DAFTAR PUSTAKA

Ade, V., 2019. Analisa Pengembangan Drone Penyemprotan Hama Tanaman Dengan Jenis Nosel Dan Ketinggian Untuk Mengetahui Luas Semprotan. Eng. J. Bid. Tek. 10, 63–69.

Afifah, L., 2020. Algoritma K-Nearest Neighbor (KNN) untuk Klasifikasi. IlmudataPy. URL https://ilmudatapy.com/algoritma-k-nearest-neighbor-knn-untuk-klasifikasi/ (accessed 7.1.22).

Ali Mustofa, S., 2013. Segmentasi Citra Digital Dengan Menggunakan Algoritma Watershed Dan Lowpass Filter Sebagai Proses Awal (Journal:eArticle). None. Brawijaya University.

Banda, F., 2020. OVERVIEW of IMAGE PROCESSING OVERVIEW OF IMAGE PROCESSING Contents. https://doi.org/10.13140/RG.2.2.23374.89929

Ghoneim, A., Gewaily, E., Osman, M., 2018. Effects of Nitrogen Levels on Growth, Yield And Nitrogen use Efficiency Of Some Newly Released Egyptian Rice Genotypes. Open Agric. 3, 310–318. https://doi.org/10.1515/opag-2018-0034

Guniganti, R.G., Ankam, S., n.d. A Comparision of RTMP and HTTP Protocols with respect to Packet Loss and Delay Variation based on QoE 68.

Herniwati -, Nappu, M.B., 2018. ANALISIS EFISIENSI PENGGUNAAN PUPUK NITROGEN PADA PADI SAWAH DI TANAH INCEPTISOLS. Inform. Pertan. 27, 119–127. https://doi.org/10.21082/ip.v27n2.2018.p119-127

Lei, T., Wang, Y., Fan, Y., Zhao, J., 2012. Vector morphological operators in HSV color space. Sci. China Inf. Sci. 56. https://doi.org/10.1007/s11432-011-4475-5

Mahesh, B., 2019. Machine Learning Algorithms -A Review. https://doi.org/10.21275/ART20203995

Manansala, J., 2021. Image Processing with Python: Image Segmentation using Thresholding Methods. The Startup. URL https://medium.com/swlh/image-processing-with-python-image-segmentation-using-thresholding-methods-423ecdaf8ab4 (accessed 6.29.22).

Pamungkas, A., 2017. Pengolahan Citra Digital. Pemrograman Matlab. URL https://pemrogramanmatlab.com/2017/07/26/pengolahan-citra-digital/ (accessed 6.29.22).

Rahmawati, E., n.d. Penggunaan Bagan Warna Daun (BWD) PadaTanamanPadi 2.

Rubesh, H., 2020. Interactive Color Image Segmentation using HSV Color Space. Sci. Technol. J. 7, 37–41. https://doi.org/10.22232/stj.2019.07.01.05

S, H.K., Dewatama, D., I, B.A., 2017. KLASIFIKASI WARNA DAUN GUNA OPTIMASI PEMUPUKAN NITROGEN/UREA PADA TANAMAN PADI. J. ELTEK 15, 1–17.

Sedo, R., Mudjirahardjo, P., Yudaningtyas, E., 2019. Identifikasi Takaran Pupuk Nitrogen Berdasarkan Tingkat Kehijauan Daun Tanaman Padi Menggunakan Metode Histogram of s-RGB dan Fuzzy Logic. J. EECCIS 13, 31–37.

Siallagan, J.O., Chalil, D., Jufri, M., 2013. Analisis Efisiensi Penggunaan Pupuk Bersubsidi pada Tanaman Padi Sawah (Studi Kasus: Desa Melati II, Kecamatan Perbaungan, Kabupaten Serdang Bedagai). J. Agric. Agribus. Socioecon. 2, 15041.

Singh, P., Singh, N., Singh, K.K., Singh, A., 2021. Chapter 5 - Diagnosing of disease using machine learning, in: Singh, K.K., Elhoseny, M., Singh, A., Elngar, A.A. (Eds.), Machine Learning and the Internet of Medical Things in Healthcare. Academic Press, pp. 89–111. https://doi.org/10.1016/B978-0-12-821229-5.00003-3

Sinwar, D., Kaushik, R., n.d. Study of Euclidean and Manhattan Distance Metrics using Simple K-Means Clustering 5.

Suwardiyasa, P., 2018. PENINGKATAN EFISIENSI PUPUK NITROGEN PADA PADI SAWAH DENGAN METODE BAGAN WARNA DAUN (BWD) | Dinas Pertanian [WWW Document]. URL https://distan.bulelengkab.go.id/informasi/detail/artikel/peningkatan-efisiensi-pupuk-nitrogen-pada-padi-sawah-dengan-metode-bagan-warna-daun-bwd-94 (accessed 6.28.22).

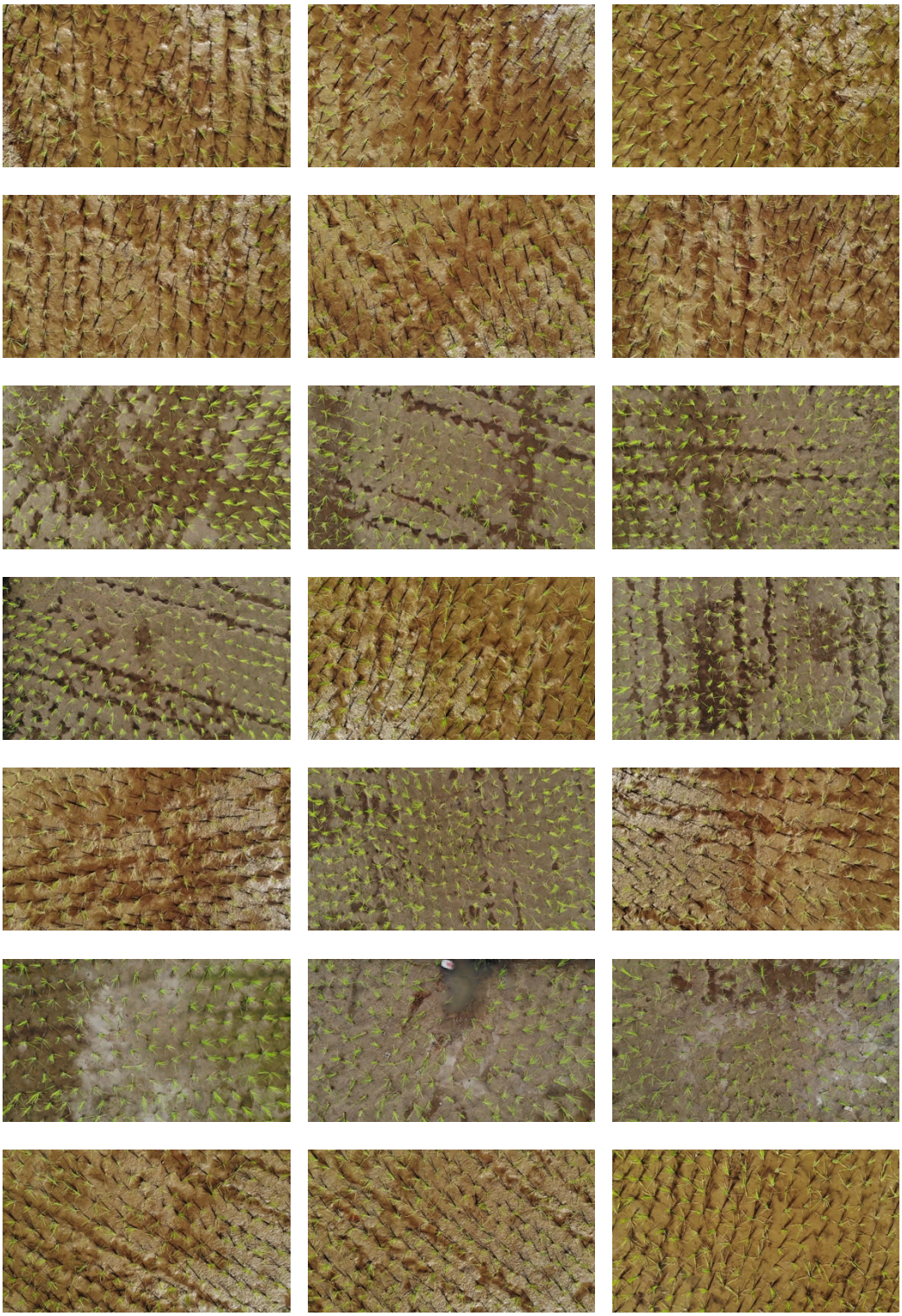
Tinning, G., n.d. Bagan Warna Daun (BWD) 19.

Triadiati, T., Pratama, A.A., Abdulrachman, S., 2012. Pertumbuhan dan Efisiensi Penggunaan Nitrogen pada Padi (Oryza sativa L.) Dengan Pemberian Pupuk Urea yang Berbeda. Bul. Anat. DAN Fisiol. Dh SELLULA 20, 1–14. https://doi.org/10.14710/baf.v20i2.4767

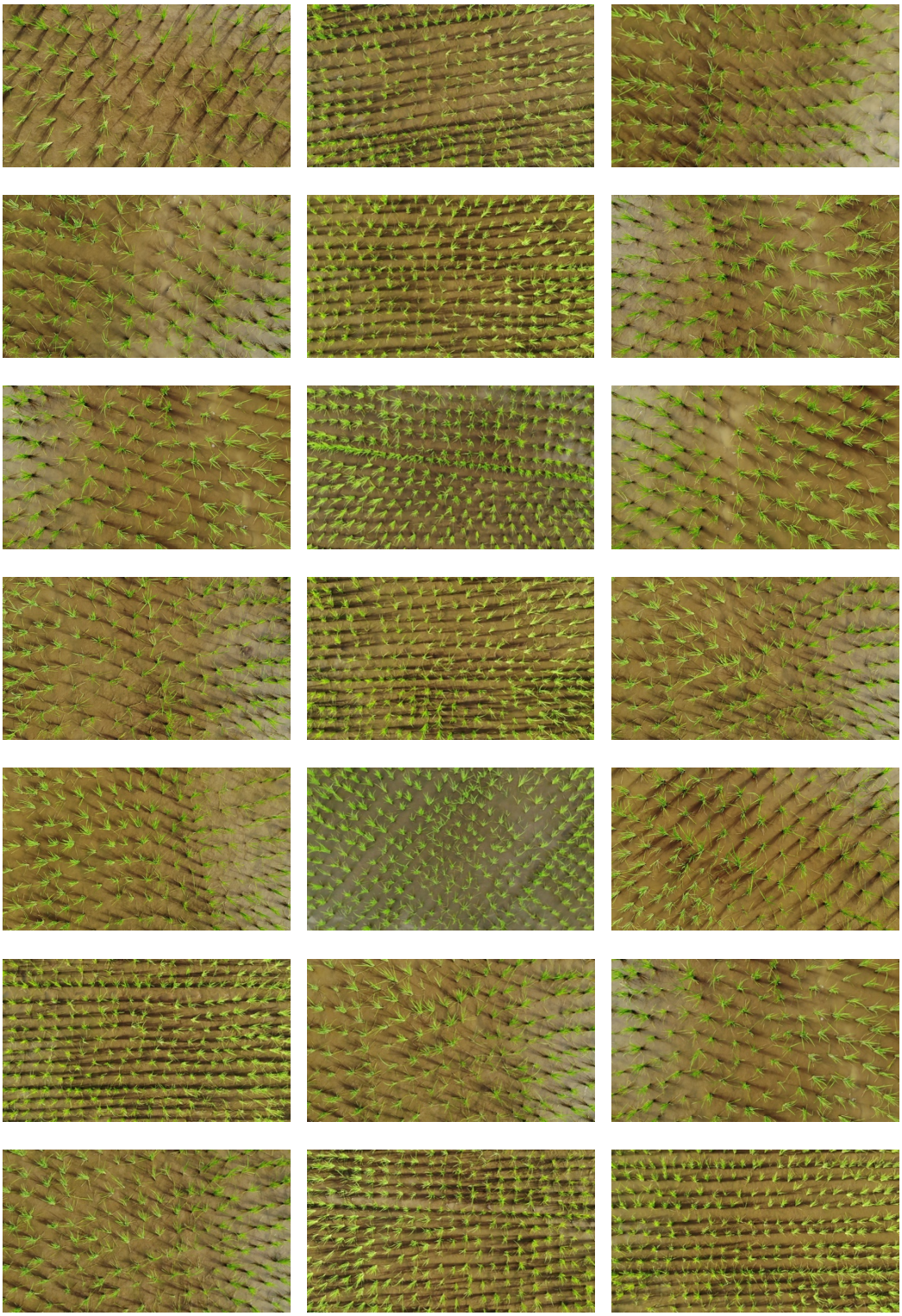
Wesolkowski, S.B., n.d. Color Image Edge Detection and Segmentation: A Comparison of the Vector Angle and the Euclidean Distance Color Similarity Measures 108.

LAMPIRAN

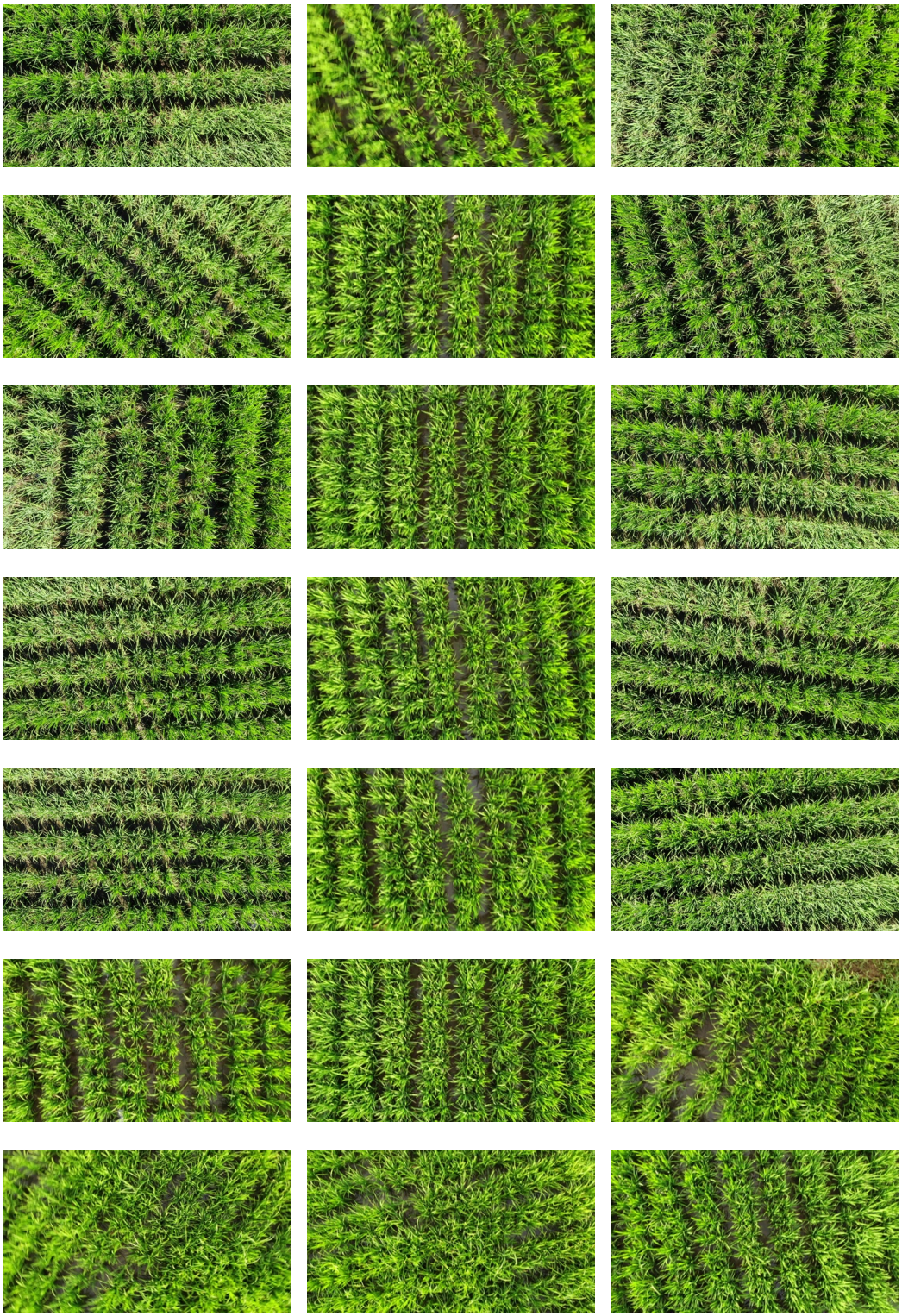
Lampiran 1: Contoh Data Tanaman Padi BWD Skala 2



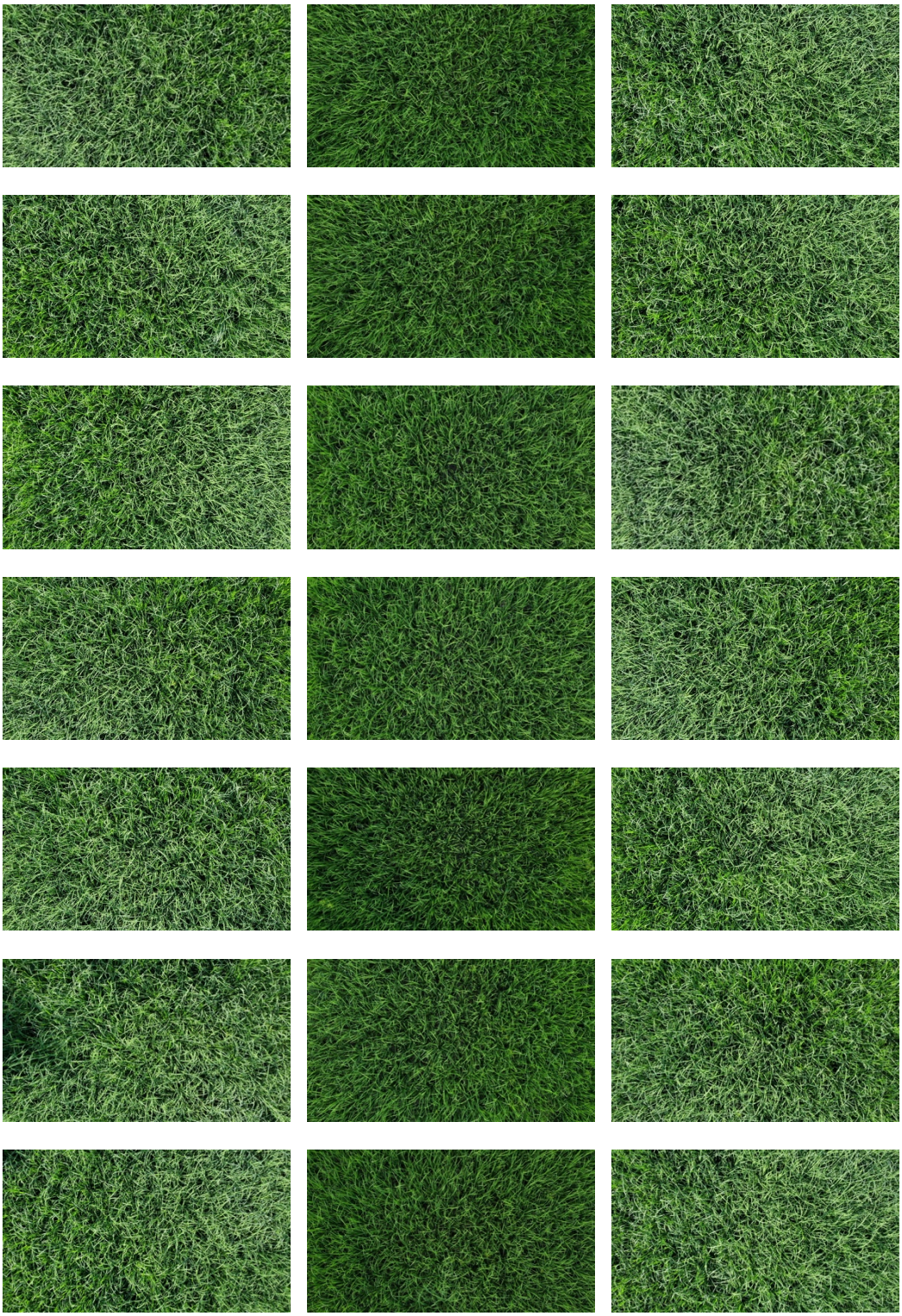
Lampiran 2: Contoh Data Tanaman Padi BWD Skala 3



Lampiran 3: Contoh Data Tanaman Padi BWD Skala 4



Lampiran 4: Contoh Data Tanaman Padi BWD Skala 5



Lampiran 5: *Source Code Preprocessing* dan Ekstraksi Fitur

```
# resize with imutils
def resize_image(image, new_width):
    return imutils.resize(image, width=new_width)

# resize all image with width = 500
images = [resize_image(image, 500) for image in images]

# convert rgb to hsv
hsv_images = [cv2.cvtColor(image, cv2.COLOR_BGR2HSV) for image in image

# hsv ranges for green colors
low_green = np.array([27, 100, 0])
high_green = np.array([85, 255, 255])

# threshold
green_masks = [cv2.inRange(hsv_image, low_green, high_green) for hsv_image in hsv_images]

greens = []
for i in range(len(images)):
    greens.append(cv2.bitwise_and(images[i], images[i], mask=green_masks[i]))


## BGR->RGB for display
rgb_images = [cv2.cvtColor(image, cv2.COLOR_BGR2RGB) for image in images]
rgb_greens = [cv2.cvtColor(green, cv2.COLOR_BGR2RGB) for green in greens]
rgb_green_masks = [cv2.cvtColor(green_mask, cv2.COLOR_BGR2RGB) for green_mask in green_masks]

---

def getMeanOfHSV(image):
    image_hsv = cv2.cvtColor(image, cv2.COLOR_RGB2HSV)
    image_hsv = image_hsv.reshape(-1, 3)
    image = image.reshape(-1, 3)
    grey = rgb2gray(image)
    non_black_pixels = len([i for i in grey if i != 0])
    mean_H = sum(image_hsv[:,0]/non_black_pixels)
    mean_S = sum(image_hsv[:,1]/non_black_pixels)
    mean_V = sum(image_hsv[:,2]/non_black_pixels)

    return numpy.array([mean_H, mean_S, mean_V], dtype='uint8')

def getModeOfHSV(image):
    image_hsv = cv2.cvtColor(image, cv2.COLOR_RGB2HSV)
    image_hsv = image_hsv.reshape(-1, 3)
    image = image.reshape(-1, 3)
    grey = rgb2gray(image)
    non_black_pixels = []

    for i in range(len(grey)-1):
        if grey[i] != 0:
            non_black_pixels.append(image_hsv[i])

    mode = stats.mode(non_black_pixels)

    return mode.mode.flatten()

# get mean of HSV of each image
mean_of_hsvs = [getMeanOfHSV(rgb_green) for rgb_green in rgb_greens]
mean_of_hsvs = numpy.array(mean_of_hsvs)

df.insert(4, 'hue', mean_of_hsvs[:,0])
df.insert(5, 'saturation', mean_of_hsvs[:,1])
df.insert(6, 'value', mean_of_hsvs[:,2])
df

# get mode of HSV of each image
mode_of_hsvs = [getModeOfHSV(rgb_green) for rgb_green in rgb_greens]
mode_of_hsvs = numpy.array(mode_of_hsvs)

df.insert(4, 'hue', mode_of_hsvs[:,0])
df.insert(5, 'saturation', mode_of_hsvs[:,1])
df.insert(6, 'value', mode_of_hsvs[:,2])
df

# save result as csv file
df.to_csv('mode_feature_extraction.csv', encoding='utf-8', index=False)
```

Lampiran 6: Contoh Hasil Ekstraksi Fitur Data Tiap Kelas

Indeks Data	Nilai Mean			Nilai Modus			Skala BWD	Indeks Data	Nilai Mean			Nilai Modus			Skala BWD
	H	S	V	H	S	V			H	S	V	H	S	V	
1	29	155	153	27	152	165	2	586	32	175	147	27	174	147	3
2	29	155	156	27	153	167	2	587	33	162	144	27	153	148	3
3	29	161	146	27	171	150	2	588	32	174	148	27	171	156	3
4	29	154	149	27	141	162	2	589	33	167	146	27	165	162	3
5	29	159	145	27	150	160	2	590	34	150	137	39	131	112	3
6	29	158	154	27	155	165	2	591	33	162	143	27	159	149	3
7	30	172	151	27	180	153	2	592	32	160	144	27	153	146	3
8	29	156	156	27	156	175	2	593	32	169	149	27	169	155	3
9	29	161	155	27	140	165	2	594	33	139	132	27	118	104	3
10	29	162	143	27	171	148	2	595	32	173	148	27	169	144	3
11	29	156	147	27	155	164	2	596	32	165	147	27	152	157	3
12	29	159	153	27	148	171	2	597	33	169	142	27	169	152	3
13	29	156	151	27	171	161	2	598	32	171	147	27	174	154	3
14	29	156	145	27	159	152	2	599	32	171	144	27	169	144	3
15	30	165	148	27	171	151	2	600	33	163	142	27	160	145	3
16	29	157	155	27	151	163	2	301	33	172	146	27	171	161	3
17	29	155	149	27	147	152	2	302	33	176	144	27	171	151	3
18	29	156	145	27	144	149	2	303	33	173	140	27	171	154	3
19	29	155	154	27	152	171	2	304	32	167	145	27	169	142	3
20	29	166	150	27	169	152	2	305	33	172	144	27	171	141	3
21	29	151	151	27	138	158	2	306	32	177	141	27	169	147	3
22	29	162	147	27	171	157	2	307	33	168	146	27	171	147	3
23	30	156	153	27	159	161	2	308	33	172	146	27	169	152	3
24	29	158	146	27	169	155	2	309	32	168	145	27	169	147	3
25	29	157	157	27	165	165	2	310	33	169	147	27	171	152	3
26	29	158	146	27	160	150	2	311	32	172	148	27	169	158	3
27	29	151	150	27	133	163	2	312	33	175	146	27	180	154	3
28	30	172	151	27	167	152	2	313	33	171	146	27	169	149	3
29	29	158	156	27	161	165	2	314	33	167	145	27	169	156	3
30	29	153	155	27	152	161	2	315	33	171	141	27	165	143	3
31	29	162	141	27	171	135	2	316	33	169	147	27	168	147	3
32	29	157	144	27	160	155	2	317	34	169	140	27	169	145	3
33	30	157	151	27	149	168	2	318	32	176	143	27	171	138	3
34	29	153	139	27	136	148	2	319	33	174	143	27	169	149	3
35	29	156	158	27	149	164	2	320	33	169	146	27	169	152	3

Index Data	Nilai Mean			Nilai Modus			Skala BWD	Index Data	Nilai Mean			Nilai Modus			Skala BWD
	H	S	V	H	S	V			H	S	V	H	S	V	
601	42	190	125	41	203	118	4	886	52	162	88	53	159	57	5
602	42	195	128	42	219	112	4	887	53	170	70	53	170	53	5
603	41	195	126	41	213	120	4	888	52	161	87	53	170	78	5
604	42	195	125	42	219	122	4	889	52	169	81	53	170	64	5
605	42	201	128	41	219	116	4	890	52	166	82	53	170	65	5
606	42	197	126	42	203	117	4	891	52	162	86	53	170	58	5
607	41	192	130	42	219	126	4	892	54	165	79	54	170	50	5
608	40	190	127	40	219	121	4	893	53	169	77	53	170	60	5
609	40	186	127	40	205	117	4	894	54	163	74	54	170	60	5
610	41	191	126	41	203	110	4	895	52	161	88	51	159	75	5
611	41	190	131	41	191	127	4	896	53	169	75	53	170	51	5
612	41	188	126	41	219	118	4	897	52	167	82	53	170	58	5
613	41	196	128	41	219	105	4	898	52	169	81	53	170	67	5
614	42	197	126	42	203	129	4	899	52	170	74	51	170	52	5
615	42	201	128	41	219	133	4	900	53	164	81	53	170	62	5
616	42	203	127	42	219	125	4	901	52	169	76	51	170	59	5
617	42	196	126	42	191	113	4	902	53	165	79	53	170	53	5
618	39	187	127	40	203	115	4	903	52	168	81	53	170	68	5
619	41	189	126	41	191	114	4	904	52	162	87	53	159	62	5
620	41	192	127	41	219	122	4	905	53	168	65	53	170	38	5
621	42	194	128	42	219	109	4	906	53	161	83	53	170	64	5
622	42	195	125	42	206	120	4	907	52	168	75	53	170	42	5
623	41	190	130	41	208	123	4	908	52	163	85	53	159	60	5
624	42	197	127	42	219	123	4	909	52	159	88	53	143	63	5
625	41	197	127	41	219	135	4	910	53	164	82	53	170	62	5
626	41	191	127	41	219	131	4	911	52	168	82	51	170	65	5
627	39	186	126	40	207	126	4	912	54	167	75	53	170	54	5
628	42	194	125	42	219	120	4	913	52	160	88	51	143	56	5
629	41	200	129	41	219	125	4	914	52	160	88	52	143	67	5
630	41	193	127	41	219	124	4	915	53	160	84	53	170	59	5
631	42	200	128	42	230	128	4	916	52	169	81	53	170	71	5
601	42	190	125	41	203	118	4	917	52	162	85	53	146	65	5
602	42	195	128	42	219	112	4	918	52	159	88	52	143	71	5
603	41	195	126	41	213	120	4	919	53	170	72	53	170	54	5
604	42	195	125	42	219	122	4	886	52	162	88	53	159	57	5
605	42	201	128	41	219	116	4	887	53	170	70	53	170	53	5

Lampiran 7: *Source Code* Proses Klasifikasi Menggunakan Algoritma KNN

```python
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
import numpy as np
import pandas as pd
from scipy.stats import mode
import matplotlib.pyplot as plt
import seaborn as sns; sns.set()


class KNN:
    def __init__(self, x, labels, k=1):
        self.X = x
        self.labels = labels
        self.k = k

    # Euclidean Distance
    def eucledian(self, p1, p2):
        distance = np.sqrt(sum((p1 - p2) ** 2))
        return distance

    # Manhattan Distance
    def manhattan(self, p1,p2):
        return sum(abs(value1 - value2) for value1, value2 in zip(p1, p2))

    # Function to get K nearest data point from input based on calculated distance
    def getKNearestNeighborsIndex(self, x):
        # calculate the eucledian distances of all the points to input data
        distances = np.array([self.manhattan(item, x) for item in self.X])
        # return k index of nearest distances
        return np.argsort(distances)[:self.k]

    # Function to calculate KNN
    def predict(self, x, k=None):
        self.k = k
        # array to store the prediction result
        op_labels = []

        # loop through the data points to be classified
        for item in x:
            # get index of k nearest data points (nearest neighbors) from input
            nearest_neighbors = self.getKNearestNeighborsIndex(item)
            # get labels of k data points
            labels = self.labels[nearest_neighbors]

            # get the most labels from that data points
            lab = mode(labels)
            lab = lab.mode[0]
            op_labels.append(lab)

        return op_labels

# Load dataset
data = pd.read_csv('../feature-extraction/mean_feature_extraction.csv', index_col=False)
data = data[data["image"].str.contains("siang")==False]

# Define the features and labels
x = data.drop(["image", "red", "green", "blue","bwd_level"], axis=1)
y = data.bwd_level

# Split the data
x_train, x_test, y_train, y_test = train_test_split(x.to_numpy(), y.to_numpy(), test_size=0.20, random_state=123)
x_train.shape, y_train.shape, x_test.shape, y_test.shape

# Train the model with different variant of K
k = []
testing_scores = []
training_scores = []

knn = KNN(x_train, y_train)

for i in range(1,51,2):
    k.append(i)
    pre = knn.predict(x_test, i)
    testing_score = accuracy_score(y_test, pre)
    testing_scores.append(testing_score * 100)
    training_score = accuracy_score(y_train, knn.predict(x_train, i))
    training_scores.append(training_score * 100)

result = pd.DataFrame({'k': k, 'training_score': training_scores, 'testing_score': testing_scores})
result

# Plot the accuration scores both training process and testing process
plt.figure(figsize=(20,10))
plt.ylim(bottom=95, top=100.1)
plt.plot(result.k, result.training_score, label = "Training Score", marker='o', linestyle='--', color='r',
markersize=12)
plt.plot(result.k, result.testing_score, label = "Testing Score", marker='o', linestyle='--', color='b', markersize=12)
plt.xticks(result.k)
plt.xticks(result.k, fontsize=19)
plt.yticks(fontsize=19)
plt.xlabel("Nilai K", fontsize=25, labelpad=10)
plt.ylabel("Skor dalam %", fontsize=25, labelpad=10)
plt.legend(fontsize=25)
plt.tight_layout()
plt.show()
```

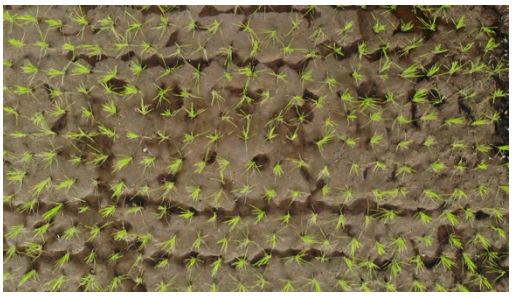
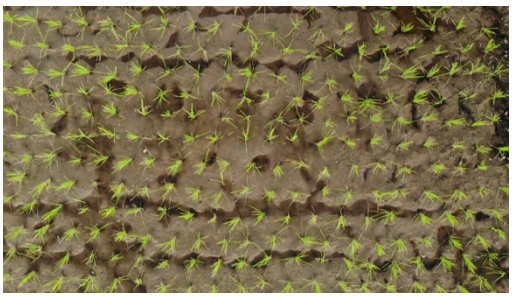
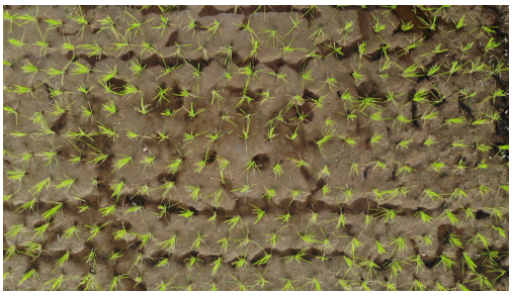
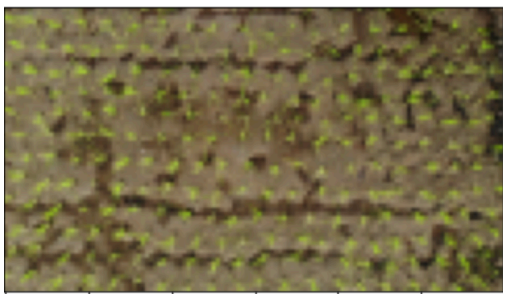
Lampiran 8: *Source Code* Sistem Identifikasi Nitrogen Berbasis *Real-Time*

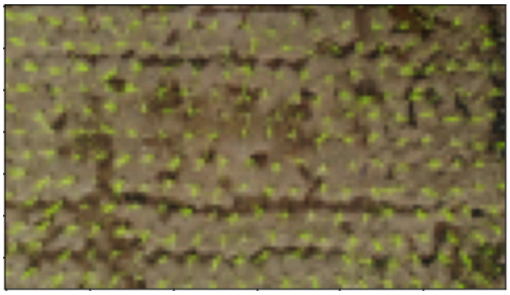
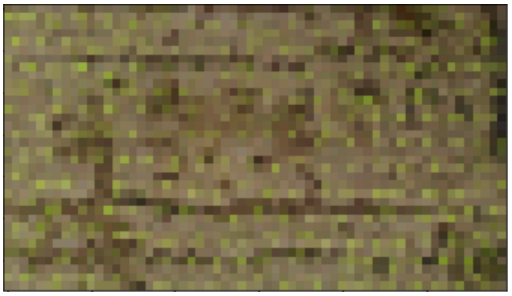
```python
import cv2
import numpy
from constant import model
from utils import Preprocessing, FeatureExtraction, NitrogenCalculator as Nc, writeText, drawColoredRectangle

if __name__ == "__main__":
    # Read video from RTMP stream
    cap = cv2.VideoCapture("rtmp://127.0.0.1/live")

    i = 0

    mean_of_hsv = numpy.zeros(3)

    while cap.isOpened():
        # Read frame from video
        ret, frame = cap.read()

        # Preprocess image
        ori, frame, mask = Preprocessing(frame).preprocess()

        if i % 30 == 0:
            mean_of_hsv = FeatureExtraction(frame).getMeanOfHSV()

        # Predict the nitrogen level
        pred = model.predict([mean_of_hsv])[0]

        # Show the result
        mask = cv2.cvtColor(mask, cv2.COLOR_GRAY2BGR)
        bp = numpy.zeros(mask.shape, dtype=numpy.uint8)
        bp = drawColoredRectangle(mean_of_hsv, bp)

        # Display text
        ori = writeText(ori, "Original Image")
        mask = writeText(mask, "Mask")
        frame = writeText(frame, "Preprocessing Result")
        bp = writeText(bp, f"Mean of HSV : {mean_of_hsv}", 100)
        bp = writeText(bp, f"BWD Level : {pred}", 60)
        bp = writeText(bp, f"Kebutuhan Nitrogen : {Nc(pred).getNitrogenDosageBasedOnBwdLevel()}", 20)

        # concatenate frames to be displayed
        horizontal_top = cv2.hconcat([ori, mask])
        horizontal_bottom = cv2.hconcat([frame, bp])
        vertical = cv2.vconcat([horizontal_top, horizontal_bottom])

        cv2.imshow('Leaf Chart Color Identification', vertical)
        i += 1
        if cv2.waitKey(10) == ord('q'):
            break

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

Lampiran 9: Hasil Percobaan Awal Terhadap Beberapa Algoritma Klasifikasi

Model	***Testing Score* (%)**	***Training Score* (%)**	***Prediction Time* (ms)**
KNN	99,35065	99,47539	0,003212
SVC	90,25974	93,46377	0,002093
Random Forest Classifier	99,35065	99,73855	0,012452

Lampiran 10: Tabel Perbandingan Waktu Pemrosesan Dan Akurasi Model Tiap Resolusi Citra Yang Diujikan

Resolusi (*pixel*)	***Processing Time* (*second*)**	**Akurasi Model**	**Citra Output**
1920x1080	0.843279	99,71%	
544x960	0.267311	99,71%	
283x500	0.090063	99,57%	
136x240	0.020790	98,62%	

68x120	0.006451	97,73%	
34x60	0.004705	97,28%	

LEMBAR PERBAIKAN SKRIPSI

"IDENTIFIKASI KEBUTUHAN NITROGEN PADA TANAMAN PADI MENGGUNAKAN CITRA DRONE DENGAN ALGORITMA KNN"

OLEH:

IRFAN RIPAT
D121171011

Skripsi ini telah dipertahankan pada Ujian Akhir Sarjana tanggal 7 September 2022.

Telah dilakukan perbaikan penulisan dan isi skripsi berdasarkan usulan dari penguji dan pembimbing skripsi.

Persetujuan perbaikan oleh tim penguji:

	Nama	Tanda Tangan
Ketua	Dr. Indrabayu, S.T., M.T., M.Bus.Sys.	
Sekretaris	Anugrayani Bustamin, S.T., M.T.	
Anggota	Prof. Dr. Ir. Syafruddin Syarif, M.T.	
	Muhammad Alief Fahdal Imran Oemar, ST., M.Sc.	

Persetujuan Perbaikan oleh pembimbing:

Pembimbing	Nama	Tanda Tangan
I	Dr. Indrabayu, S.T., M.T., M.Bus.Sys.	
II	Anugrayani Bustamin, S.T., M.T.	