UAV ground detection and tracking systems: Tree detection and tracking change species and amount of tree

Christian Ekeigwe
Purdue University
Computer Information Technology, Cybersecurity
West Lafayette, Indiana, United States
cekeigw@purdue.edu

line 1: Daehyeon Jeong
line 2: *Pusan National University*
line 3: *Aerospace Engineering*
line 4: Pusan, Republic of Korea
line 5: empcik@pusan.ac.kr

Jaeyoung Shim
Pusan National University
Computer Engineering
Pusan, Republic of Korea
jysim0129@pusan.ac.kr

Jeonghwan Kang
Pusan National University
Computer Engineering
Pusan, Republic of Korea
chanelacy@outlook.com

Seoungheong Jeong
Pusan National University
Computer Engineering
Pusan, Republic of Korea
korssouna1030@pusan.ac.kr

***Abstract*—This paper examines the technic and algorithm about detect tree using Unmanned Aerial Vehicles (UAV) in real-time. The purpose is grasp distribution of tree species using real-time air-view videos contain GPS information. This examination use 'Haar Cascades'[1] to detect tree and using Raspberry Pi to receive real-time video data.**

This examination shows 3% of accuracy in when using 17000 units of data. Real-time data receive using Raspberry Pi on network shows 4% of accuracy.

The limitation of examination is 'Haar Cascades' need enough data to detect tree, but this paper examines only 1700 units of data, so the accuracy is little low. And Raspberry Pi CPU core speed is slow so that can't handle high resolution video in real-time.

Keywords—OpenCV, Haar Cascades, UAV, tree detection

I. Introduction

The BGCI(Botanic Gardens Conservation International) estimated that at least 11,000 tree species are threatened with extinction in the wild[2]. This mean the importance of tracking change species and amount of tree is increased.

The use of remote sensed data in forest management is a common practice, since it allows the collection of a large amount of information about the environment with less human effort in field. This is even more important in large or inaccessible areas, where the field work is more exhausting and expensive.

In this examination design a novel way to fly over tree by UAV and determine what kind tree it is. Then record the GPS point of the tree. At the end point of a flight a map will show all of the GPS points of tree.

Detect tree amount and classify tree species with openCV 'Haar Cascades'. 'Haar Cascades' needs a lot of positive images (images of trees) and negative images (images without trees) to train classifier. This examination use 7 units of video and split it into 253 units of images. And convert color-image to gray-image for clearness of contour line. Examination use Raspberry Pi and camera module to test real-time processing of classify tree data. Connect to Raspberry Pie remotely and get real-time air-view video of tree. This attempt to use Raspberry Pie shows 00% of accuracy in 00x00 resolution. Finally classified data is show all of the GPS points for the specific variety of tree.

This technic make possibility to contribute detect tree and find new species or tracking change of tree environment for UAV consumer not commercial purpose user. Also government or Environment group can use this technic for tracking tree species and amount. But because of small units of data examination shows low tree detect accuracy.

II. Materials and Methods

A. Materials

The subject of this examination is distinguish tree species and detect tree and its amount using artificial intelligence. This examination use weak artificial intelligence(AI). The weak AI is artificial that is limited to a specific or narrow area. And that's the reason why this examination use weak AI to detect and classify trees.

B. Methods

For detect trees and its species, in this examination use Haar feature-based cascade classifiers. Many positive and negative images needed for detect stuff well. This examination use 00 units of positive image and 00 units of negative images. The positive image is part of UAV video that sliced with 0.05 second. This images convert to gray-image for clearness of contour line. And for the best result, this examination use OpenCV Cascade Classifier which is well-known for easy to use. To see the real-time detect of tree, Raspberry Pie with camera module is prepared. And access to the Raspberry Pie and receive air-view video data that contains GPS data. At the end classify tree amount and species.

III. Results and Disccustion

This examination divides 3 step. First one is finding the way to how to detect tree. Second step is gathering data for training and increase accuracy. The final step is conduct real-time tree detection using UAV.

A. Method for tree detection

This paper examines two methods for detect tree. First one is edge detection. Edge detection process serves to simplify the analysis of images by drastically reducing the amount of

data to be processed, while at the same time preserving useful structural information about object boundaries [3]. The first step for Canny edge detection is eliminate noise because edge detection results are highly sensitive to image noise. Gaussian blurring is used for decreasing noise. This method can find best line for find tree edge by adjust sigma value and Gaussian kernel. And finally use Non-Maximum Suppression for detect tree but the parameter value is wide video to video, so this method (Canny edge detection) is not proper for detect tree. As a result, this method is useful to distinguish tree or not but there are two problems. One is that it is hard to make criterion for detect tree. The second problem is handling with each tree leaf shape because the tree leaf shape is diverse tree by tree.

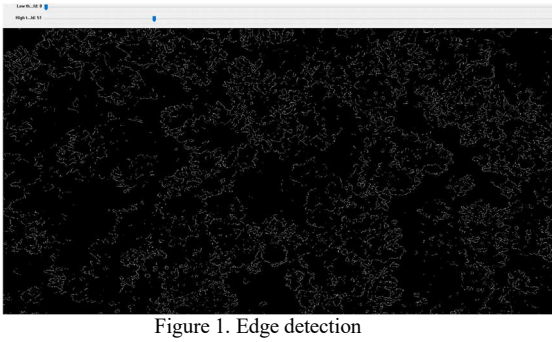

Figure 1. Edge detection

So, in this paper examines processed with second solution, 'Haar Cascades.' 'Haar Cascades' is extremely rapidly and achieving high detection rates. The key insight of cascade of classifier is that smaller, and therefore more efficient, boosted classifier can be constructed which reject many of the negative sub-windows while detecting almost all positive instances. It means that for tree detection, it only needs enough positive and negative image.

B. Gathering data and increase accuracy

At first, this examination tries to use Google Map because it has many data set but Google Map has only low resolution so the data set should be gathering in other way.

Figure 2. Google Map air-view image

Using UAV take 7 units of video of tree air-view contains 107 units of positive and 146 units of negative data.

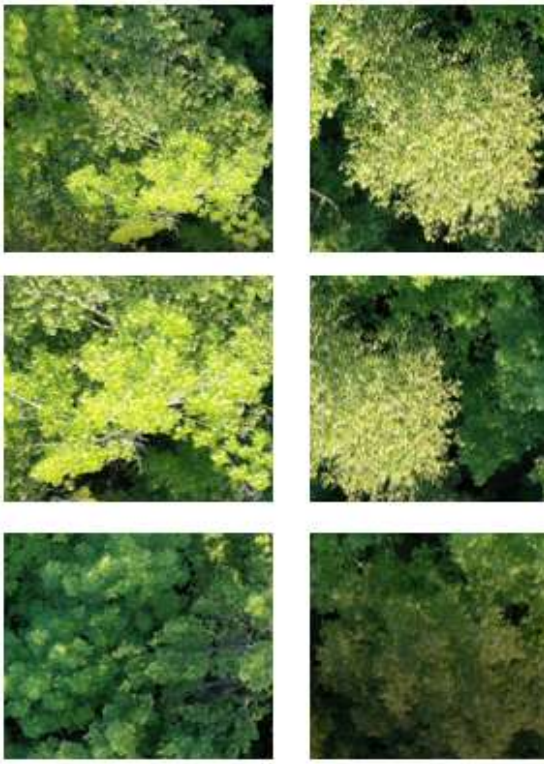

Figure 3. Positive data from UAV video

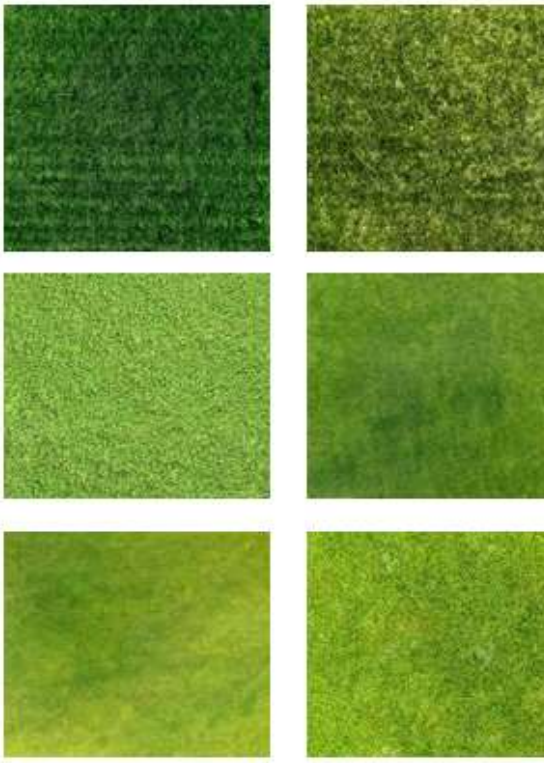

Figure 4. Negative data from UAV video

This examination use 'Cascade Trainer' version 3.3.1 for easy training. It helps selecting training data and has automated training system.

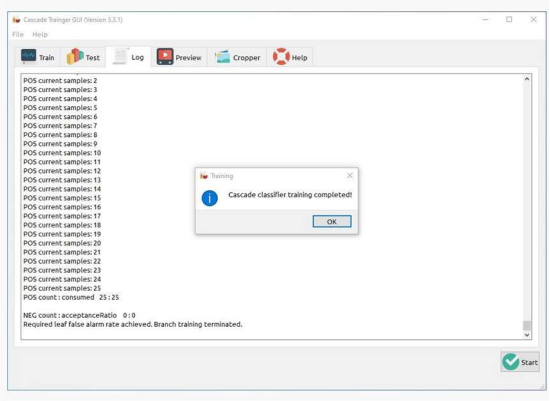

Figure 5. Cascade Trainer

The negative images required by the 'Cascade Trainer' are about 1,000 and the positive images require an appropriate amount of sample data. Therefore, a separate program was developed to secure a large amount of data. The program provides the following functions. It captures and saves the image in the area designed by the user while playing the video. User can specify the image in the red box as shown in the figure. And user can save the image by setting image in the designated area to positive or negative.

Figure 6. Separate Solution

The saved images area divided and saved in p and n folders. The positive image saved with program was 107 samples and the negative 146 samples were used for learning. Although it is less than the 1000 negative images required by the program. Considering the amount of data this examination has, it is an appropriate amount.

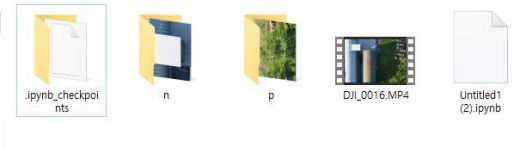

Figure 7. Separate Solution

At the end of training, check the result of training is necessary. Writing code for conduct the result of training. This examination uses the OpenCV with Python. The code means capture video, slice it with 0.05 second, convert to gray-image, detect tree using trained data, and show by rectangle image cover tree.

- Image processing

The trained 'Haar Cascade' model was used. To evaluate the performance of the model. The video used for evaluation in the model used 4K video. As you can see in the figure 8, the test method gets the images of the video one by one. A general RGB image is of a size that is difficult to process in a Raspberry Pi, so change the image to a black and white photo. The transformed image is used in the classification model. Arrays storing the coordinates and widths of recognized trees are returned. By using the coordinates recognized in this way, a box is marked on the image. The successive changes of these images are displayed as moving pictures.

```python
import os
import re
import time
import cv2
import numpy as np
from os.path import isfile, join

tree_classifier = cv2.CascadeClassifier('<Cascade_File_Path>')

cap = cv2.VideoCapture('<Video_File_Path>')

while True:
    time.sleep(.05)
    ret, frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    trees = tree_classifier.detectMultiScale(gray, 1.3, 5)
    for (x, y, w, h) in trees:
        image = cv2.rectangle(frame, (x, y), (x+w, y+h), (0,0,255), 2)
        cv2.imshow('Trees', image)
        #cv2.namedWindow('Trees', cv2.WINDOW_NORMAL) #optional
        #cv2.resizeWindow('Trees', 1900, 1000) #optional
        cv2.waitKey(1)

cap.release()
cv2.destroyAllWindows()
```

Figure 8. Separate Solution

C. *Real-time tree detection using UAV*

Real-time tree detection using UAV demonstration conducted with Raspberry Pi. The main idea is detecting from Raspberry Pi, store to SQL and the extraction and visualization.

Detecting from Raspberry is conducted with detecting model that trained with 'Haar cascade.' To find best result, this paper examines with four resolution videos.

Resolution	3840x2160	1920x1080	1280x720
Bit-rate	102132	160251	76133
Frame	29.97	29.97	29.97
Calculation time	478 second	133 second	71 second

Figure 5. Cascade Trainer

Unusually HD(1280x720) resolution has best and most accuracy. This progress can find amount of tree at specific latitude and longitude position.

- Raspberry Pi4 Environment Setting

Libraries including OpenCV 4.5.3, MariaDB 10.3.29, mysql 1.0.5 (SQL management), xrdp 0.9.9-1 (Remote control management), gpsd 3.17-7 (Connecting GPS module).

For store GPS and tree amount data to SQL, this paper use MariaDB. This database stores GPS and tree amount data. Insert GPS and tree amount data conducted with python code.

```python
if trees is True:
    data = ser.readline().decode('utf-8')
    while True:
        if data[0:6] == '$GPGGA':
            msg = pynmea2.parse(data)
            latval = msg.latitude
            longval = msg.longitude
            sql = "INSERT INTO TreeTable VALUES('" + latitude  + "','" + longitude + "','" + len(trees) +
            cur.execute(sql)
            conn.commit()
            conn.close()
            break
```

Figure 9. Insert GPS and tree amount data(code)

When Raspberry Pi detects a tree, 'Figure 6' calls GPS data. And the parsed longitude and latitude value will call SQL

saved to DB that we have defined in advance. Then, if each interval has a tree, it will insert a row containing the latitude and longitude trees.

```
MariaDB [(none)]> use project17;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [project17]> select * from TreeTable;
+----------+-----------+------------+
| latitude | longitude | TreeNumber |
+----------+-----------+------------+
|  35.2379 |   129.077 |          2 |
+----------+-----------+------------+
1 row in set (0.000 sec)
```

Figure 10. Insert GPS and tree amount data(result)

This process will stack the data in the MariaDB of Raspberry Pi. Then it is possible to access GPS and tree amount data remotely. Finally, extract valuable data from MariaDB and visualize this data with a map.

Raspberry Pi is a device to collect data. Using an external computer to output data would be better because it is easy to receive the results and more convenient to process.

We build up an internal network. First, we set up Raspberry Pi to connect to a fixed internal network. When this device collects data and returns to a base that has access to the internal network, the following actions begin:

```
#print from sql
conn = None
cur = None

latitude = []
longitude = []
Treenum = []

conn = pymysql.connect(host='192.168.0.6', user='root', password='project17', db='project17', charset='utf8')
cur = conn.cursor()

cur.execute("SELECT * FROM TreeTable")
df = pd.DataFrame(columns = ['latitude' , 'longitude', 'TreeNumber'])

while (True) :#  repeat
    row = cur.fetchone()# Enter a single line of cursor (table select) in row and move on to the next line
    if row== None :# If the cursor is no longer in value,
        break;--#out loop
    new_data = {
    'latitude' : row[0],
    'longitude' : row[1],
    'TreeNumber' : row[2]
    }
    df.loc[len(df)] = [row[0],row[1],row[2]]
```

Figure 11. Extract data remotely

External computers can access Raspberry Pi DB with fixed IP. However, external computers only have access to the DB but cannot modify the data, which prevents data contamination.

The external computer connects to the DB calling the data row by row from that table. We reconstruct the data in Dataframe format to easily manipulate this data. Now, we have data on latitude, longitude, and the number of trees.

```
import folium
from folium import Choropleth, Circle, Marker

#Create base map
m_4 = folium.Map(location=[35.2379088,129.076661666], tiles='cartodbpositron', zoom_start=13

def color_producer(val):
    if val >= 10:
        return 'forestgreen'
    elif val >=5:
        return 'limegreen'
    else:
        return 'greenyellow'

# Add a circle to the base map
for i in range(0,len(df)):
    Circle(
        location=[df.iloc[i]['latitude'], df.iloc[i]['longitude']],
        radius=1, color=color_producer(df.iloc[i]['TreeNumber'])).add_to(m_4)
# Display the map
m_4
```

Figure 12. Visualize data remotely (code)

We're going to visualize them. Python's representative visualization library, ‘folium’[4], will be used to mark the location of the tree. We create the base map and a function that determines the circle's color displayed on the map. The more trees converge, the deeper the green color is. Repeating marking practices by the length of the data frame, we draw circles with different green color tones depending on the number of trees.

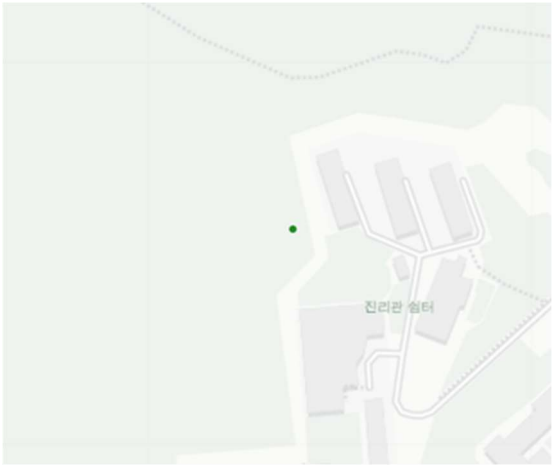

Figure 13. Visualize data remotely (result)

After all, data extraction, storage, and visualization have been completed.

In summary, we visualize the picture and video that a drone takes in a particular area. We can identify where trees are concentrated depending on the location and color of the circle stamp.

D. Conclusion

This paper examines tree detection and visualizing its distribution. Examination use ‘Haar Cascade’, Raspberry pi. ‘Haar Cascade’ uses for detecting tree and Raspberry pi uses for sending tree data remotely.

This paper can find the possibility of detecting tree and visualizing it to map. Some result shows that HD content has more accuracy than others. And because of the limitation of the Raspberry pi, the detect speed is slow. It means that ‘Haar Cascade’ is maybe not proper to 4K images. But if enough data and machine available, then this examination could improve and can uses for many other fields like tracking tree destroy of forest.

REFERENCES

[1] P. Volia and M. Jones. Rapid obejct detection using a boosted cascade of simple features, IEEE, 2003

[2] https://www.bgci.org/our-work/projects-and-case-studies/global-tree-assessment

[3] J. Canny. A Computational Approach to Edge Detection, IEEE, 1986

[4] http://python-visualization.github.io/folium/