Mapping Coconut trees using AI and Remote Sensing technologies

Anusri Sanyadanam[1] and Thomas Cecere[2]

[1]University of Hyderabad, Hyderabad, Telangana, India
[2]National Land Imaging Program, 516 National Center Reston, VA 20192, United States

March 19, 2025

1 Introduction

The Pacific ocean is the largest ocean in the world and contains roughly 30,000 islands and islets. Most of the Pacific atoll communities depend on limited resources for sustaining their traditional agro-forestry practices, maintaining biodiversity, supporting local economies through subsistence agriculture and marine resources, and ensuring food security [1]. The most widespread vegetation on these atolls consists of coconut plantations, with a majority of coconut palm trees, followed by pandanus, breadfruit and other kinds of indigenous trees and plants [2].

Coconut trees (*Cocos nucifera*) are an important member of the palm tree family and play a vital role in tropical ecosystems, agriculture, and livelihoods [3]. They are primarily grown as a cash or commodity crop for the production of oil which is exported around the world. Natural disasters like tsunamis and hurricanes can lead to severe catastrophic loss of the plantations. Monitoring coconut trees is important for a detailed survey of planting areas, predicting the yield from crops and the long-term impact on the ecology and biodiversity of a place, especially as their dominance has led to severe deforestation, depletion of groundwater resources along with adverse effects on seabird and coral reef populations [4].

Estimating and regularly tracking each of these trees by hand is not a viable solution as it would require a lot of time and is labor-intensive. Hence, satellites and transformative tools such as remote sensing techniques can be used for mapping coconut trees in atoll regions, as satellites have large-scale coverage, which includes accessibility to remote areas. They also capture a wide range of spectral data which allows Remote Sensing technologies to provide fine-grained, real-time data for temporal analysis of vegetation change and this can also be integrated with several environmental metrics to analyze the health of vegetation or for detecting pest infestations. Coconut trees can grow upto 30 metres in height, which makes them suitable for detection using high-resolution imagery captured by satellites and UAVs (unmanned aerial vehicles like drones).

Artificial Intelligence techniques such as Machine Learning models enable the analysis of large datasets of data to identify hidden patterns and obtain crucial insights for effective decision-making, especially in the areas of biodiversity conservation and agricultural monitoring.

This paper explores training machine learning models on different platforms for object detection to identify individual coconut trees from satellite imagery of various Pacific atoll regions. Additionally, tools such as ArcGIS Pro and QGIS, along with platforms like Google Earth Engine and Google Earth Pro, were used to analyze vegetation cover and identify areas with coconut trees.

2 Keywords

A list of the important keywords and terminology used in this paper:

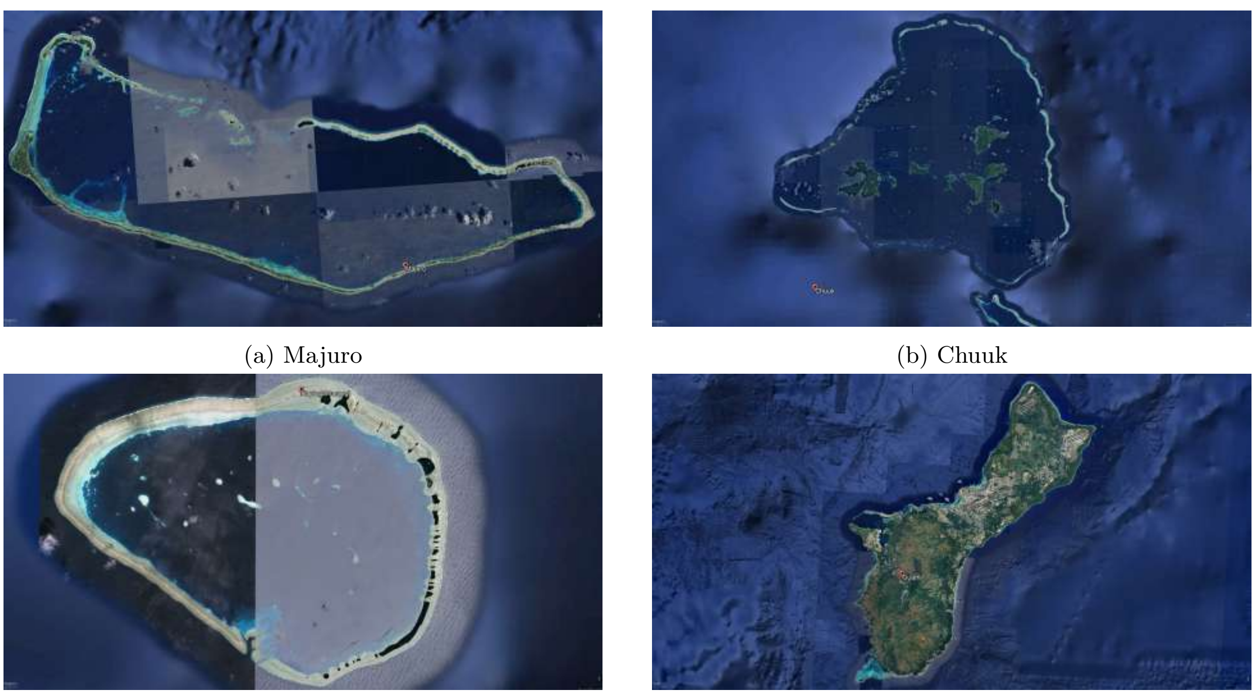
- *Neural Network:* A machine learning model inspired by the human brain, consisting of layers of interconnected nodes (neurons) that learn patterns from data.
- *CNN (Convolutional Neural Network):* A specialized neural network for image processing that uses convolutional layers to detect patterns like edges, textures, and objects.

- *mAP (Mean Average Precision):* A key metric in object detection that measures how well a model detects and classifies objects, averaged across multiple intersection-over-union (IoU) thresholds.
- *Precision:* The ratio of correctly predicted positive cases to all predicted positives, measuring how many detected objects are actually correct.
- *Recall:* The ratio of correctly predicted positive cases to all actual positives, indicating how well the model captures all relevant objects.
- *Loss:* A numerical value representing the difference between the model's predictions and the actual labels, guiding optimization during training.

3 Study Areas and Data Sources

Pacific islands and atolls were considered for mapping coconut trees. The following regions were chosen for understanding the spread of coconut plantations and groves along with their impact on agriculture, biodiversity and the local community:

- Majuro, Majuro Atoll, Republic of Marshall Islands (RMI)
- Chuuk, Federated States of Micronesia (FSM)
- Kapingamarangi, FSM and
- Guam, US territory in Micronesia



(a) Majuro

(b) Chuuk

(c) Kapingamarangi

(d) Guam

Figure 1: Regions of Interest in this work, images obtained from Google Earth Pro

In this work, the following data sources were used to obtain imagery:

- Multiresolution Seamless Image Database (MrSID) format Natural Color imagery of Majuro from USDA.

- High-resolution UAV-based (drone) UAS Imagery made with Agisoft software as a GeoTIFF file for Majuro.
- Google Earth Pro (Desktop Application)

The source imagery was stored in the JPEG or TIFF format and QGIS (Quantum-GIS) application was used to visualize the imagery in detail.

In addition to satellite images, Google Earth Engine (GEE) and Sentinel-2 Explorer App were used to obtain pre-existing land cover, agriculture and vegetation maps to visualize the extent of tree cover and vegetation categories for the RoIs. The GEE platform was also used to create custom composites and vegetation maps based on the band values of satellite imagery, where the images were clipped to consider the RoIs and a cloud cover removal step was performed as part of the pre-processing stage.

4 Methodology and Implementation Details

In this work, three approaches were considered for the detection of coconut tree instances from the imagery.

- Deep Learning using ArcGIS Pro
- Training a YOLOv5 Deep Learning model
- Fine-tuning the custom YOLOv5 model for other regions

4.1 Deep Learning using ArcGIS Pro

This implementation was performed based on the tutorial from ArcGIS Pro [5] to assess the health of Coconut Trees based on the Visible Atmospherically Resistant Index (VARI) and classify them into different categories based on the condition. While the original implementation considered Kolovai from the island of Tongatapu in Oceania with the source drone imagery from the OpenAerialMap data repository, in this implementation, the MrSID data was considered to detect trees for Majuro which has RGB bands for the image.

Access to quality-resolution spectral and spatial imagery is crucial, as the pixel size must be sufficient to distinguish palm tree canopies from other trees and objects. To generate high-quality training samples, the Label Objects in Deep Learning tool in ArcGIS Pro was used to manually label individual coconut trees. This process ensured that the model learned to recognize the shape, size, and spectral characteristics of coconut trees while also capturing the percentage of pixels corresponding to each marked instance.

Since deep learning models improve with a larger number of training samples, eight sub-regions from Majuro were selected for labeling. These areas provided a sufficiently large dataset, which was then digitized into image chips, ensuring a comprehensive record of the distinct features that differentiate coconut trees from other types of vegetation. The image chips were formatted to the JPEG type with tile size of 448 x 448 and stride of 128 x 128, which refers to the step size used when sliding a window over the original image to extract these chips. Since each image chip is 448 pixels wide and 448 pixels tall, the window moves 128 pixels at a time both horizontally and vertically when extracting the next chip. PASCAL VOC, a widely used data annotation format for object detection and segmentation tasks was considered as the Meta Data Format. It defines how bounding boxes, class labels, and metadata are stored for training deep learning models and stores the relevant details in the form of XML files.

For model training, ArcGIS Pro has a 'Train Deep Learning Model' geoprocessing tool which uses the image chips to understand which combination of pixels represent coconut trees. Single-Shot Detector (SSD) [6], a well-known deep learning-based object detection model that performs both object localization and classification in a single forward pass of the neural network was chosen for its speed and efficiency, with a ResNet-34 backbone to enhance feature extraction. SSD models in 2 have two main components:

- *Backbone:* A backbone is a pretrained CNN that extracts key features from an image before passing them to the object detection phase.
- *SSD Head:* One or more convolutional layers added to the backbone.

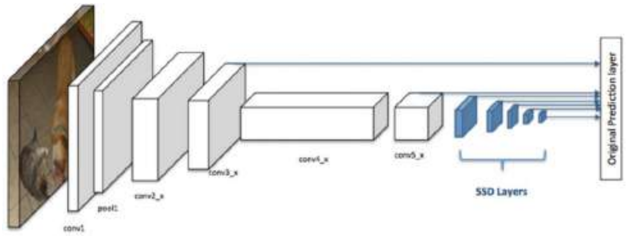


Figure 2: Architecture of SSD Model

Since palm trees are well defined, relatively uniform in shape, and spread across an image, SSD can efficiently detect them without the need for a complex, slow object detection approaches.

Before training, the following key parameters were configured in ArcGIS Pro: a batch size of 8, a 10% validation percentage and a chip size of 448. Validation loss was monitored as the primary metric, with a mechanism to freeze the model if further epochs did not lead to improvements. This ensured optimal performance and prevented overfitting. The model was trained for 75 epochs, which took approximately 15 hours and 18 minutes on a CPU-only system due to lack of access to GPUs. During each epoch, the entire training dataset is passed forward and backward through the neural network architecture.

After the training process completes, ArcGIS Pro has a tool called 'Detect Objects Using Deep Learning' where certain parameters like the padding, threshold, and, nms_overlap are provided. Padding refers to adding extra pixels (usually zeros) around the borders of an image. As the convolution operation reduces the size of image by shrinking the data represented in the image, padding helps to preserve the spatial dimensions, while also allowing edge details (like boundary of canopies) to be learned. A threshold is a confidence score (between 0 and 1) that determines whether a detected object should be kept or discarded. Each bounding box prediction has a confidence score and if the score is equal or above the chosen threshold (to ensure high accuracy of output predictions), the provided output is considered as a valid detection. When multiple bounding boxes detect the same object, NMS (Non-Maximum Suppression) removes redundant boxes, keeping only the one with the highest confidence and setting the value determines how much overlap is allowed before two detections are considered duplicates. This entire process generates a layer with predictions of coconut tree instances for a new, unseen region (which can be selected in advance).

To assess the health of the vegetation, the VARI index is considered which is an indirect measure of leaf area index (LAI) and vegetation fraction (VF) using only reflectance values from the visible wavelength (source image has all bands in the visible spectrum only).

$$VARI = \frac{R_g - R_r}{R_g + R_r - R(R_g - R_b)}$$

where,

$$R_r, R_g, and, R_b$$

are reflectance values for the red, green, and blue bands. Trees were classified into health categories based on their average VARI values:

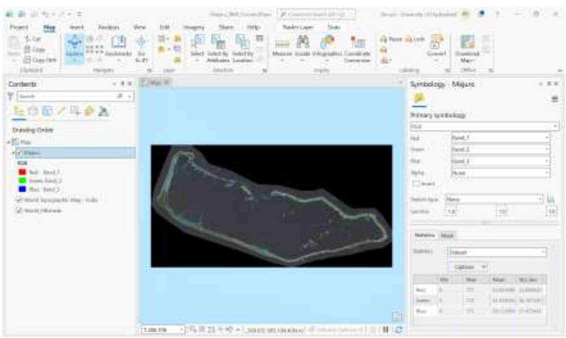
1. Healthy
2. Moderate
3. Indication of Declining Health and
4. Needs Inspection

To calculate VARI, the Band Arithmetic raster function was configured with band index values as 1 2 3 (for RGB) and stretch type was set to Standard Deviation which rendered a VARI layer for the image. Using the detected tree layer, the Feature to Point tool is applied to generate a point feature class representing each tree's centroid. Since palm trees have an average crown radius of 2.5 to 3 meters, a buffer is created around each point to approximate the tree canopy. This results in a polygon feature class representing individual tree crowns, along with information on the location, condition and model confidence for each palm tree.

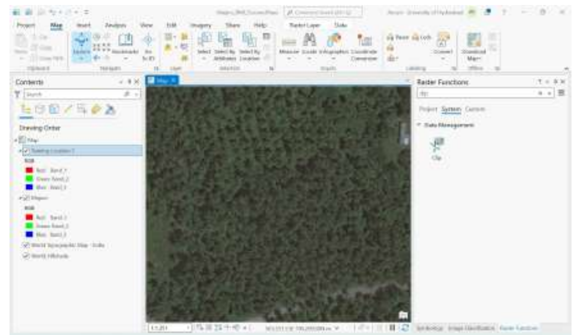
To assess tree health, the Vegetation Adjusted Red Index (VARI) is calculated from the palm tree raster layer using the Band Arithmetic function. Since VARI is a raster dataset, we need to extract the average VARI value within each tree's polygon. The Zonal Statistics as Table tool computes the mean VARI for each buffered tree polygon, producing a table of vegetation health values.

Next, the Join Field tool is used to associate the computed mean VARI values with the tree polygons. Finally, graduated color symbology is applied based on the VARI categories, visually classifying tree health into four levels. The final map displays the location and health status of each detected palm tree.

A visual representation of the workflow is provided in Figures 3 and 4:

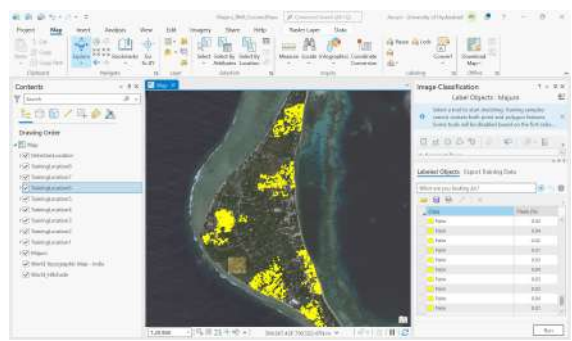


(a) Majuro Imagery in ArcGIS Pro



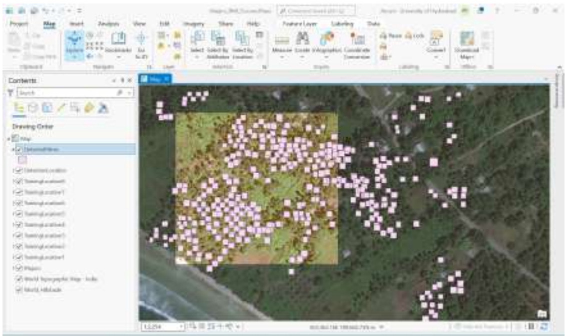
(b) Training Location image with Coconut trees

(c) Labeling coconut trees in each training location

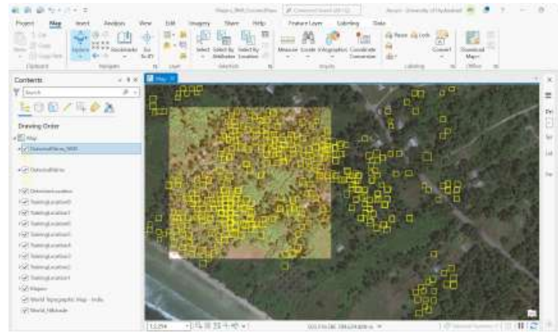


(d) Representation of some labeled training location subsets on the actual island

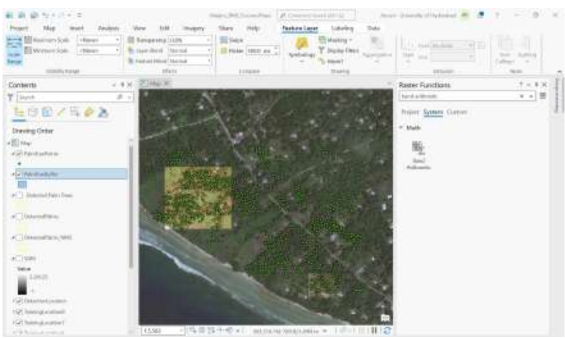
Figure 3: Workflow of the Deep Learning Implementation in ArcGIS Pro



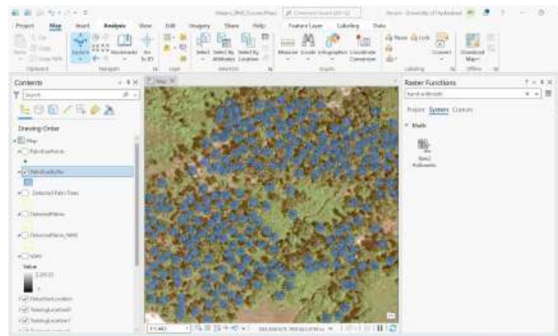
(a) Layer with Detected Palm trees



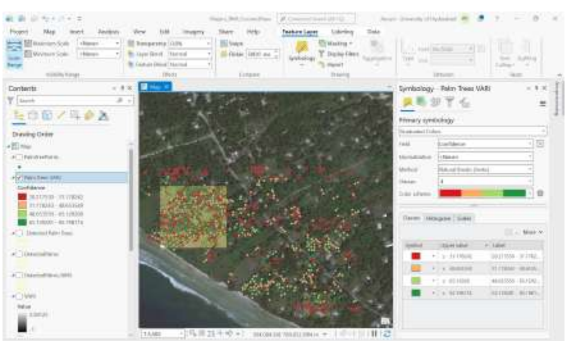
(b) Output layer after NMS to remove overlapping detections



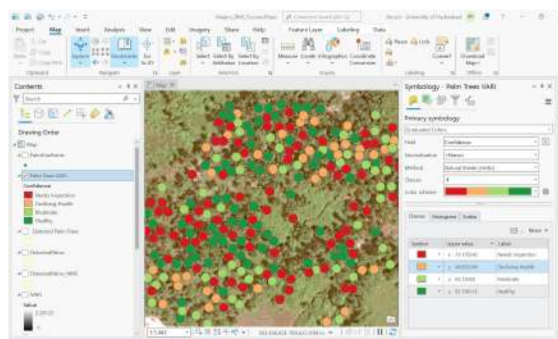
(c) Palm Tree layer with detected tree points



(d) Close up view of the coconut tree points



(e) Final Output Layer with Health Classification of detected trees



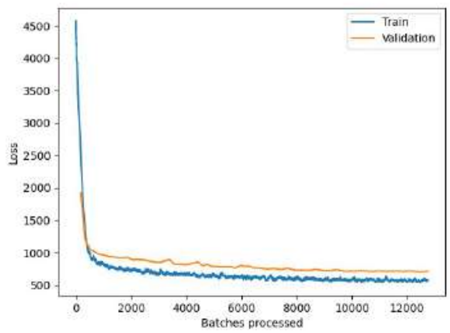
(f) Close up view of the predicted health condition values

Figure 4: Workflow of the Deep Learning Implementation in ArcGIS Pro

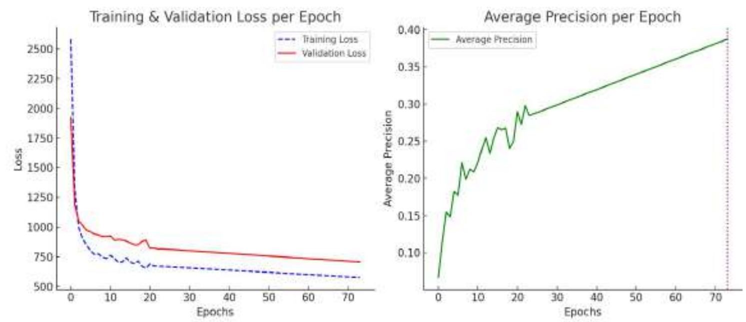
4.1.1 Results

The graph in 5a shows the training and validation loss of the SSD Model with the ResNet-34 backbone for detecting coconut trees. The X-axis (”Batches processed”) represents training progress, while the Y-axis (”Loss”) quantifies model error, combining localization loss (bounding box accuracy) and confidence loss (classification). These values have no physical unit but reflect the error magnitude. Initially high (≈ 4000 to 4500), the loss decreases as the model learns, stabilizing around 500-1000. A lower loss indicates better performance, while the gap between training and validation loss helps assess overfitting or underfitting.

Epoch 73 has the best results as it has both the lowest validation loss and the highest average precision, with validation loss of 705.71356 and average precision of 0.3873685026065561. Training loss (blue, dashed) and validation loss (red) in 5b show a decreasing trend. This suggests the model is learning well, though validation loss fluctuates. The average precision per epoch shows an increasing trend, meaning the model's performance improves over time. The best epoch (73) is marked with a dotted purple line.



(a) Loss Graph from ArcGIS Pro training



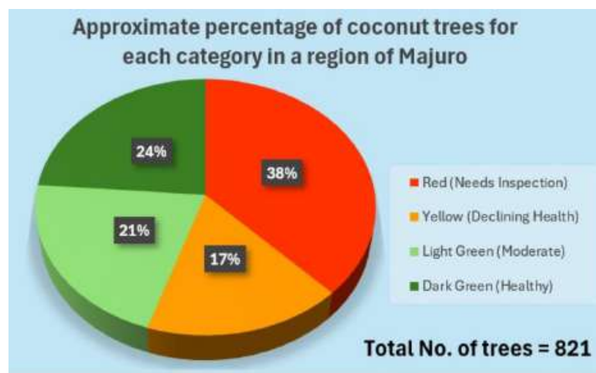
(b) Graphs generated based on the metrics from training

Figure 5: Graphs to represent various metrics from the SSD Model training

The average upper bound values for each health category are depicted in 6a. A particular sub-region of the island was considered as the detection zone to test the performance of the model. In this scenario, the model was able to detect a total of 821 coconut trees in the form of shapefile points shown in 7a and the distribution according to health classes is provided in 6b. Most of the trees were detected correctly, and the model's performance can be further improved with a larger training dataset, which would require more instances of manually labeling the trees along with access to systems with GPU for faster training.

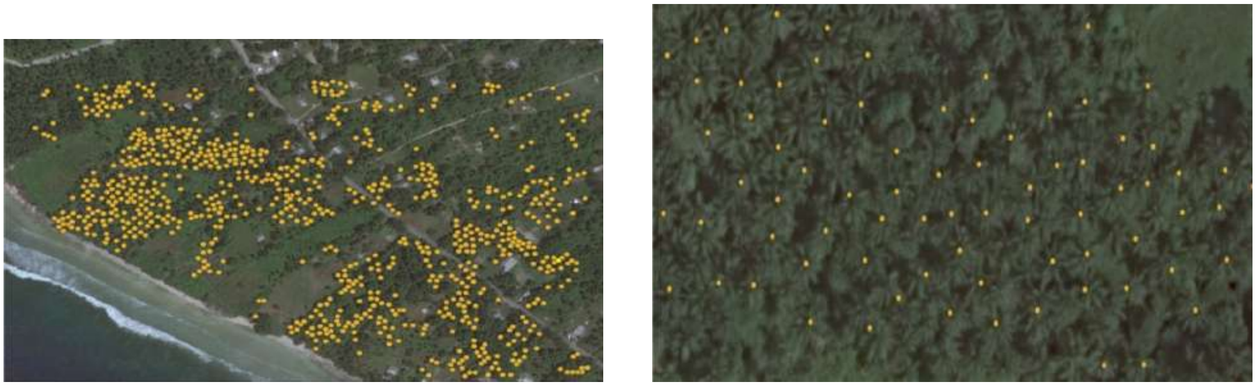
Symbol	Upper value	Label
	≤ 31.178242	Needs Inspection
	≤ 48.653549	Indication of Declining Health
	≤ 65.12928	Moderate
	≤ 92.198116	Healthy

(a) Table with classes of various health conditions based on average VARI values



(b) A pie-chart with approximate percentage distribution of trees

Figure 6: Images of health categories classification



(a) Final output of detected trees (821 shapefile points)

(b) Closer view of the shapefile points on the trees

Figure 7: Images of detected coconut tree points on the original imagery mosaic

This workflow demonstrates the benefit of automating the process, which allows to save a lot of time and resources. A major advantage of this work is the ability to make data-driven decisions. The health metrics derived using the VARI index give valuable insights, especially for imagery where Near-Infra Red (NIR) reflectance band data is not available, which is crucial for popular vegetation indices like NDVI (Normalized Difference Vegetation Index). Farmers and land managers can use this data to decide where to allocate resources or when certain areas need attention. A satellite imagery time-series can be used to verify whether there is an increase / decrease in the number of trees over the years and perform detailed surveys for a specific detection region with the help of output shapefiles that are generated.

4.2 Deep Learning using YOLOv5

This implementation involves two parts: The first aspect is to generate shapefiles of possible coconut tree positions from the model training for Majuro and the second aspect involves generating Landcover and vegetation maps for the areas of interest.

4.2.1 Coconut Tree points from YOLOv5

The YOLO (You Only Look Once) model was developed by [7] and it runs much more faster than SSDs. Input images are partitioned as grids with cells to predict bounding boxes in each grid-cell center along with objectness scores. Classifications are performed with an Inception-model like CNN called DarkNet to learn features which are unique to objects of different classes. YOLO also uses NMS to remove boxes with lower objectness scores and it finally chooses the best remaining box based on the intersection over union (IoU) similarity metric.

YOLOv5 is a powerful, single-stage object detection algorithm and deep learning model developed by Ultralytics, which predicts bounding boxes for the objects of interest along with the class probabilities and other metrics in one pass with a single neural network architecture to process the entire image. YOLOv5 is a robust choice for applications prioritizing speed and efficiency, especially in resource-limited scenarios compared to other versions of the model.

The YOLOv5 model architecture consists of the following parts:

- *Backbone:* This is the main body of the network which is designed using CSPDarkNet53, a Convolutional Neural Network.
- *Neck:* This connects the backbone of the model with the head and uses SPPF (Spatial Pyramid Pooling - Fast - it allows the network to capture information at different spatial scales) and PANet (Path Aggregation Network - it improves the flow of information between different network levels) structures.
- *Head:* This generates the final output.

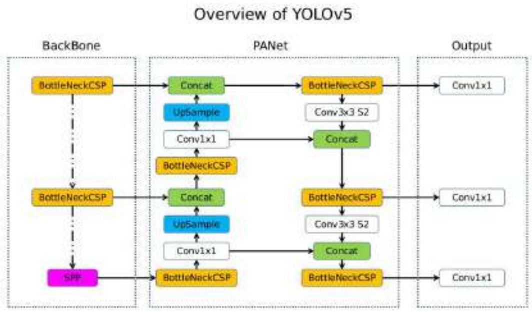


Figure 8: Architecture of the YOLOv5 Model

In this implementation, a photo mosaic of the island from UAS Imagery made with Agisoft software as a GeoTIFF file ($\approx$ 8.66 GB) for Majuro was considered. This imagery is of a much higher resolution and captures many features (especially, coconut tree canopies) in great detail compared to the MrSID format Natural Color imagery of Majuro from USDA. To collect images, the mosaic was loaded in QGIS as a raster layer and for each non-overlapping area, the jpeg images along with the corresponding .jgw files (JPEG World File is an ASCII text file that contains geographic information: location, scale, and orientation for a JPEG image) were captured, to create a dataset of 260 images.

LabelImg, a popular image annotation graphical tool developed by Tzutalin was used to label coconut tree instances from the images. Around 70 images consisting of nearly 5,645 trees were manually labeled with bounding boxes to create the training dataset for the model, which generates txt files consisting of the class label, normalized x, y centers of the bounding box (relative to the image width and height) along with the width and height of the bounding box in the YOLO format.

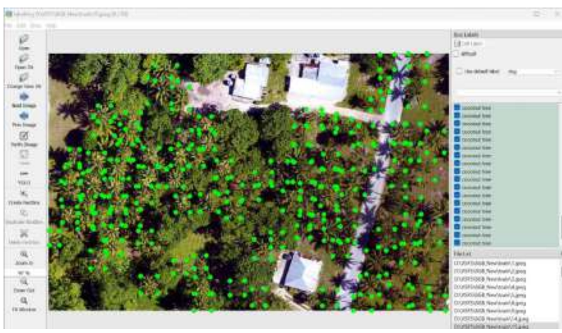
To train the YOLOv5 model, a Google Colab-based Jupyter notebook was used, taking advantage of the free Tesla T4 GPU access. The first step was to clone the YOLOv5 repository and install various dependencies. A pre-trained YOLOv5 small (yolov5s.pt) model was loaded from Ultralytics' Torch Hub. This model was trained on the COCO dataset and used as the starting point for transfer learning. The custom dataset was provided in a compressed format and a dataset.yaml file was created to define the dataset paths, number of classes, and class names for YOLOv5 training. Training was initiated using the YOLOv5 training script, where the key training parameters included: Image size of 512, batch size of 8, 75 epochs, Weights initialized from the pretrained yolov5s.pt model and 2 workers (number of CPU workers / processes) for data loading.

The best-performing model (best.pt) was loaded for inference on the new images. The model's accuracy was verified by testing it on a subset of unseen images to understand how well it was able to identify and locate coconut trees.

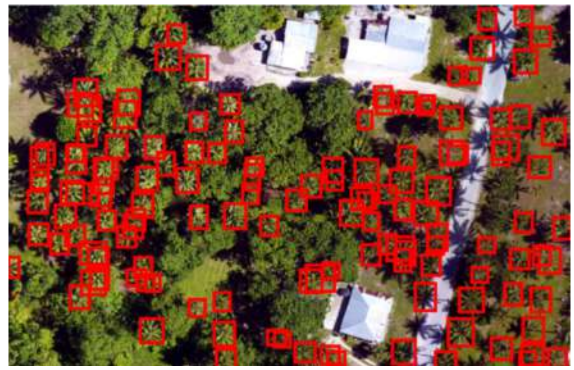
Another script was used to process images all the 260 images in batches of 30, where the YOLOv5 model detection was applied and map the detected objects to real-world geographic coordinates using .jgw files. A .jgw file is a plain text file with 6 numerical values that define how an image aligns with real-world coordinates in map projections via pixel size in x-direction, rotation / skew parameter in x and y directions, pixel size in y direction, Longitude and Latitude of the upper-left corner of the image. It extracts bounding box centers from detected objects and converts their pixel coordinates into geographic coordinates using corresponding JGW world files. The processed detections, including latitude, longitude, confidence scores, and class IDs, are saved as Shapefiles and KML files for geospatial analysis. The shapefiles were then loaded in QGIS to visualize the extent of coconut trees in the island by mapping them onto the original mosaic layer. The visual representation and outputs from this implementation are depicted in Figures 9 and 10.

(a) Majuro UAS Imagery visualization in QGIS

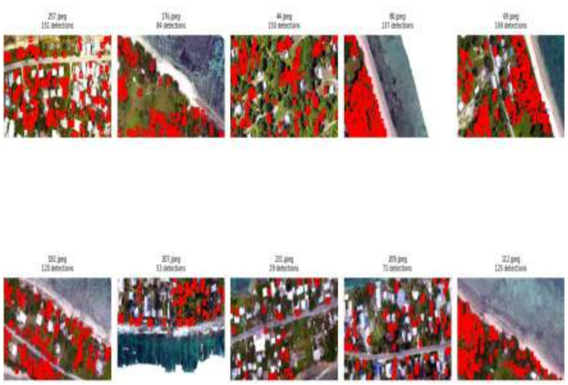
(b) A close up of the High-quality resolution imagery



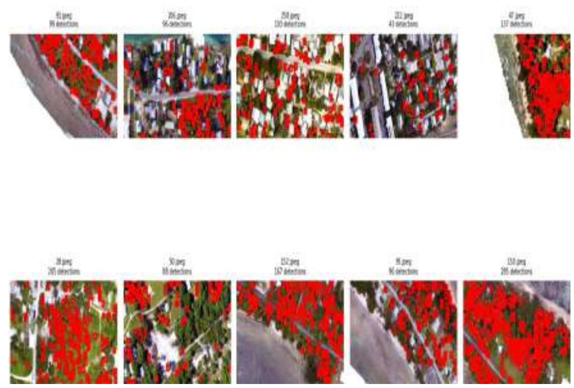
(c) Labeling coconut trees using the LabelImg tool



(d) Detections by the model on an unseen image

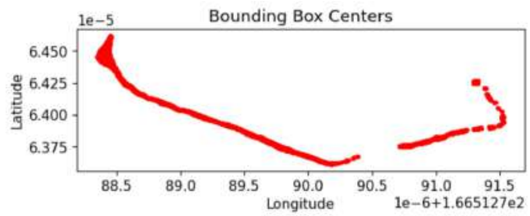


(e) Detections for multiple images



(f) Detections for multiple images

Figure 9: Workflow of the Deep Learning Implementation with YOLOv5 model



(a) Shapefile representation for the entire island

(b) Mapping the shapefile points on the original mosaic

(c) Closer view of shapefile points on the mosaic

(d) Close up view of the shapefile points which closely align with actual tree locations

Figure 10: Workflow of the Deep Learning Implementation using YOLOv5 model

4.2.2 Results

Once the training was completed, performance metrics were analyzed:

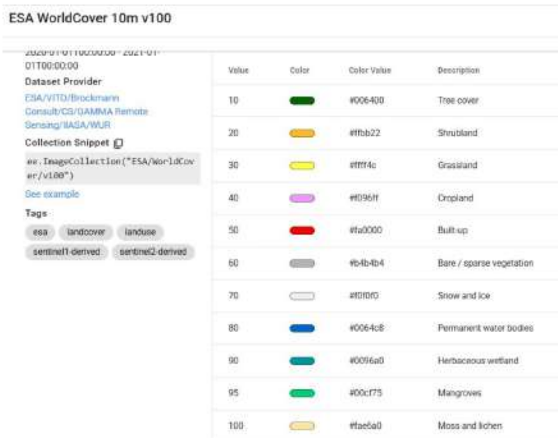
Metric	Value	Interpretation
mAP@0.5	0.66187	Good overall detection accuracy
mAP@0.5:0.95	0.26216	Moderate performance across different IoU thresholds
Precision	0.6902	High precision, meaning fewer false positives
Recall	0.64345	The model detects a decent number of objects
Box Loss (Train/Val)	0.08044 / 0.07378	Indicates how well object bounding boxes are predicted
Cls Loss (Train/Val)	0.0021 / 0.00181	Low classification loss, meaning accurate label predictions
Obj Loss (Train/Val)	0.35324 / 0.35789	Objectness loss, showing the confidence in object presence

Table 1: YOLOv5 Training Performance Metrics

The model achieved mAP@0.5 of 66.1%, indicating good detection accuracy. The final trained model had 157 layers and a total of 7053277 parameters. A total of 32,453 trees were detected for the entire Majuro island which were obtained as points from the shapefiles.

4.2.3 Generating Landcover and vegetation maps

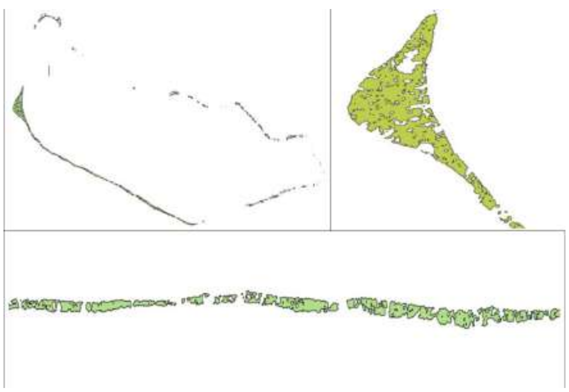
The goal of this implementation is not only to detect individual coconut trees but also to identify and analyze regions with significant coconut tree populations, including plantations and cultivated areas. This helps in understanding how coconut trees are distributed within the broader vegetation landscape. To achieve this, the ESA WorldCover 10m v100 global land cover classification map (2020–2021) was utilized in Google Earth Engine to generate a land cover map with 11 different land classes as in 11a. A land cover map was specifically created for Majuro, providing a visualization of vegetation and other land types shown in 11b. From this, a refined Tree Cover layer was extracted to highlight vegetated areas as in 11c. To assess the accuracy of this approach, the tree cover layer was mapped alongside the shapefiles containing coconut tree locations on the original mosaic in 11e. This visualization helps in identifying which regions within the general tree cover also contain significant coconut tree populations, providing valuable insights into the spatial distribution of coconut plantations.



Value	Color	Color Value	Description
10		#006400	Tree cover
20		#ffbb22	Shrubland
30		#ffff4c	Grassland
40		#f096ff	Cropland
50		#fa0000	Built-up
60		#b4b4b4	Bare / sparse vegetation
70		#f0f0f0	Snow and ice
80		#0064c8	Permanent water bodies
90		#0096a0	Herbaceous wetland
95		#00cf75	Mangroves
100		#fae6a0	Moss and lichen

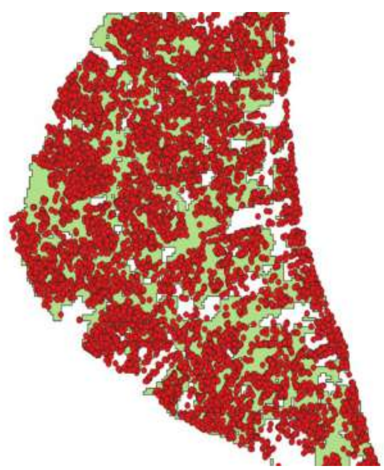
(a) 11 different classes of vegetation

(b) Land Cover map visualization for Majuro



(c) Tree Cover map visualization for Majuro

(d) Mapping the tree layer on the original mosaic



(e) Tree Cover map with coconut tree points from shapefiles

Figure 11: Workflow of creating the vegetation maps

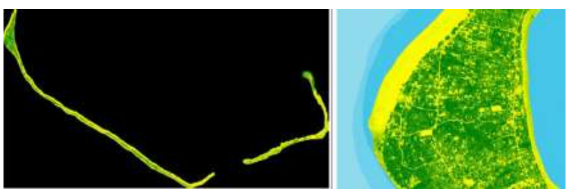
Two other vegetation maps were custom created using Google Earth Engine (GEE). First, The Vegetation Atmospherically Resistant Index (VARI) map in 12a was generated using a formula that combines the blue (B1), green (B2), and red (B3) bands.

$$VARI = \frac{Green - Red}{Green + Red - Blue}$$

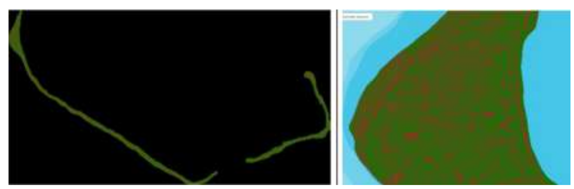
This index helps enhance vegetation visibility while reducing the influence of atmospheric effects. High vegetation cover appears in green, representing areas with significant plant growth. Non-vegetated surfaces like roads, buildings, and barren land appear in yellow due to their lower VARI values. This visualization provides a robust way to distinguish vegetation from other land cover types in the region.

The second map was created using only the blue band (B1) in 12b to highlight tree cover. The blue band is useful for vegetation mapping because healthy vegetation strongly absorbs blue light and reflects it less compared to non-vegetated surfaces. Tree-covered areas are visualized in green, indicating the extent of vegetation. Non-vegetated areas such as roads, urban areas, or bare land appear in brown, making it easier to assess tree plantation extent. This map aims to provide a simplified yet effective way to differentiate tree-covered regions from non-vegetated areas.

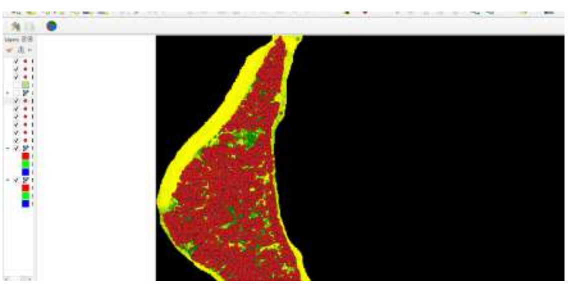
The goal of this task is to visualize the predicted locations of coconut trees, represented by the centers of bounding boxes from the YOLOv5 model's output, overlaid onto the tree cover map of the Majuro Islands. In this visualization, red points indicate areas with coconut tree plantations, while green regions (derived from the original Land Cover classification map or the VARI Index-based Tree Cover map) represent other types of vegetation across the island. This approach helps in distinguishing coconut tree populations from the broader vegetative landscape.



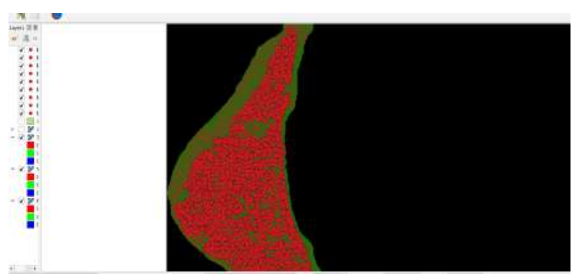
(a) Vegetation Map based on VARI index



(b) Vegetation Map based on B1 band



(c) Tree Cover map visualization for Majuro



(d) Mapping the tree layer on the original mosaic

Figure 12: Workflow of creating additional vegetation maps

A separate map was created to visualize the extent of the tree cover in the complete vegetation map for Majuro represented in Figure 13.

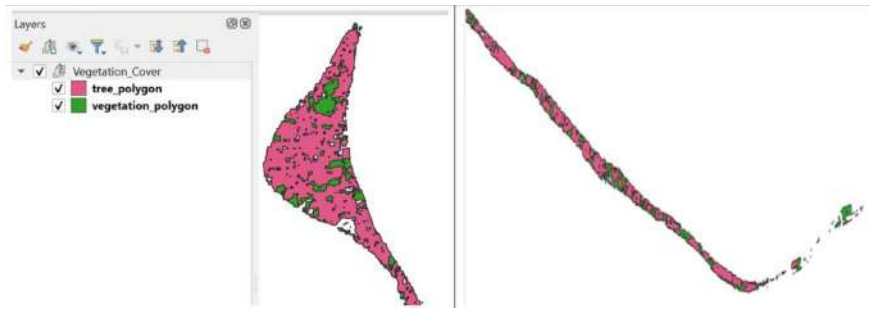


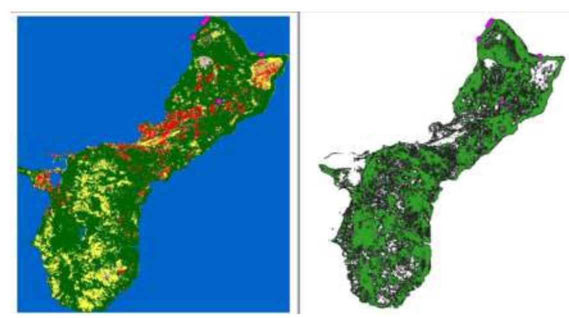
Figure 13: Tree polygon represents the Tree cover (pink) and vegetation polygon represents different types of vegetation (green)

4.3 Deep Learning for coconut detection in other areas

The goal was to evaluate how well the trained weights from the custom YOLOv5 model for Majuro could generalize to identifying coconut trees in other atolls and islands. Google Earth Pro was used to obtain high-resolution satellite imagery and capture JPG images. For geographic mapping, the latitude and longitude of each image center were recorded in a TXT file and mapped to image pixels using a Python script. This approach was applied to images from Guam, Kapingamarangi, and Chuuk to assess the model's adaptability across different regions.

The best.pt model was directly used for object detection on images from Chuuk, Guam, and Kapingamarangi. However, the performance was poor: detections were sparse and often misidentified objects. This discrepancy may be due to differences in image resolution, lighting conditions, and other environmental factors compared to the original training dataset which had a higher-resolution and better image quality.

(a) Vegetation layer for Chuuk, red points represent the very few detections possible



(b) Land cover layer for Guam, pink points represent the very few detections possible

(c) Sparse detection of coconut trees in Kapingamarangi

Figure 14: Poor detection of coconut trees using the same best.pt model for Chuuk, Guam, and, Kapingamarangi

For Kapingamarangi, fine-tuning was applied to the best.pt model by further training it with additional images containing multiple bounding boxes of coconut trees specific to this region. The goal was to improve detection accuracy by adapting the model to the local tree distribution.

Fine-tuning a machine learning model like YOLOv5 involves continuing the training process on new, domain-specific data while leveraging the knowledge already learned from pretraining. This technique helps improve performance for specialized tasks. The best.pt model trained for Majuro was loaded into Google Colab and fine-tuned using a batch size of 8 for 15 epochs. This approach significantly enhanced the model's ability to detect trees compared to using the best.pt model directly.

4.3.1 Results

Metric	**Value**
mAP@0.5	0.13918 (Very Low)
mAP@0.5:0.95	0.04141 (Very Low)
Precision	0.20052
Recall	0.21178
Box Loss (Train)	0.07295
Box Loss (Val)	0.07054
Cls Loss (Train)	0.00157
Cls Loss (Val)	0.00150
Obj Loss (Train)	0.23441
Obj Loss (Val)	0.33419
Learning Rate (LR0)	0.05633 (Unusually High)

Table 2: Performance Metrics for fine-tuning (Best Epoch: 8)

Despite fine-tuning, the model still performed poorly on Kapingamarangi. This could be attributed to the domain gap between the two regions: Majuro's dataset may differ significantly from Kapingamarangi's in terms of tree density, appearance, and environmental factors, making generalization difficult. While fine-tuning improved detection accuracy compared to using the best.pt model without adaptation, certain challenges remained:

- Misidentified trees: Some non-coconut trees were incorrectly classified.
- Occlusion and Clutter: Smaller, overlapping, or densely packed coconut trees were often missed by the model.
- Labeling Differences: Variations in annotation quality and tree characteristics between datasets likely affected training outcomes.

These factors suggest that further improvements, such as additional labeled data, data augmentation, and longer fine-tuning, may be required for better adaptation to the landscape of each region.

5 Conclusion

This study demonstrates the potential of deep learning-based object detection for mapping and monitoring coconut tree populations using satellite imagery. By leveraging YOLOv5, coconut trees were located successfully, generating a foundational geospatial dataset stored in shapefiles and maps. These datasets provide a valuable resource for long-term vegetation monitoring, eliminating the need for repeated model retraining and ensuring more efficient analysis over time.

However, the findings also highlight key challenges in model generalization. The variations in detection accuracy across different regions, such as Majuro, Chuuk, Guam, and Kapingamarangi demonstrate that a single trained model does not perform consistently across all landscapes. Differences in vegetation density, image resolution, annotation quality, and environmental conditions emphasize the need for localized model adaptation. Fine-tuning on region-specific datasets significantly improved performance, but further enhancements are required for robust, scalable detection across diverse geographic areas.

6 Future Directions

To improve accuracy and expand the applicability of this approach, future work can focus on the following aspects:

Expanding the Training Dataset: Incorporating more labeled coconut tree images from various islands and environments to enhance model robustness. Additionally, multi-seasonal and multi-sensor satellite imagery can be included to account for variations in tree appearance due to lighting, weather, and seasonal changes.

Advanced Model Optimization: Other kinds of deep learning architectures (e.g., YOLOv8, Faster R-CNN) can be trained to compare performance on coconut tree detections. Additionally, deep learning models can be directly combined with remote sensing indices (NDVI, VARI, etc.) to improve tree classification and separate coconut trees from other vegetation.

Integration with Remote Sensing tools: Use of Google Earth Engine (GEE) and GIS-based spatial analysis to monitor coconut tree distribution, track changes over time, and study environmental factors affecting coconut tree populations.

Temporal Analysis and Change Detection: Compare historical and future datasets to monitor trends in deforestation, assess climate change impacts, and evaluate post-disaster recovery in coastal and island regions. Develop an automated coconut tree monitoring system that periodically updates maps with newly acquired satellite images.

By advancing these research directions, this work can contribute to precision agriculture, ecological conservation, and climate resilience strategies, ensuring more effective and sustainable management of coconut tree populations worldwide.

References

[1] Wikipedia contributors. *Pacific Ocean — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=Pacific_Ocean&oldid=1279610302. [Online; accessed 14-March-2025]. 2025.

[2] R. R. Thaman. "Atolls of the Tropical Pacific Ocean: Wetlands Under Threat". In: *The Wetland Book: II: Distribution, Description and Conservation*. Ed. by C. Max Finlayson et al. Dordrecht: Springer Netherlands, 2016, pp. 1–25. DOI: 10.1007/978-94-007-6173-5_270-1. URL: https://doi.org/10.1007/978-94-007-6173-5_270-1.

[3] Wikipedia contributors. *Coconut — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=Coconut&oldid=1275633665. [Online; accessed 14-March-2025]. 2025.

[4] Michael W Burnett et al. "Satellite imagery reveals widespread coconut plantations on Pacific atolls". In: *Environmental Research Letters* 19.12 (Dec. 2024), p. 124095. DOI: 10.1088/1748-9326/ad8c66. URL: https://dx.doi.org/10.1088/1748-9326/ad8c66.

[5] Julia Lenhardt. *Use deep learning to assess palm tree health*. https://learn.arcgis.com/en/projects/use-deep-learning-to-assess-palm-tree-health/. Accessed: 2024-10-20.

[6] Wei Liu et al. "SSD: Single Shot MultiBox Detector". In: *Computer Vision – ECCV 2016*. Springer International Publishing, 2016, pp. 21–37. ISBN: 9783319464480. DOI: 10.1007/978-3-319-46448-0_2. URL: http://dx.doi.org/10.1007/978-3-319-46448-0_2.

[7] Joseph Redmon et al. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. arXiv: 1506.02640 [cs.CV]. URL: https://arxiv.org/abs/1506.02640.