Tabel 4.2 Kriteria Range Pengujian Buah Tomat

no	Jenis	Data Range RGB to HSV						Klasifikasi tomat	Kualitas tomat	Keterangan
		L-H	L - S	L-V	U-H	U-S	U-V			
1	Tomat 1	0	115	56	179	255	255	Matang	Baik	S3 ON P3
2	Tomat 2	0	115	56	179	255	255	Matang	Baik	S8 ON P6
3	Tomat 3	0	115	56	179	255	255	Matang	Baik	S1-10 OFF P9
4	Tomat 4	10	96	45	179	255	255	Setengah Matang	Baik	S4 ON P2
5	Tomat 5	10	96	45	179	255	255	Setengah Matang	Cukup baik	S7 ON P5
6	Tomat 6	10	96	45	179	255	255	Setengah Matang	Cukup baik	S10 ON P8
7	Tomat 7	27	99	47	179	255	255	Mentah	Baik	S3 ON P1
8	Tomat 8	27	99	47	179	255	255	Mentah	Baik	S6 ON P4
9	Tomat 9	27	99	47	179	255	255	Mentah	Baik	S9 ON P7

```
import numpy as np
import cv2
import serial
import struct

def nothing(x):
   pass
data_serial = serial.Serial('/dev/ttyS0',9600)

# Capturing video through webcam
webcam = cv2.VideoCapture(0)

# Start a while loop
while(1):

        # Reading the video from the
        # webcam in image frames
        _, imageFrame = webcam.read()

        # Convert the imageFrame in
        # BGR(RGB color space) to
        # HSV(hue-saturation-value)
        # color space
        hsvFrame = cv2.cvtColor(imageFrame, cv2.COLOR_BGR2HSV)

        # Set range for red color and
        # define mask
        red_lower = np.array([136, 87, 111], np.uint8)
        red_upper = np.array([180, 255, 255], np.uint8)
        red_mask = cv2.inRange(hsvFrame, red_lower, red_upper)
        print (“MATANG”)
        data_serial.write('M'.encode())

        # Set range for green color and
        # define mask
        green_lower = np.array([25, 52, 72], np.uint8)
        green_upper = np.array([102, 255, 255], np.uint8)
        green_mask = cv2.inRange(hsvFrame, green_lower, green_upper)
        print (“MENTAH”)
        data_serial.write('H'.encode())


        # Set range for yellow color and
        # define mask
```

```
yellow_lower = np.array([94, 80, 2], np.uint8)
yellow_upper = np.array([120, 255, 255], np.uint8)
yellow_mask = cv2.inRange(hsvFrame, yellow_lower, yellow_upper)
print (“SETENGAH MATANG”)
data_serial.write('K'.encode())


# Morphological Transform, Dilation
# for each color and bitwise_and operator
# between imageFrame and mask determines
# to detect only that particular color
kernel = np.ones((5, 5), "uint8")

# For red color
red_mask = cv2.dilate(red_mask, kernel)
res_red = cv2.bitwise_and(imageFrame, imageFrame,
                                                    mask = red_mask)

# For green color
green_mask = cv2.dilate(green_mask, kernel)
res_green = cv2.bitwise_and(imageFrame, imageFrame,
                                                        mask = green_mask)

# For yellow color
yellow_mask = cv2.dilate(yellow_mask, kernel)
res_yellow = cv2.bitwise_and(imageFrame, imageFrame,
                                                    mask = yellow_mask)

# Creating contour to track red color
contours, hierarchy = cv2.findContours(red_mask,

cv2.RETR_TREE,

cv2.CHAIN_APPROX_SIMPLE)

for pic, contour in enumerate(contours):
        area = cv2.contourArea(contour)
        if(area > 300):
                x, y, w, h = cv2.boundingRect(contour)
                imageFrame = cv2.rectangle(imageFrame, (x, y),
                                                                (x + w, y + h),
                                                                (0, 0, 255), 2)

                cv2.putText(imageFrame, "Red Colour", (x, y),
```

```
                                        cv2.FONT_HERSHEY_SIMPLEX, 1.0,
                                        (0, 0, 255))

# Creating contour to track green color
contours, hierarchy = cv2.findContours(green_mask,

cv2.RETR_TREE,

cv2.CHAIN_APPROX_SIMPLE)

for pic, contour in enumerate(contours):
        area = cv2.contourArea(contour)
        if(area > 300):
                x, y, w, h = cv2.boundingRect(contour)
                imageFrame = cv2.rectangle(imageFrame, (x, y),
                                                        (x + w, y + h),
                                                        (0, 255, 0), 2)

                cv2.putText(imageFrame, "Green Colour", (x, y),
                                        cv2.FONT_HERSHEY_SIMPLEX,
                                        1.0, (0, 255, 0))

# Creating contour to track yellow color
contours, hierarchy = cv2.findContours(yellow_mask,

cv2.RETR_TREE,

cv2.CHAIN_APPROX_SIMPLE)
for pic, contour in enumerate(contours):
        area = cv2.contourArea(contour)
        if(area > 300):
                x, y, w, h = cv2.boundingRect(contour)
                imageFrame = cv2.rectangle(imageFrame, (x, y),
                                                        (x + w, y + h),
                                                        (255, 0, 0), 2)

                cv2.putText(imageFrame, "Yellow Colour", (x, y),
                                        cv2.FONT_HERSHEY_SIMPLEX,
                                        1.0, (255, 0, 0))

# Program Termination
cv2.imshow("Multiple Color Detection in Real-TIme", imageFrame)
if cv2.waitKey(10) & 0xFF == ord('q'):
        cap.release()
```

```
cv2.destroyAllWindows()
break
```