2D Mapping and Forest Cover Detection of Locality Using U-Net and Drone Imagery

Group 4

Flight Testing Laboratory

Aerospace Engineering Department, Indian Institute of Technology, Kharagpur

ABSTRACT

This project focuses on enhancing forest monitoring through the integration of advanced technologies. By utilizing orthorectified imagery captured by Unmanned Aerial Vehicles (UAVs) and employing machine learning algorithms, specifically U-Net models, trained with labeled data, we aim to estimate the forest cover of a region. Our approach also involves creating precise Digital Surface Models (DSMs) to accurately depict the vertical structure of forest canopies. This innovative methodology promises to significantly improve the efficiency and accuracy of forest monitoring efforts, contributing to the preservation and sustainable management of forest ecosystems.

INTRODUCTION

Amidst the global urgency to address the escalating rates of deforestation and its multifaceted environmental ramifications, the imperative for robust monitoring and intervention measures has become increasingly apparent. Conventional methodologies, while foundational, often exhibit limitations in providing the requisite precision and scalability necessary for comprehensive and accurate assessments of forest ecosystems. Recognizing this critical juncture, our project emerges as a pioneering effort to confront these challenges head-on through the integration of advanced technologies, including Unmanned Aerial Vehicles (UAVs). By harnessing the capabilities of orthorectified imagery captured by UAVs and sophisticated machine learning algorithms, we aim to transcend the constraints of traditional monitoring approaches. Through meticulous analysis and strategic deployment of these innovative tools, our project aspires to redefine forest monitoring paradigms, offering a transformative solution that not only enhances the efficiency and accuracy of estimating average forest cover and height but also promises to catalyze significant strides in the preservation and sustainable stewardship of our planet's invaluable forested landscapes.

Problem statement:

The goal is to create an accurate orthorectified 2D map to assess forest cover and determine the average height of the forest canopy, using digital surface modelling, for environmental monitoring purposes.

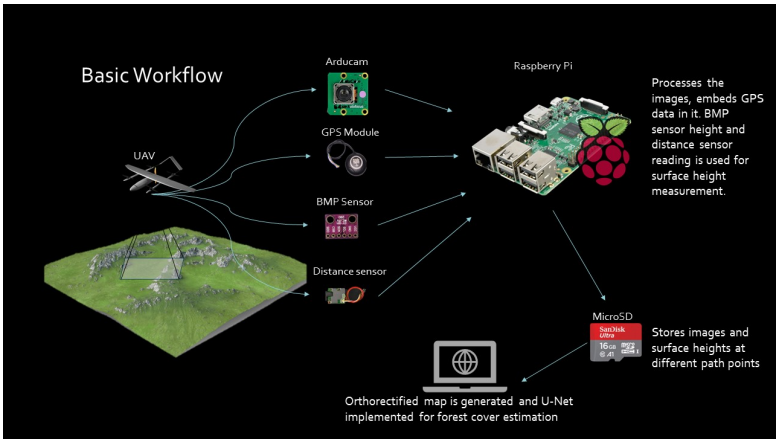

Figure 1. Workflow Diagram.

OBJECTIVES:

Aerial Imaging, Orthorectification, and 2D Mapping:-

The objective is to employ Pix4Dmapper, an advanced photogrammetry software, to conduct orthorectification, and 2D mapping of forested areas from aerial imagery captured by UAVs. By harnessing the capabilities of Pix4Dmapper, the aim is to correct distortions arising from terrain relief and camera angles in the aerial imagery captured by UAVs. Through this process, orthomosaic images will be generated, offering precise and true-to-scale representations of the forested terrain. Additionally, the generated 2D maps will enable comprehensive analysis and monitoring of the forest cover, facilitating informed environmental management decisions.

Estimation of Forest Cover:-

By employing the orthorectified 2D maps generated by Pix4Dmapper, we intend to utilize a machine learning approach, specifically a U-Net model, to accurately delineate the forested areas within the target region. This integration of advanced technology allows for automated analysis of forest cover, facilitating efficient and accurate assessments. To train the U-Net model effectively, we will utilize Label Studio, a versatile data labeling tool, to create precise masks delineating forested areas within various sample aerial images. By leveraging Label Studio, we ensure the creation of accurate and comprehensive training datasets, enhancing the performance and reliability of the machine learning model. This streamlined workflow significantly improves the efficiency and effectiveness of automated forest cover analysis.

Estimation of Average Forest Height:-

In addition to leveraging the precise measurements enabled by orthorectified imagery, the report aims to enhance the accuracy of estimating the average height of the forest canopy by creating a Digital Surface Model (DSM). This model will be constructed using data from aerial imagery and Infrared Distance Sensor.The collected height data will be complemented with corresponding spatial information obtained from the orthorectified 2D maps,a Digital Surface Model (DSM) can be made.

MATERIALS REQUIRED

- Fixed Wing UAV
- Arducam 64MP Autofocus Camera Module for Raspberry Pi
- Raspberry Pi 4 Model-B (4GB)
- SanDisk Micro SD/SDHC 16GB Class 4 Memory Card
- Infrared Laser Distance Sensor [50m/80m]
- GPS Module
- BMP(Barometric Pressure Sensor)
- Battery
- Connecting Wires

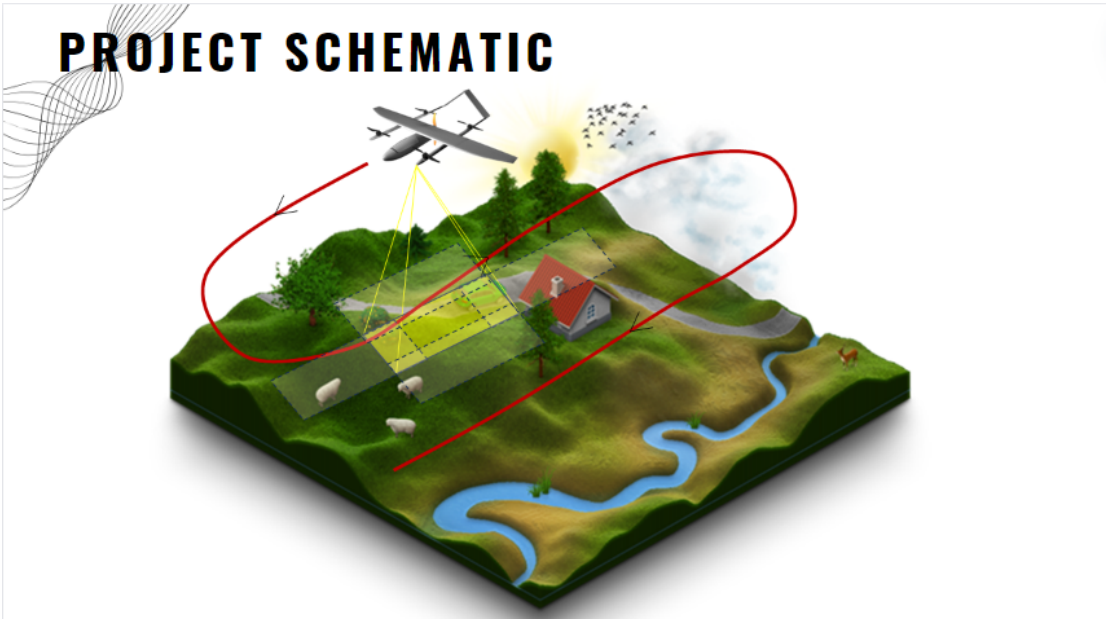

Figure 2. Project schematic.

1 U-NET MODEL

The U-Net model is a convolutional neural network architecture widely used for image segmentation tasks in machine learning and computer vision, particularly in medical imaging. Its distinctive U-shaped structure comprises a contracting path, which captures context and features through convolutional and max-pooling layers, and an expansive path, symmetric to the contracting path, which gradually recovers spatial information through upsampling and convolutional layers. Notably, the architecture incorporates skip connections, enabling the direct concatenation of information from early layers in the contracting path with corresponding layers in the expansive path, thus preserving fine-grained details and spatial information during the upsampling process. The final layer typically consists of a convolutional layer with a sigmoid activation function, producing a binary segmentation mask indicating the presence or absence of objects of interest in the input image. This architecture excels in tasks requiring high-resolution output and precise object localization, such as medical image analysis and semantic segmentation in natural images.

We trained an already existing machine learning u-net model with already annotated aerial images containing 30 epochs, which means that the model undergoes the training process using the entire training dataset 30 times, taking 32 annotated images from the dataset at a time. Each epoch was followed by a validation process, which was made visible in the code, and the model got better and better with each epoch. The image directory we used contained about 5500 images and their corresponding masks. The masks of the images were in black and white format with the forest cover in white shade and the remaining areas in black.
The training process then involved a testing process on images from the dataset itself.

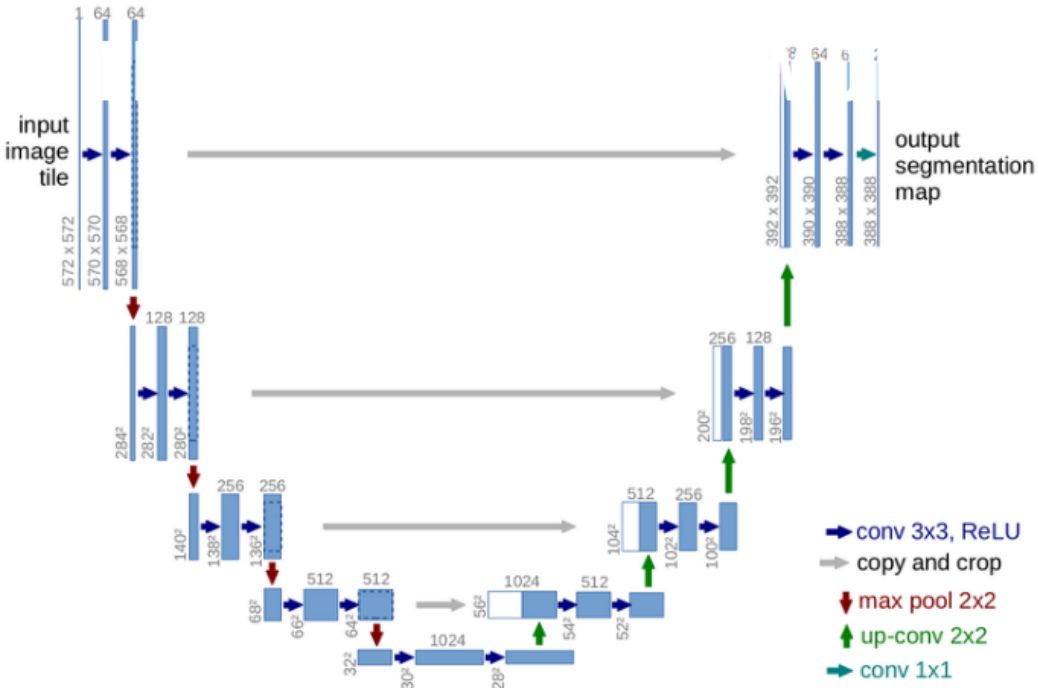

Figure 3. Process behind the U-net Model

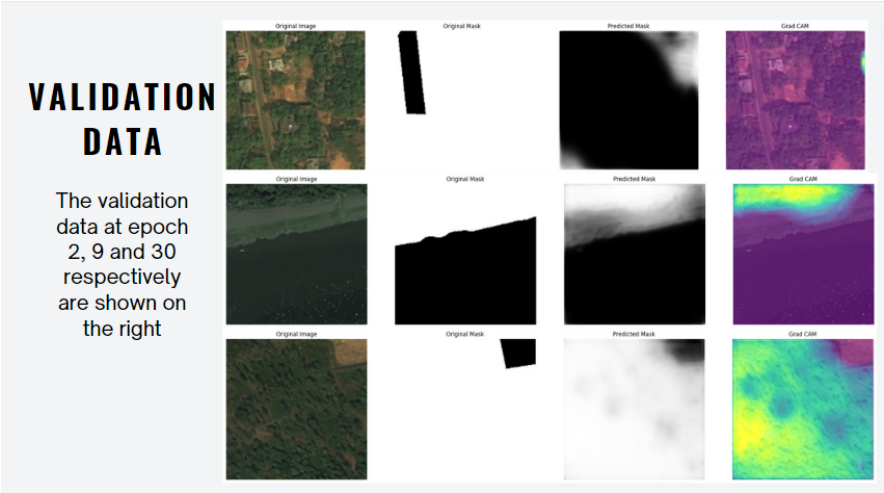

Figure 4. Validation data

The result we got from the model by inputting custom images was in a black and white gradient format. We converted individual pixels to either pure black or white format based on the intensity to increase sharpness. But this greatly affected the accuracy of output. Hence decided to use gradient format. The code for the U_NET model trained can be found at:
`https://github.com/RahulS-16/FTL_Group4.git`

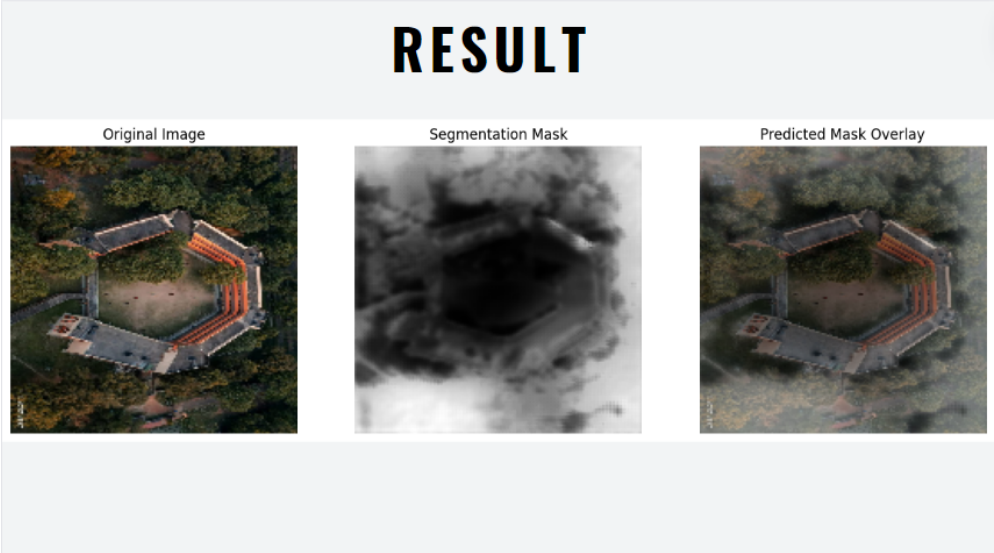

Figure 5. Result obtained from the dataset we found online.

However we later realized that the training dataset that we found online was not annotated properly and had a lot of inaccuracies. So we decided to annotate images ourselves to make our own dataset and hence increase the classification accuracy of the algorithm. We used the software Label studio for doing this task.

2 LABEL STUDIO

Label Studio is an open-source data labeling tool designed to streamline the process of annotating data for machine learning models. It provides a user-friendly interface for creating labeled datasets by allowing users to define labeling tasks and manage labeling projects efficiently. Label Studio supports various data types, including text, images, audio, and video, making it versatile for a wide range of machine learning applications. Its flexible architecture allows integration with popular machine learning frameworks and libraries, enabling seamless workflow integration from data labeling to model training and evaluation. Overall, Label Studio empowers developers and data scientists to annotate data quickly and accurately, accelerating the development of machine learning models.

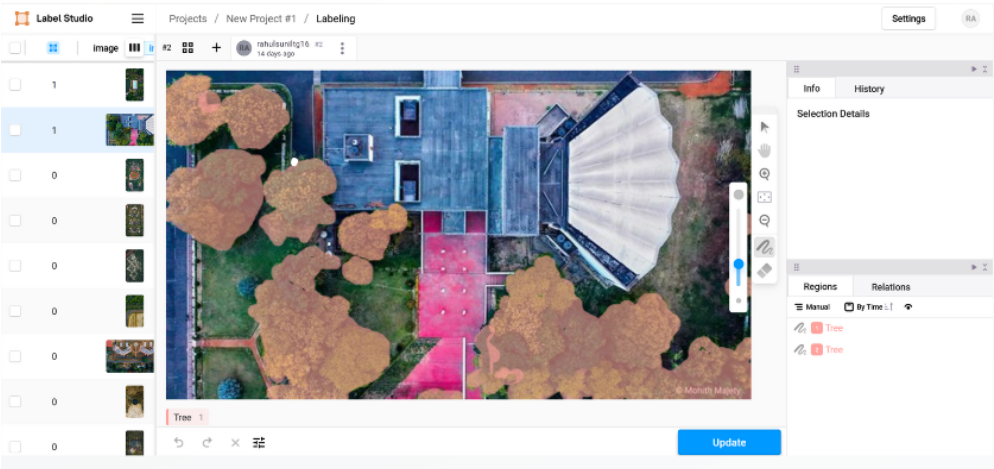

Figure 6. Working in label studios.

We utilized Label Studio as our primary tool for annotating our dataset, employing it to generate masks

crucial for training the U-Net model. The process involved meticulously outlining and labeling specific features or regions of interest within our data, such as trees in the images . With Label Studio's intuitive interface, we efficiently delineated these areas, ensuring accurate and detailed annotations.

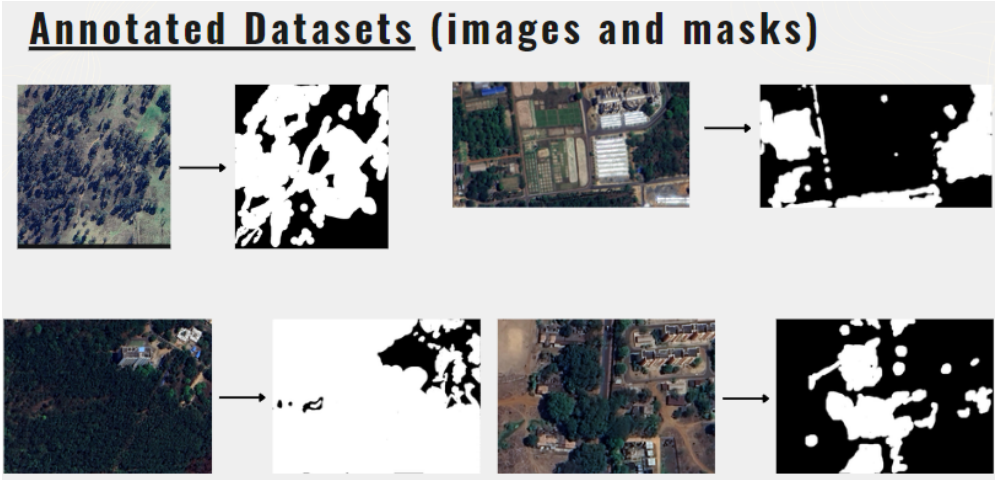

Figure 7. Annotated images and their corresponding masks.

The generated masks serve as ground truth data for training the U-Net model. By training the model on these meticulously annotated masks, it learns to accurately segment and identify similar features or regions within new, unseen data.
This annotated dataset, comprising both the original data and corresponding masks, forms the foundation for training our U-Net model. The model learns to predict masks for unseen data, enabling automated segmentation tasks with high precision and efficiency.
Results obtained after using new masks:

Figure 8. Result obtained implementing the trained U-Net model.

3 ORTHORECTIFICATION

Orthomosaic is a large, map-quality image with high detail and resolution made by combining many smaller images called orthophotos. Orthorectification is a crucial process in remote sensing and aerial

photography that involves correcting the geometric distortions in satellite or aerial images to create a true representation of the Earth's surface. This can be used to measure true distances since it accurately represents the earth's surface. The true distances obtained can further be used to measure area and volume. This procedure adjusts the raw imagery for topographic relief, sensor distortions, and camera tilt to ensure that the scale of the imagery is uniform across the entire image. The result is a map-like image that accurately portrays the features on the ground in their exact geographic locations. The distortions that orthorectification corrects arise from several factors. Firstly, the Earth's surface is not flat, and variations in elevation can cause objects to appear shifted in imagery. Secondly, the angle at which the photograph is taken (camera tilt) and the motion of the sensor platform introduce perspective distortions. Lastly, the optical system of the camera itself can cause lens distortions. To achieve orthorectification, specialized software uses complex algorithms that combine raw image data with geographic and sensor information. We use Pix4d software for the orthorectification of aerial photos taken using the drone.

One of the main use cases for an orthomosaic is simply to get an updated view of a large area of land. Google Earth is helpful, but often the imagery is outdated. A new development project could be finished, new construction projects could be underway, or the landscape could have changed from natural disasters. Real estate agents may also want an updated view of property to see if it is a good fit for their needs. Construction firms also find orthomosaics particularly useful. They can get updated views of their construction sites on a regular basis, and they can leverage the accuracy of the orthomosaics for precise measurements. Orthomosaics are occasionally used in legal cases too. Certain cases require updated and accurate imagery of a property in question. Orthomosaics provide the necessary imagery needed for the legal proceedings. Ecological health monitoring by vegetation cover assessment is also possible using orthomosaics. . Whatever the reason is, orthomosaics are the best way to combine aerial images for an updated view of a land.

Figure 9. Aerial and orthomosaic.

Pix4dmapper

Pix4D is a leading photogrammetry application for professional drone mapping, offering robust tools for transforming raw aerial imagery into precise, georeferenced 2D maps and 3D models. In this project, Pix4D software is utilized to create an orthomosaic from a series of aerial photographs captured by a drone. The process begins by capturing overlapping photographs from our drone, ensuring each photo contains embedded metadata on latitude, longitude, and altitude. This geotagging is crucial as it enables Pix4D to accurately stitch the images together, maintaining precise geographic positioning throughout the resulting orthomosaic map. The GPS coordinates along with the altitude can be encoded into an image using the Geosetter software. One of the key strengths of Pix4D is its highly automated process, which allows users to generate models and maps with minimal manual intervention.

Testing of Pix4dmapper

Images were taken from the third floor of RP Hall and an orthomosaic was generated using Pix4DMapper.

Figure 10. Images taken at the top of RP hall.

The Obtained Orthomosaic was:

Figure 11. Obtained orthomosaic.

3.1 Increasing image resolution

The resolution of the orthorectified map can be increased upto 4x using deep learning models such as EDSR (Enhanced Deep Residual Networks for Single Image Super-Resolution) and OpenCV computer vision library in python.

Listing 1. code for increasing image resolution

```
import cv2
from cv2 import dnn_superres
sr = dnn_superres.DnnSuperResImpl_create()
path ='EDSR_x4.pb'
sr.readModel(path)
sr.setModel('edsr',4)
sr.setPreferableBackend(cv2.dnn.DNN_BACKEND_CUDA)
sr.setPreferableTarget(cv2.dnn.DNN_TARGET_CUDA)
image = cv2.imread('satellite-aerial-photos-of-earth-1.jpg')
upscaled = sr.upsample(image)
cv2.imwrite('upscaled_test.png', upscaled)
```

Figure 12. increased resolution.

4 DIGITAL SURFACE MODEL

A Digital Surface Model (DSM) is a digital representation of the Earth's surface that includes all objects and features on the terrain, such as buildings, vegetation, and other structures. It is a three-dimensional

model that provides information about the height or elevation of objects on the Earth's surface. DSMs are typically created using remote sensing technologies such as LIDAR or photogrammetry, which use laser pulses or aerial photographs to measure the distance between the Earth's surface and the sensor. The data collected by these technologies is processed to create a three-dimensional model of the terrain, including all the objects and features on the surface.
In our case we plan to do so by collecting the (x,y) coordinates from the GPS data. The height will be obtained by taking the difference between the barometric height and the infrared distance sensor reading.

We will take zi=Hmax-Hi to get the ith z coordinate with respect to the lowest point on the surface.
Then our task is to find a z as a continuous form of (x,y) or to plot a z profile on the 2-D surface. For this we perform a regression on the data points to obtain a curve. We do this regression using the random forest regression decision tree technique.

In the sample implementation we have generated the random data from the code itself.
Here goes the code for the random forest regression:

Listing 2. code for DSM

```
 import numpy as np
import plotly.graph_objs as go
from sklearn.ensemble import RandomForestRegressor

def generate_synthetic_data(n_samples=100):
    np.random.seed(42)
    x_values = np.random.uniform(0, 100, n_samples)
    y_values = np.random.uniform(0, 100, n_samples)
    z_values = np.random.uniform(0, 100, n_samples)
    return x_values, y_values, z_values

# Generate synthetic data
x_values, y_values, z_values = generate_synthetic_data()

# Create a Random Forest Regressor with increased complexity
(overfitting)
rf_regressor_overfit = RandomForestRegressor(n_estimators=500,
max_depth=10, max_features=10, random_state=42)

# Reshape x and y for sklearn input
X = np.column_stack((x_values, y_values))

# Fit the Random Forest model
rf_regressor_overfit.fit(X, z_values)

# Create a grid of points in the xy-plane
x_grid = np.linspace(min(x_values), max(x_values), 100)
y_grid = np.linspace(min(y_values), max(y_values), 100)
x_grid, y_grid = np.meshgrid(x_grid, y_grid)
xy_grid = np.column_stack((x_grid.flatten(), y_grid.flatten()))

# Predict z-values for the grid using the trained model
z_predictions_overfit = rf_regressor_overfit.predict(xy_grid)

# Reshape predictions for plotting
z_grid_overfit = z_predictions_overfit.reshape(x_grid.shape)
```

```
# Specify a solid color for the surface
surface_overfit = go.Surface(
    x=x_grid,
    y=y_grid,
    z=z_grid_overfit,
    colorscale=[[0, 'rgb(169,169,169)']],  # Solid grey color
    opacity=1,  # Solid color, so opacity is set to 1
    name='Overfitted Surface'
)

# Create the layout for the 3D plot
layout = go.Layout(
    scene=dict(
        xaxis=dict(title='X'),
        yaxis=dict(title='Y'),
        zaxis=dict(title='Z'),
    )
)

# Create the figure and add surface plot
fig = go.Figure(data=[surface_overfit], layout=layout)

# Show the interactive plot
fig.show()
```

The following link redirects you to the code implementation of the random forest. The 3d model can be viewed there.
https://colab.research.google.com/drive/1_9ysYZ_6pGZgP1WYhOFXoxwA6F8sq9Jq?usp=sharing
Here are a few snapshots of the 3d model:

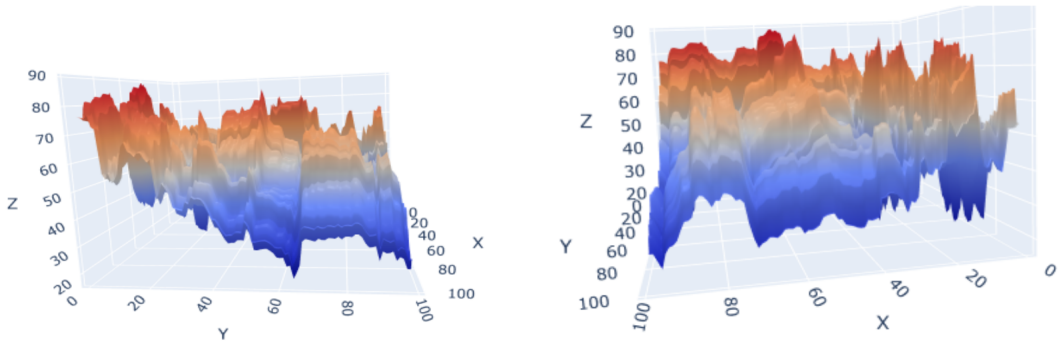

Figure 13. Obtained Digital Surface Model.

GROUP MEMBERS

Allwin Johnjo Prince -21AE10002
Anfal S-21AE10003
Arjun Biju-21AE10007
Nikhith Mathai Manoj-21AE10027
Safa N-21AE10033
Aswin D Menon-21AE10044
Adithyan US-21AE10049
Rahul Sunil-21AE10050
Jasothan V-21AE30008
Sagar KP-21AE30021
Nirvik Paul-21AE3FP15