inner surface of the object of interest in the idealized image; and

performing a dot product between pixels of an image gradient of the image of the object of interest and pixels of the outline of the shadow.

15. The method of claim 8, wherein the matching of the image of the object of interest to each idealized image comprises:

performing a dot product between pixels of the image of the object of interest and pixels of the idealized image to obtain a first response map;

performing a dot product between pixels of an image gradient of the image of the object of interest and pixels of an outline of a top rim of the object of interest in the idealized image to obtain a second response map;

performing a dot product between pixels of the image gradient and pixels of an outline of a shadow on an inner surface of the object of interest in the idealized image to obtain a third response map; and

summing up the three response maps.

16. A non-transitory computer-readable storage medium comprising instructions executable by a processor, the instructions when executed by the processor cause the processor to:

detect by a machine learning model, using an image of an object of interest, an amount of cloud cover present in the image of the object of interest;

responsive to the amount of cloud cover present in the image of the object of interest

being less than a cloud detection threshold, extract, from the image of the object of interest, parameters describing at least an azimuth angle of the aerial imaging device;

determine, from the parameters, a filled volume of the object of interest; and

transmit information corresponding to the determined filled volume of the object of interest to a user device.

17. The non-transitory computer-readable storage medium of claim 16, the instructions further causing the processor to:

create training sets comprising labeled images of a geographical area, the labeled images of the geographical area comprising distinct amounts of cloud cover; and

configure the machine learning model, based on the training sets, to detect an amount of cloud cover present in an unlabeled image by differentiating features in the unlabeled image corresponding to the cloud cover present in the unlabeled image from background features present in the unlabeled image.

18. The non-transitory computer-readable storage medium of claim 16, the instructions further causing the processor to extract a feature vector from the image of object of interest, the feature vector comprising features representing attributes of pixels in the image of the object of interest, wherein the detecting of the amount of cloud cover present in the image of the object of interest comprises transmitting the feature vector to the machine learning model.

19. The non-transitory computer-readable storage medium of claim 16, wherein the determining of the filled volume of the object of interest comprises:

generating, from the parameters, idealized images of the object of the interest, wherein each idealized image corresponds to a distinct filled volume of the object of interest; and

matching the image of the object of interest to each idealized image to determine the filled volume of the object of interest.

20. The method of claim 19, wherein the generating of each idealized image comprises:

generating a first circle corresponding to a top rim of the object of interest using the parameters;

generating a second circle corresponding to an arc of a shadow on an inner surface of the object of interest using the parameters;

generating a third circle corresponding to a roof of the object of interest using the parameters; and
synthesizing the idealized image by performing a convolution on the three circles.