

PERANCANGAN SISTEM DETEKSI WARNA UNTUK MEMBANTU ORANG BUTA WARNA BERBASIS MACHINE LEARNING MENGGUNAKAN TENSORFLOW

I Putu Ari Susila¹, Setia Juli Irzal Ismail S.T., M.T.², Gandeva Bayu Satrya, ST., MT., Ph.D.³

¹²³Prodi D3 Teknologi Komputer, Fakultas Ilmu Terapan, Universitas Telkom

¹arisusila@student.telkomuniversity.ac.id, ²jul@tass.telkomuniversity.ac.id, ³gbs@telkomuniversity.ac.id

Abstrak- Salah satu masalah dalam bidang kesehatan adalah penyakit buta warna, dimana penyakit ini merupakan dimana mata tidak bisa untuk membedakan jenis warna tertentu. Kasus penyandang buta warna di Indonesia setiap tahunnya semakin meningkat. Berdasarkan permasalahan tersebut maka dibuatlah sistem deteksi warna untuk membantu orang buta warna berbasis *machine learning* menggunakan tensorflow. Sistem ini dibuat dapat mendeteksi warna objek menggunakan tensorflow dan algoritma K-NN untuk klasifikasi warna, pengambilan objek menggunakan modul kamera raspi v2. Sistem ini dapat menampilkan akurasi deteksi objek dan juga mengeluarkan informasi warna berupa suara menggunakan speaker.

Abstract- One of the health problems is color blindness. This disease is the inability to determine certain types of colors. The number of people with color blindness in Indonesia is increasing every year. Based on this problem, a color detection system was created to help color blind people based on machine learning using tensorflow. This system is made to detect the color of objects using tensorflow and K-NN algorithm for color classification, object is captured using the Raspi v2 camera module. This system can measure the accuracy of object detection and able to inform the detected color information in the form of sound using speakers.

Keywords: Color object detection, Raspberry Pi, K-NN, Tensorflow, Machine learning.

1. Pendahuluan

1.1 Latar Belakang

Buta warna merupakan salah satu gangguan mata yang terjadi pada manusia. Buta warna sendiri adalah ketidak mampuan mata untuk membedakan warna-warna tertentu. Sebanyak 99% penderita buta warna tidak bisa membedakan antara warna hijau dan merah. Sebuah penelitian menyebutkan sebesar 8-12% lelaki di belahan bumi Eropa adalah pengidap buta warna[1]. Sementara persentase perempuan Eropa yang mengidap buta warna di perkirakan 0,5-1%(Rahmat Kurnia,2009)[2]. Di Indonesia sendiri penyandang buta warna setiap tahunnya semakin meningkat. Dari total penduduk yang berjumlah 255 juta jiwa, hampir sebanyak 0,7% memiliki kelainan genetika dalam membedakan tingkat gradasi suatu warna[3].

Memiliki kelainan seperti buta warna tentu menjadi suatu perkara yang sangat sulit dalam melakukan rutinitas sehari-hari. Seperti contoh dalam membedakan benda yang sesuai dengan warna yang diinginkan, atau bahkan dalam mengenali setiap warna objek yang digunakan dalam sehari hari seperti halnya, warna pakaian, warna benda rumah tangga dan lain sebagainya, tentu semua hal tersebut akan menjadi sulit bagi para penyandang buta warna.

Berdasarkan kasus tersebut, maka dirancang suatu sistem berbasis machine learning untuk membantu memudahkan penyandang buta warna untuk membedakan warna pada beberapa objek. Sistem ini dirancang menggunakan framework Tensorflow yang sudah terdapat library-library yang mendukung dalam pengembangan machine learning. Sistem ini dibangun dengan Raspberry Pi model 3 B yang terhubung dengan module kamera raspi v2 sebagai penangkap objek dan speaker sebagai keluaran suara. Sistem ini nantinya bekerja dengan memberikan informasi berupa suara dari setiap warna pada objek benda yang terdeteksi, yang memudahkan untuk membedakan warna dari setiap objek benda yang ada. Dengan

1.2 Rumusan Masalah

Berdasarkan uraian di atas, maka rumusan masalah dari Proyek Akhir ini adalah sebagai berikut :

1. Bagaimana cara membangun sebuah sistem yang dapat mendeteksi warna objek berbasis machine learning menggunakan tensorflow ?
2. Bagaimana cara membangun sistem berbasis machine learning menggunakan tensorflow yang memberikan informasi berupa suara?

1.3 Tujuan

Tujuan dari perancangan sistem ini antara lain yaitu:

1. Membangun sistem untuk mendeteksi warna objek berbasis *machine learning* menggunakan *tensorflow*.
2. Membangun sistem yang memberikan informasi warna berupa suara.

1.4 Batasan Masalah

Batasan masalah pada Proyek Akhir ini adalah

1. Program ini hanya mendeteksi jenis warna-warna merah, putih, hijau, biru, hitam, oranye dan kuning. Diluar dari warna tersebut tidak terdeteksi.
2. Program diuji hanya pada laptop dan raspberry Pi.
3. Objek pengujian hanya mengambil beberapa objek dari dataset pre-trained model `ssd_mobilenet_v1_coco` dan menggunakan `ssdlite_mobilenet_v2_coco` diantaranya botol, mangkuk, remote, gunting.
4. Output sistem hanya berupa suara dan text akurasi yang ditampilkan pada layar monitor.
5. Hanya diujikan pada tempat dengan cahaya terang dan cahaya redup
6. Pengujian hanya dilakukan pada jarak 20 cm, 30 cm, 40 cm, 50 cm, 60 cm.

1.5 Definisi Operasional

Definisi operasional dimaksudkan untuk menghindari perbedaan penafsiran yang berkaitan dengan istilah-istilah dalam judul penelitian yaitu “Sistem Deteksi Warna Untuk Membantu Orang Buta Warna Berbasis Machine Learning Menggunakan Tensorflow” maka definisi yang perlu dijelaskan yaitu:

a. Machine Learning

Machine Learning merupakan bagian dari AI(Artificial Intelligence) dimana machine dapat belajar sendiri tanpa perlu lagi di program eksplisit. [4]

b. Tensorflow

Tensorflow merupakan framework yang di produksi oleh google dimana framework ini bertujuan untuk memudahkan dalam pembelajaran machine learning.

c. Python

Bahasa pemrograman yang digunakan dalam pengembangan aplikasi ini menggunakan bahasa python. Python merupakan bahasa pemrograman yang mudah dipelajari, selain itu

library python juga mendukung dalam pengembangan machine learning.

2. Tinjauan Pustaka

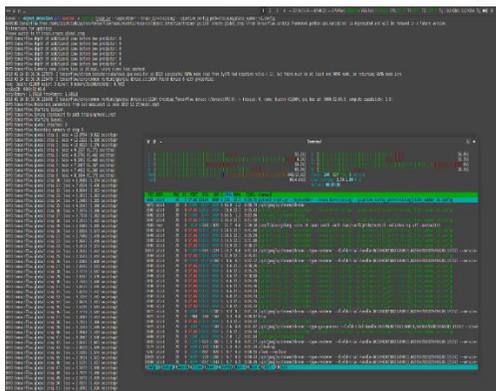
2.1 Penelitian Sebelumnya

Berdasarkan penelitian yang dilakukan oleh Syarifah Rosita Dewi yang berjudul Deep Learning Object Detection Pada Video Menggunakan Tensorflow Dan Convolutional Neural Network. Penelitian ini dibuat sebuah sistem untuk mendeteksi antara meja dan kursi motif ukiran Jepara pada suatu gambar dan video.[5] dan Penelitian oleh Siti Zahrina Jasmine yang berjudul Rancang Bangun Alat Pendeteksi Warna Pada Kaleng Minuman Menggunakan Sensor TCS3200 Berbasis Arduino Uno. Sistem dibangun berbasis arduino dengan sensor TCS3200 sebagai alat untuk deteksi warna pada kaleng minuman, untuk kasus ini kaleng yang diujikan antaranya kaleng fanta, pocari sweat dan milo.[6]

2.2 Teori

2.2.1 Aplikasi

Aplikasi adalah perangkat lunak atau *software* yang di buat untuk mengerjakan suatu pekerjaan tertentu yang bertujuan untuk memudahkan para pengguna. Jadi aplikasi merupakan *tools* untuk membuat perubahan dari permasalahan sulit menjadi lebih sederhana lagi [7]. Menurut Nazrudin Safaat H (2012 : 9) Perangkat lunak aplikasi adalah suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna[8].



Gambar2. 1 Program Aplikasi

2.2.2 Machine Learning

Machine learning atau bisa disebut juga dengan pembelajaran mesin merupakan bagian dari AI atau yang dikenal dengan (*Artificial Intelligence*). Dimana *machine learning* sendiri digunakan untuk menggantikan atau menirukan perilaku dari manusia untuk menyelesaikan sebuah permasalahan. Ciri tertentu dari mesin learning adalah dimana adanya proses pelatihan, pembelajaran, atau *training*. ML (*Machine Learning*) memiliki metode yaitu klasifikasi yang digunakan untuk memilah atau mengklasifikasi objek berdasarkan ciri tertentu sama seperti halnya manusia mencoba untuk membedakan benda satu dengan yang lainnya. [9]

2.2.3 Tensorflow

Tensorflow merupakan platform yang bertujuan untuk pembelajaran machine learning. Tensorflow sendiri sudah memiliki perpustakaan atau sumber daya komunitas komprehensif yang fleksibel dimana memudahkan di dalam pengembangan dan pembangunan aplikasi berbasis ML (*Machine Learning*).[10]

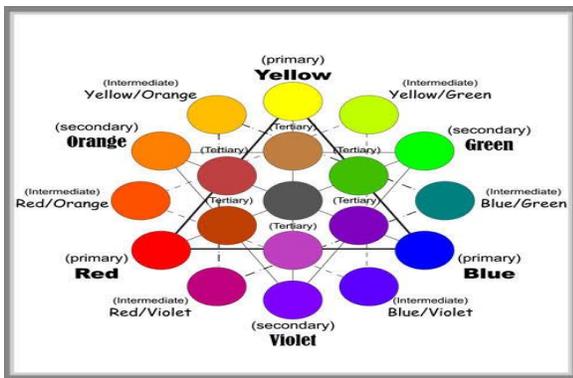
2.2.4 Dataset

Dataset merupakan sekumpulan objek yang nantinya akan di *training* atau dilatih, yang bertujuan untuk pelatihan *Neural Network*, sehingga data dapat dikenali[11]. Semakin banyak data yang akan di training maka akan semakin bagus kerja dari *Neural Network* tersebut. Tujuan dari dataset ini adalah untuk memudahkan *machine learning* dalam pengenalan objek yang dimaksud. Untuk pengujian pada sistem digunakan 2 data training, untuk data training pada objek yang berbentuk graph dari *ssd_mobilenet* dan juga data training dengan color histogram sehingga

diperoleh nilai RGB pada gambar warna yang dilatih.

2.2.5 Warna

Warna merupakan indikator pembeda antara objek satu dengan yang lainnya[12]. Dalam warna ada yang disebut dengan citra warna. Citra warna merupakan citra digital yang setiap pixel berisikan informasi warna. Informasi warna ini dibentuk oleh 3 sampel warna dasar. Warna dasar yang digunakan adalah RGB(*Red-Green-Blue*)[13]. Dari tiga warna dapatlah diperoleh jenis warna yang berda dari campuran ketiga warna dasar tersebut.



Gambar 2. 2 Macam-macam warna

2.2.6 K-NN (K-Nearest Neighbor)

K-Nearest Neighbor adalah salah satu metode algoritma yang digunakan dalam pengklasifikasian. Dalam prinsip kerjanya, *K-Nearest Neighbor* mencari jarak terdekat antara data yang akan dievaluasi dengan K tetangga (*neighbor*) terdekat dalam data pelatihan. Metode ini nantinya digunakan untuk menghitung jarak terdekat data yang akan dievaluasi dan pelatihan untuk diklasifikasi jenis warna yang ada. Penggunaan k-NN dipilih sebagai pengklasifikasian dikarenakan k-NN tangguh terhadap data training yang memiliki

banyak *noise*, dibandingkan dengan metode klasifikasi lain k-NN dapat menggeneralisasi himpunan data training yang relatif kecil dan selain itu k-NN memberikan hasil rasio kesalahan yang kecil untuk prediksi[14]. Adapun rumus pencarian jarak menggunakan rumus *Euclidian*[15]. dapat

$$d_i \sqrt{\sum_{i=1}^p (x_{2i} - x_{1i})^2}$$

Dengan :

X1 = sampel data

X2 = data uji

i = Variabel data

dist = jarak

p = dimensi data

2.2.7 Raspberry Pi 3 Model B

Raspberry Pi adalah sebuah SBC (*Single Board Computer*) dengan *wireless LAN* and *Bluetooth connectivity*. *Raspberry Pi 3 Model B* adalah generasi ke 3 dan model paling awal dari generasi tersebut. *Raspberry Pi 3* menggantikan *Raspberry Pi 2 Model B* [16]. Spesifikasi *Raspberry Pi 3 Model B*, dapat dilihat pada



Tabel 2.2

Gambar2. 3 Raspberry Pi Model B

Tabel 2. 1 Spesifikasi Raspberry Pi 3 Model B

NO	Spesifikasi
1	Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
2	1GB RAM
3	BCM43438 wireless LAN dan Bluetooth Low Energy (BLE) on board
4	100 Base Ethernet
5	40-pin extended GPIO
6	4 port usb, 1 port Ethernet, 1 port HDMI, 1 port audio jack, 1 port Camera Serial interface (CSI), 1 port Display Serial Interface (DSI)
7	4 Pole stereo output and composite video port
8	Full size HDMI
9	CSI camera port untuk koneksikan Raspberry Pi camera
10	DSI display port untuk koneksikan Raspberry Pi touchscreen display
NO	Spesifikasi
11	Micro SD port untuk pemuatan system operasi dan penyimpanan data
12	Upgraded switched Micro USB power source up to 2.5A

2.2.8 Raspberry Pi Camera V2

Module Kamera Raspberry Pi bertujuan sebagai pengambil gambar dalam bentuk video, yang dilengkapi dengan resolusi kamera sebesar 8 Megapixels. Memudahkan di dalam pengambilan video dengan kualitas yang baik [17]. Spesifikasi module kamera dapat dilihat pada Tabel 2.3

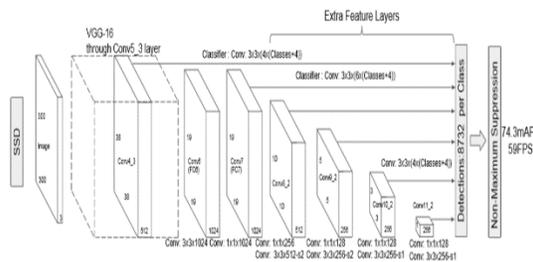


Gambar2. 4 Raspberry Pi Camera V2

No	Nama	spesifikasi
1	Berat	3g
2	Resolusi kamera	8 Megapixels
3	Mode video	1080p30, 720p60 and 640 x 480p60/90
4	Integrasi Linux	V4L2 driver available
5	C programming API	OpenMAX IL and others available
6	Sensor	Sony IMX219
7	Resolusi sensor	3280 x 2464 pixels
8	Area gambar sensor	3.68 x 2.76 mm (4.6 mm diagonal)
9	Ukuran pixel	1.12 µm x 1.12 µm
10	Ukuran optikal	1/4"
11	Focal length	3.04 mm
12	Bidang tampilan horizontal	62.2 degrees
13	Bidang tampilan vertical	48.8 degrees
14	Focal ratio (F-Stop)	2.0

2.2.9 Single Shot MultiBox Detector (SSD)

SSD merupakan salah satu metode untuk deteksi objek, Model pendekatan ssd di dasari pada jaringan *convolutional* yang menghasilkan koleksi kotak pembatas berukuran tetap untuk klasifikasikan objek dan skor untuk *instance* kelas objek dalam kotak kotak tersebut. SSD menggunakan fitur multi skala untuk deteksi sehingga meningkatkan akurasi deteksi, yang



memungkinkan prediksi deteksi pada berbagai skala. Gambar di bawah merupakan proses pembuatan model dengan melakukan pendekatan jaringan *convolutional* [18].

Gambar2. 5 Model SSD

2.2.10 Supervised Learning

Supervised learning bertujuan untuk menentukan pola baru yang dihubungkan pada pola data yang sudah ada dengan data yang baru [19]. *Supervised* bisa juga diartikan sebagai label ditiap datanya. Label sendiri merupakan output, label yang dimaksud adalah tag dari data yang ditambahkan dalam *machine learning* model. Seperti contoh di tag “mangkuk” ditiap masing masing gambar mangkuk dan gambar botol di tag “botol” pada masing gambar botol [20]. Label yang digunakan yaitu tag berdasarkan objek dan warna, contoh pada

gunting di tag “gunting” pada setiap masing masing gambar gunting, sedangkan pada warna tag “merah” pada masing masing gambar warna merah.

2.2.11 Perhitungan Score Akurasi

Actual Class	Predicted Class	
Negati	Negative TN	Positive FP
Positive	FN	TP

Akurasi sendiri adalah salah satu cara sistem mengukur objek dengan benar dari banyaknya data[21]. Pada proses pengklasifikasian terdapat yang namanya *True Negative* (TN), *True Positives* (TP), *False Negative* (FN), *False Positives* (FP). Ilustrasi dari empat situasi dan 2 kondisi yaitu Positive dan Negative dapat dilihat pada Tabel2.4

Tabel 2. 2 Empat Situasi dua kondisi

Untuk menghitung nilai akurasi yang diperoleh dengan rumus perhitungan sebagai berikut:

$$Akurasi = \frac{TP + TN}{TP+TN+FP+FN}$$

Untuk menentukan perhitungan disini semisal diperoleh data sebanyak 100.000 data gambar botol, diperoleh TN sebanyak 97.750, FP sebanyak 150, FN sebanyak 330 dan TP sebanyak 1.770 maka akurasi yang di peroleh dengan rumus tersebut adalah 0,9952 atau jika dikalikan dengan 100 adalah 99,52%.

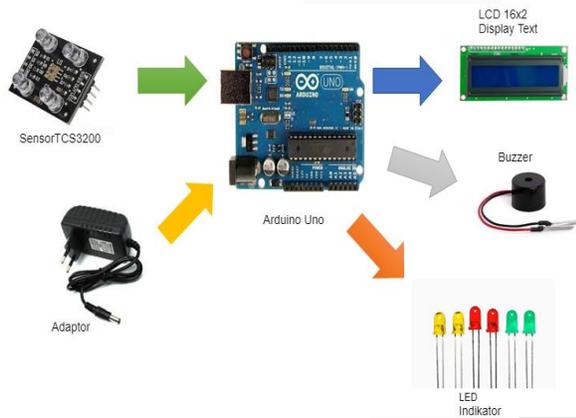
$$Akurasi = \frac{97,750 + 1,770}{100,000} = 0.9952$$

3. Analisis dan Perancangan

3.1 Analisis

3.1.1 Gambaran Sistem Saat ini

Adapun gambaran sistem saat ini dalam pembahasan Proyek Akhir adalah sebagai berikut:



Pada Gambar 3.1 diatas merupakan gambaran sistem yang ada saat ini. Sistem dibangun berbasis arduino dengan sensor TCS3200 sebagai alat untuk deteksi warna pada kaleng minuman, untuk kasus ini kaleng yang diujikan antaranya kaleng fanta, pocari sweat dan milo. Cara kerja sistem ini adalah kaleng didekatkan pada sensor TCS3200, kemudian nilai yang terbaca yaitu RGB (Red, Green, Blue) pada serial monitor. Jika kaleng fanta yang

didekatkan maka hasil yang keluar pada LCD adalah merah, begitu juga untuk jenis kaleng lainnya hasilnya akan sesuai dengan warna pada kaleng dan hasil warna tertampil pada LCD. Namun jika kaleng tidak terdeteksi sensor,

No	Kebutuhan fungsional
1	Mendeteksi Objek Benda
2	Mendeteksi Objek warna
3	Menampilkan nama objek dan akurasi yang diperoleh
4	Mengaktifkan suara informasi warna

hasil tidak tertampil pada LCD[6].

3.1.1 Analisis Kebutuhan Sistem

Adapun analisa kebutuhan fungsional dapat dilihat pada Tabel 3.1 dan non fungsional dapat dilihat pada Tabel 3.2

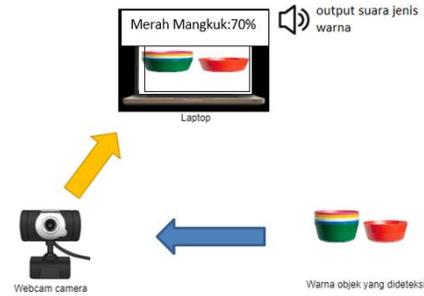
Tabel 3. 1 Kebutuhan Fungsional

Tabel 3. 2 Kebutuhan Non Fungsional

No	Kebutuhan Non Fungsional
1	Software Tensorflow, software open source untuk pengembangan <i>machine learning</i> yang mendukung.
2	Dataset yaitu kumpulan data-data dari jenis warna yang ada, yang akan di pelajari oleh program.
3	Software Notepad++ yaitu text editor untuk pengembangan program yang

	akan di bangun.
4	Dibutuhkan satu buah Laptop yang digunakan untuk menjalankan program.
5	Dibutuhkan satu Raspberry Pi untuk menjalankan program.
6	Dibutuhkan satu Raspberry Pi kamera untuk menangkap objek

3. Output yang akan dikeluarkan adalah berupa tulisan jenis warna dan nama objek yang akan muncul pada layar laptop. Dan juga akan muncul pemberitahuan berupa suara dari warna objek tersebut.



3.2 Perancangan

3.2.1 Gambaran Sistem Usulan

Perancangan sistem ini menggunakan dua hardware yang berbeda yaitu menggunakan



Laptop dan Raspberry Pi sebagai pengujian.

Gambar3. 1 Perancangan Sistem

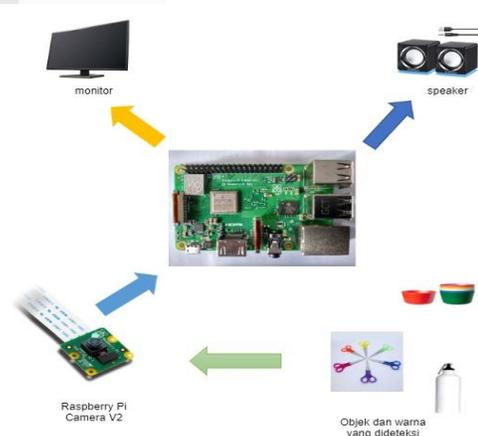
1. Input berasal dari objek yang akan di deteksi nantinya, untuk menangkap objek menggunakan bantuan dari webcam, nantinya objek akan dibandingkan dengan dataset model `ssd_mobilenet_v1_coco`.

2. Proses terjadi pada sistem adalah jika ada objek sesuai dengan dataset model `ssd_mobilenet_v1_coco` ditangkap oleh webcam maka nantinya warna pada objek akan dibandingkan dengan warna yang terdapat dalam dataset yang telah di *training* (data yang telah di pelajari sistem).

Gambar3. 2 Sistem Menggunakan Laptop

Penjelasan singkat Gambar 3. 3 Blok Diagram Sistem diatas:

1. Webcam digunakan untuk menangkap objek video dan di teruskan ke laptop.
2. Laptop digunakan untuk menjalankan program yang telah dibuat serta memberikan hasil/*output* berupa suara, nama objek, dan akurasi deteksi pada objek.



Gambar3. 3 Sistem menggunakan Raspberry Pi 3 B

Penjelasan singkat Gambar 3. 4 Blok Diagram Sistem diatas:

1. Raspberry Pi Camera digunakan untuk pengambilan objek benda dan di teruskan ke Raspberry Pi.
2. Raspberry Pi akan jalankan program untuk menangkap setiap frame video dengan OpenCv kemudian akan dibandingkan dengan dataset objek dan dataset warna yang tersedia
3. Monitor berfungsi sebagai penampil nama objek , jenis warna, dan akurasi deteksi objek yang di tangkap dengan Pi camera.



4. Speaker mengeluarkan hasil berupa suara dari warna objek yang dideteksi.

3.2.2 Cara Kerja Sistem

Gambar3. 4 flowchart Sistem menggunakan laptop dan Raspberry Pi

Penjelasan singkat Gambar 3.5 flowchart sistem sebagai berikut:

1. Start untuk memulai sistem yaitu mendeteksi objek dengan bantuan Tensorflow Objek Deteksi Api.
2. Mendeteksi warna dan membandingkan warna objek dengan warna pada dataset.
3. Apabila warna dan objek sesuai dengan pada dataset maka sistem akan memberikan keluaran berupa suara dan text jenis warna dan jenis objek yang ditampilkan pada monitor dan suara pada speaker.

3.2.3 Spesifikasi Sistem Hardware dan Software

Untuk spesifikasi sistem untuk Proyek Akhir dapat di lihat pada Tabel 3.3 untuk spesifikasi hardware dan Tabel 3.4 untuk spesifikasi software di bawah ini:

Tabel 3. 3 Spesifikasi Hardware

NO	Hardware	Spesifikasi/fungsi
1	Laptop Lenovo Y520	Intel Core i7 Kaby-Lake 15,6 inci IPS FHD (1920 x 1080) RAM 8GB NVIDIA GeForce GTX 1050/TI HDD 1TB
2	Raspberry Pi 3 Model B	Quad Core 1.2GHz Broadcom BCM2837 64bit CPU 1GB RAM BCM43438 wireless LAN dan Bluetooth Low Energy (BLE) on board 100 Base Ethernet 40-pin extended GPIO 4 USB 2 ports 4 Pole stereo output dan composite video port Full size HDMI CSI camera port for connecting a Raspberry Pi camera DSI display port for connecting a Raspberry Pi touchscreen display Micro SD port for loading your operating system and storing data Upgraded switched Micro USB power source up to 2.5A. Berfungsi untuk menjalankan program yang telah di buat.

No	Hardware	Spesifikasi/fungsi
3	Raspberry Pi Camera v2	Berat : 3g Resolusi : 8 Megapixels Mode video: 1080p30, 720p60 and 640 x 480p60/90 Linux integration : V4L2 driver available C programming API : OpenMAX IL and others available Sensor : Sony IMX219 Resolusi sensor : 3280 x 2464 pixels Sensor area gambar : 3.68 x 2.76 mm (4.6 mm diagonal) Ukuran piksel : 1.12 µm x 1.12 µm Ukuran optik: 1/4" Focal length : 3.04 mm Horizontal field of view : 62.2 degrees Vertical field of view : 48.8 degrees Focal ratio (F-Stop) : 2.0 Berfungsi untuk penangkapan objek melalui pi camera

Tabel 3. 4 Spesifikasi Software

No	Software	Spesifikasi/fungsi
1	Notepad++	Version: 7.8.5 Sebagai text editor penulisan program
2	Tensorflow	Version: 14.1.1 Digunakan sebagai framework untuk pengembangan sistem
4	OpenCv	Version : 4.3.0 Digunakan untuk membaca frame-frame video.
5	Spyder	Version: 4.0.1 Digunakan sebagai text editor selain itu memudahkan untuk debugging program

3.4 Rancangan Pengujian

3.4.1 Deteksi Warna

Adapun rencana pengujian terhadap deteksi warna, rencana pengujian sebagai berikut:

1. Pengujian hanya 1 warna pada beberapa objek

Pengujian 1 warna bertujuan untuk pengujian sistem apakah berhasil mendeteksi warna tersebut.

2. Pengujian beberapa warna pada beberapa objek

Pengujian ini bertujuan untuk pengujian sistem apakah berhasil mendeteksi lebih dari 1 warna.

3.4.2 Rentangan Deteksi

Adapun rencana pengujian rentangan terhadap deteksi warna pada objek, rencana pengujian sebagai berikut :

1. Pengujian dilakukan mulai pada jarak 20cm sampai dengan 60cm

Pengujian ini bertujuan untuk mengetahui Batasan deteksi warna pada objek benda yang diujicobakan.

4. Implementasi dan Pengujian

4.1 Langkah Pengerjaan

Pada bagian implementasi proyek akhir ini langkah yang dilakukan adalah sebagai berikut:

4.1.1 Mencari Referensi

Tahapan pertama yaitu mencari referensi yang di butuhkan untuk pengembangan sistem pendeteksi warna. Dimulai dengan mempelajari

mengenai framework tensorflow, cara memperoleh nilai pixel RGB atau menentukan fitur ekstraksi pada setiap warna agar bisa dilakukan klasifikasi, cara mengklasifikasikan warna menggunakan algoritma K-NN. Salah satu referensi proyek akhir diperoleh melalui github [22]

biru	4/30/2020 11:31 AM	File folder
hijau	4/30/2020 11:34 AM	File folder
hitam	4/30/2020 11:29 AM	File folder
kuning	4/30/2020 11:42 AM	File folder
merah	4/30/2020 11:38 AM	File folder
oranye	4/30/2020 11:37 AM	File folder
putih	5/17/2020 1:48 PM	File folder

Gambar 4. 1 Klasifikasi Warna

4.1.2 Pengumpulan Sampel Warna

Pengumpulan beberapa sampel warna yang nanti akan di training, warna dibagi menjadi tujuh warna dengan jumlah sampel warna pada setiap warna dapat di lihat pada tabel 4.1 di bawah dan Gambar 4.1 :

Tabel 4. 1 Pengumpulan Sampel Warna

No	Wana	Jumlah warna
1	Hitam	11 sampel warna
2	Putih	10 sampel warna
3	Biru	10 sampel warna
4	Hijau	11 sampel warna
5	Kuning	10 sampel warna
6	Merah	12 sampel warna
7	Oreanye	10 sampel warna

4.1.3 Pembuatan data pelatihan (training data)

Pada proses ini sampel warna di training menggunakan *color histogram* dimana bertujuan

```

with open("training.data", "a") as myfile:
    myfile.write(feature_data + "," + data_source + "\n")

for i in range(0, 255):
    for j in range(0, 255):
        for k in range(0, 255):
            image = cv2.imread(img_name)
            if warna in namawarna then
                datasource = warna
            colors = ("b", "g", "r")
            features = []
            feature_data = ""
            counter = 0

            for (split1, color) in zip(split, colors)do
                counter = counter + 1

                hist = cv2.calcHist([split1], [0], None, [256], [0, 256])
                features.extend(hist)

                # find the peak pixel values for R, G, and B
                elem = np.argmax(hist)

                if (counter == 1)then
                    blue = str(elem)
                elif (counter ==2)then
                    green = str(elem)
                elif (counter ==3)then
                    red = str(elem)
                    feature_data = red + "," + green + "," + blue
                endif
            endfor
    endfor
    
```

untuk mendapatkan nilai pixel RGB dengan rentangan 0 sampai dengan 255. Untuk psodocode dapat di lihat pada Gambar 4.2 dan hasil training pada Gambar 4.3:

Gambar 4. 2 psodocode pembuatan training data

Gambar 4. 3 Hasil training data

4.1.4 Melakukan Deteksi Objek

Tahapan ini melakukan deteksi objek dengan bantuan tensorflow *Object Detection Api* untuk mendeteksi objek benda secara *real time*. Dengan dataset *pre-trained coco* dengan model *SSD_Mobilenet_v1* dan *SSDlite_mobilet_v2*. Tahapan ini objek akan diklasifikasi untuk menentukan jenis dari objek yang di deteksi.

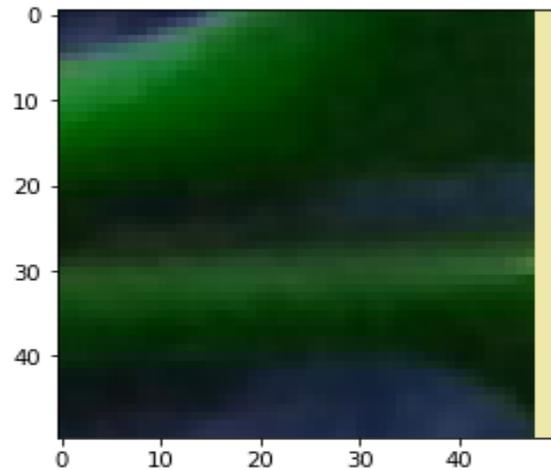
4.1.5 Melakukan crop frame gambar menjadi 50X50

Tahapan ini melakukan crop gambar dari hasil objek yang di deteksi dengan tensorflow api yang bertujuan untuk mendapatkan gambar

```
Myfunction(img,cropx,cropy)
y,x,channels = img.shape
startx = x//2-(cropx//2)
starty = y//2-(cropy//2)

return img[starty:starty+cropy,startx:startx+cropx]
```

Gambar 4. 4 pseudocode crop data



Gambar 4. 5 Hasil crop data

1	61,13,3,merah
2	254,0,2,merah
3	236,34,50,merah
4	125,4,9,merah
5	254,0,0,merah
6	206,0,25,merah
7	209,23,23,merah
8	204,22,0,merah
9	220,0,3,merah
10	139,0,0,merah
11	174,32,26,merah
12	128,24,24,merah
13	255,242,39,kuning
14	255,215,12,kuning
15	252,234,4,kuning
16	254,242,0,kuning
17	247,224,23,kuning
18	255,183,9,kuning
19	255,255,0,kuning
20	255,166,0,kuning
21	249,217,94,kuning
22	246,191,39,kuning
23	37,202,38,hijau
24	35,67,17,hijau
25	0,255,0,hijau
26	64,189,85,hijau
27	159,217,140,hijau
28	0,166,82,hijau
29	0,128,1,hijau
30	33,83,54,hijau
31	125,194,75,hijau
32	123,252,1,hijau
33	125,232,89,hijau
34	255,103,0,oranye
35	236,193,91,oranye
36	255,103,0,oranye
37	255,122,1,oranye
38	255,102,0,oranye
39	254,101,35,oranye
40	255,153,0,oranye
41	252,79,19,oranye
42	255,127,0,oranye
43	255,128,0,oranye
44	253,251,251,putih
45	248,249,254,putih
46	249,255,241,putih
47	250,240,230,putih
48	255,237,231,putih
49	243,239,227,putih
50	229,224,221,putih
51	242,233,226,putih
52	238,228,220,putih
53	242,233,228,putih
54	47,47,47,hitam
55	28,28,32,hitam
56	10,18,13,hitam
57	0,0,0,hitam
58	9,0,0,hitam

4.1.6 Membuat data evaluasi (Data Testing)

Pada proses pembuatan data testing ini, data testing di peroleh dari warna pada objek yang di tangkap oleh webcam yang sudah di proses di dalam tensorflow, kemudian warna objek tersebut akan ditentukan nilai pixel RGBnya, proses selanjutnya dengan color histogram untuk menentukan fitur ekstraksinya atau nilai pixel RGB warna tersebut. Untuk pseudocode menentukan nilai RGB dapat di lihat pada Gambar 4.6 dan hasil data *testing* pada gambar 4.7

tengah dan juga memfokuskan warna pada objek yang di deteksi. Untuk pseudocode program *crop* dapat di lihat pada Gambar 4.4 dan hasilnya dapat dilihat pada Gambar 4.5 di bawah ini

```

Myfunction(testImage):
    image = testImage

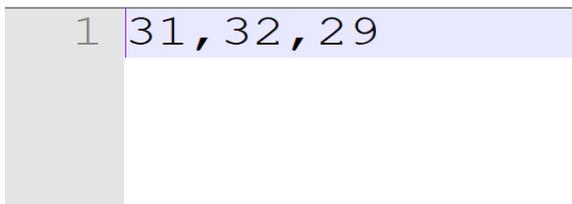
    split = cv2.split(image)
    colors = ("b", "g", "r")
    features = []
    features_data = ""
    counter = 0
    for (splitti, color) in zip(split, colors)do
        counter = counter + 1

        hist = cv2.calcHist([splitti], [0], None, [256], [0, 256])
        features.extend(hist)#menggabungkannya menjadi satu list array 1 dimensi

        # find the peak pixel values for R, G, and B
        elem = np.argmax(hist) #mencari nilai maximal pada array

        if (counter == 1)then
            blue = str(elem)
        elif (counter ==2)then
            green = str(elem)
        elif (counter ==3)then
            red = str(elem)
        features_data = red + "," + green + "," + blue
    endif
endfor
with open(current_path+"/utils/color_recognition_module/"+"test.data", "w") as myfile:
    myfile.write(features_data)
    
```

Gambar 4. 6 Pembuatan data testing



Gambar 4. 7 Hasil data testing

4.1.7 Melakukan klasifikasi warna

Tahapan selanjutnya pengklasifikasi warna objek, tahapan ini bertujuan untuk menentukan warna dari objek yang dideteksi, dengan cara menggunakan algoritma K-NN (*K-Nearest Neighbor*). Tahapan ini mencari jarak yang terdekat dari data testing dalam data pelatihan untuk menentukan hasil dari warna objek tersebut. Adapun tahapan proses dalam melakukan klasifikasi menggunakan K-NN.

4.1.7.1 Menentukan nilai K

Tahapan yang pertama perlu ditentukan banyaknya k tetangga terdekat bertujuan untuk melakukan klasifikasi data baru. Nilai k yang

digunakan adalah 3. Banyaknya nilai K sebaiknya menggunakan nilai ganjil untuk menentukan klasifikasi. Penentuan nilai k dipertimbangkan oleh banyaknya data yang ada dan ukuran dimensi yang di bentuk oleh data, semakin banyaknya data maka jumlah k yang digunakan semakin kecil begitu juga sebaliknya.

4.1.7.2 Menghitung Jarak antara data evaluasi dengan data pelatihan

Tahapan ini

```

neighbors = []
for x in range(k)do
    neighbors.append(distances[x][0])
endfor
return neighbors
    
```

menghitung jarak dengan menggunakan rumus jarak *Euclidian*. Gambar di bawah ini merupakan psodocode untuk menghitung jarak menggunakan rumus *euclidian* dapat di lihat pada Gambar 4.8

```

calculateEuclideanDistance(variable1, variable2, length)
distance = 0
for x in range(length)do
    distance += pow((variable1[x] - variable2[x]), 2)
endfor
return np.sqrt(distance)
    
```

Gambar 4. 8 Menghitung jarak Euclidean

4.1.7.3 Mengurutkan hasil perhitungan jarak secara ascending

Tahapan ini merupakan proses pengurutan hasil proses perhitungan dengan jarak *Euclidian* secara *ascending*. Tahapan pertama data hasil penghitungan jarak digabungkan dengan data keseluruhan training kemudian dilakukan

```

kNearestNeighbors(training_feature_vector, testInstance, k):
    distances = []
    length = len(testInstance)
    for x in range(len(training_feature_vector))do
        dist = calculateEuclideanDistance(testInstance,
        training_feature_vector[x], length)
        distances.append((training_feature_vector[x], dist))
    endfor
    distances.sort(key = operator.itemgetter(1))
    neighbors = []
    for x in range(k)do
        neighbors.append(distances[x][0])
    endfor
    return neighbors
    
```

sorting. Gambar 4.9 adalah pseudocode proses penyatuan data dan sorting data

Gambar 4. 9 Penyatuan data dan sorting data

4.1.7.4 Mengumpulkan Kategori klasifikasi berdasarkan nilai k

Tahapan ini mengumpulkan kategori berdasarkan nilai k. Data yang telah di sorting kemudian akan diklasifikasi sesuai dengan nilai k yaitu 3, pseudocode bisa dilihat pada Gambar 4.10

Gambar 4. 10 Mengumpulkan Kategori klasifikasi berdasarkan nilai k

4.1.7.5 Mencari nearest neighbor mayoritas

Tahapan ini mencari kategori nearest neighbor yang paling mayoritas maka dapat di prediksi kategori jenis dari warna. Gambar 4.11 adalah pseudocode proses mencari nearest neighbor mayoritas

```

responseOfNeighbors(neighbors)
  all_possible_neighbors = {}
  for x in range(len(neighbors)):
    response = neighbors[x][1]
    if response in all_possible_neighbors then
      all_possible_neighbors[response] += 1
  return sortedVotes[0][0]

endif
endfor
sortedVotes = sorted(all_possible_neighbors.items(), key=operator.itemgetter(1), reverse=True)
    
```

Gambar 4. 11 Mencari nearest neighbor mayoritas

4.1.8 Mengeluarkan hasil berupa suara

Tahapan ini pemberian hasil berupa suara warna yang dari objek yang dideteksi.

4.2 Pengujian

4.2.1 Tujuan Pengujian

Tujuan dari pengujian ini adalah menguji apakah sistem dan alat yang di bangun sudah berjalan sesuai dengan fungsinya yaitu mendeteksi warna pada objek benda. Tahapan pengujian ini dilakukan dengan berbagai cara mulai dari mendeteksi objek didalam ruangan yang minim cahaya, mendeteksi objek benda didalam ruangan dengan cahaya yang cukup terang dan mendeteksi setiap objek dan warna mulai dari jarak 20 cm sampai dengan 60cm.

4.2.2 Skenario Pengujian

4.2.2.1 Pengujian sistem pada laptop dengan suara dan tanpa suara

Skenario ini bertujuan untuk melihat kecepatan proses deteksi pada saat sistem dijalankan pada laptop.

4.2.2.2 Pengujian sistem dengan raspberry Pi model 3 B dengan suara dan tanpa suara.

Skenario ini bertujuan untuk melihat kecepatan proses deteksi pada saat sistem di jalankan menggunakan raspberry Pi.

4.2.3 Alur Pendeteksian Pada Program

Tujuan dari alur pendeteksian agar mengetahui cara objek terdeteksi oleh sistem

1. Pada saat program dijalankan pada potongan program webcam.py ,langkah pertama dibuatnya variabel untuk menampung nilai untuk deteksi warna dengan *value* 1 dan tahapan

mengirimkan 2 parameter yaitu parameter model dan *label map* pada fungsi *set_model* yang bertujuan untuk mendefinisikan *ssd mobile v1 coco* sebagai objek yang di deteksi dan melakukan pemetaan index pada *ssd mobilenet* menggunakan *label map* *mscoco_label_map.pbtxt*.

- Hasil dari definisi objek tersebut, kemudian di simpan pada 2 variabel dengan nama *category_index* dan *detection graph*. Variabel *category_index*, *detection graph* dan *is_color_recognition_enabled* yang berisikan nilai, kemudian dibawa pada program *webcame.py* pada bagian fungsi *detection*.
- Pada bagian *webcame.py* menjalankan atau mengeksekusi grafik pada *detection_ graph*, kemudian membuat variable untuk menampung deteksi objek, gambar objek, *score* objek, class objek, kotak deteksi dari proses hasil pelatihan pada *ssd mbilenet v1 dataset coco* sebelumnya. Tahapan selanjutnya mengaktifkan kamera dengan bantuan library *cv2* untuk membuka camera *webcam*. Pada saat bersamaan mengirimkan frame yang di tangkap dan data pada proses pembuatan box dan pendeteksi warna pada potongan program *visualization_util1.py*
- Pada *visualization_util1.py* merupakan proses pembuatan box untuk mendefinisikan objek yang dideteksi, kemudian penulisan string nama objek yang terdeteksi dan warna objek. Untuk proses pendeteksian warna objek yang dideteksi, objek yang terdeteksi kemudian dikirim pada potongan program *color_recognition_api.py*.

- Pada *color_recognition_api.py* gambar yang ukuran awalnya kurang lebih 300 x 400 akan di crop menjadi 50x50 untuk memfokuskan pada warna objek. Kemudian dari warna yang di dapat dibuatlah data testing atau evaluasi untuk dicocokkan dengan data pelatihan.
- Pada tahapan selanjutnya data evaluasi dan data training di kirim pada potongan program *knn_classifier* untuk melakukan klasifikasi warna pada objek.

4.2.4 Hasil Pengujian

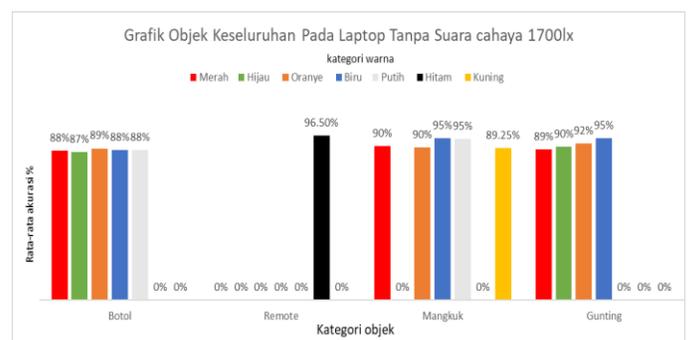
Berikut adalah hasil pengujian dari berbagai cara seperti yang sudah di jelaskan pada skenario di atas

4.2.4.1 Akurasi terbaik laptop tanpa informasi suara pada penerangan sebesar 1700lx

Tabel 4. 2 Tabel rentang akurasi terbaik pada penerangan 1700lx

Nama	Warna	jarak akurasi terbaik (cm)	Rata-rata akurasi di peroleh	Hasil
Botol	merah	20-50	87.5%	BERHASIL
	hijau	20-50	86.75%	BERHASIL
	oranye	20-50	89%	BERHASIL
	biru	20-40	88%	BERHASIL
	putih	20-50	88%	BERHASIL
Remote	hitam	20-30	96.5%	BERHASIL
Mangkuk	merah	20-50	90.25%	BERHASIL
	putih	20-30	94.5%	BERHASIL
	biru	20-30	95%	BERHASIL
	oranye	20-50	89.5%	BERHASIL
	kuning	20-50	89.25%	BERHASIL
Gunting	merah	20-30	88.5%	BERHASIL
	oranye	20-30	92%	BERHASIL
	biru	20	95%	BERHASIL
	hijau	20-30	90%	BERHASIL

Pada Tabel 4.50 dapat dilihat pada objek botol rentang jarak terbaik berkisar 20-50 cm, pada remote 20-30 cm, pada mangkuk berkisar 20-50



cm dan pada gunting berkisar 20-30 cm

Gambar 4. 12 Grafik rentang akurasi terbaik dengan laptop pada penerangan 1700lx

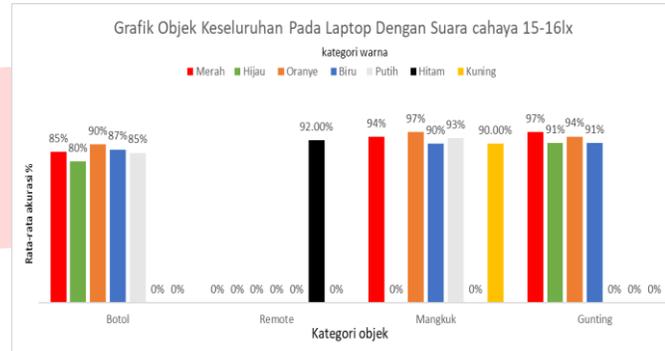
4.2.4.2 Akurasi terbaik laptop dengan informasi suara pada penerangan sebesar 1700lx

Tabel 4. 3 Tabel rentang akurasi terbaik dengan informasi suara

Nama	warna	jarak akurasi terbaik (cm)	Rata-rata akurasi di peroleh	Hasil	Suara
Botol	merah	20-50	88.25%	BERHASIL	BERH
	hijau	20-50	90%	BERHASIL	BERH
	oranye	20-50	89.25%	BERHASIL	BERH
	biru	20-30	90.5%	BERHASIL	BERH
	putih	20-50	84.75%	BERHASIL	BERH
Remote	hitam	20-40	91.3%	BERHASIL	BERH
Mangkuk	merah	20-50	89.75%	BERHASIL	BERH
	putih	20-50	92%	BERHASIL	BERH
	biru	20-40	91.3%	BERHASIL	BERH
	oranye	20-50	87.25%	BERHASIL	BERH
	kuning	20-50	89.75%	BERHASIL	BERH
gunting	merah	20-30	93.5%	BERHASIL	BERH
	oranye	20-30	91%	BERHASIL	BERH
	biru	20-30	92%	BERHASIL	BERH
	hijau	20-30	88.5%	BERHASIL	BERH

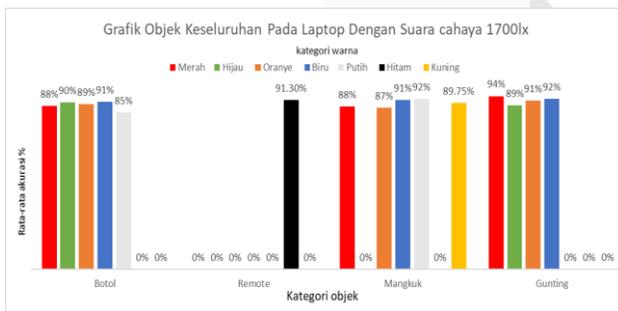
Nama	warna	jarak akurasi terbaik (cm)	Rata-rata akurasi di peroleh	Hasil
Botol	merah	20-30	92%	BERHASIL
	hijau	50	88%	BERHASIL
	oranye	20-50	86.5%	BERHASIL
	biru	20-30	89.5%	BERHASIL
	putih	20-30	87.5%	BERHASIL
Remote	hitam	20-40	96.3%	BERHASIL
Mangkuk	merah	20-50	89.5%	BERHASIL
	putih	20-30	97.5%	BERHASIL
	biru	20-50	92.25%	BERHASIL
	oranye	20-50	90%	BERHASIL
	kuning	20-50	92%	TIDAK BERHASIL
gunting	merah	20-30	94%	BERHASIL
	oranye	20-30	96%	BERHASIL
	biru	20-30	83.5%	BERHASIL
hijau	20	98%	BERHASIL	

Pada Tabel 4.52 dapat dilihat pada objek botol rentan



g jarak terbaik berkisar 20-30 cm, pada remote

Pada Tabel 4.51 dapat dilihat pada objek botol rentang jarak terbaik berkisar 20-50 cm, pada remote berkisar 20-40 cm, pada mangkuk berkisar 20-50 cm dan pada gunting berkisar 20-30 cm.



Gambar 4. 13 Grafik rentang akurasi terbaik dengan informasi suara pada penerangan 1700lx

4.2.4.3 Akurasi terbaik laptop tanpa informasi suara pada penerangan sebesar 15-16lx

Tabel 4. 4 Tabel rentang akurasi terbaik pencahayaan 15-16lx

berkisar 20-40 cm, pada mangkuk berkisar 20-50 cm dan pada gunting berkisar 20-30 cm.



Gambar 4. 14 Grafik rentang akurasi terbaik laptop pencahayaan 15-16lx

4.2.4.4 Akurasi terbaik laptop dengan informasi suara pada penerangan sebesar 15-16lx

Tabel 4. 53 Tabel rentang akurasi terbaik pencahayaan 15-16lx dengan informasi suara

Nama	warna	jarak akurasi terbaik (cm)	Rata-rata akurasi di peroleh	Hasil	Si
Botol	merah	20-40	85.3%	BERHASIL	B
	hijau	40	80%	BERHASIL	B
	oranye	20-40	89.7%	BERHASIL	B
Remote	biru	20-30	86.7%	BERHASIL	B
	putih	20-50	84.75%	BERHASIL	B
	hitam	20-40	92%	BERHASIL	B
Mangkuk	merah	20-40	93.7%	BERHASIL	B
	putih	20-30	93%	BERHASIL	B
	biru	20-30	90%	BERHASIL	B
gunting	oranye	20-50	96.5%	BERHASIL	B
	kuning	20 dan 40	90%	BERHASIL	B
	merah	20-30	96.5%	BERHASIL	B
gunting	oranye	20-30	94%	BERHASIL	B
	biru	20-30	90.5%	BERHASIL	B
	hijau	20-30	90.5%	BERHASIL	B

Pada Tabel 4.53 dapat dilihat pada objek botol rentang jarak terbaik berkisar 20-40 cm, pada remote berkisar 20-40 cm, pada mangkuk berkisar 20-50 cm dan pada gunting berkisar 20-30 cm.

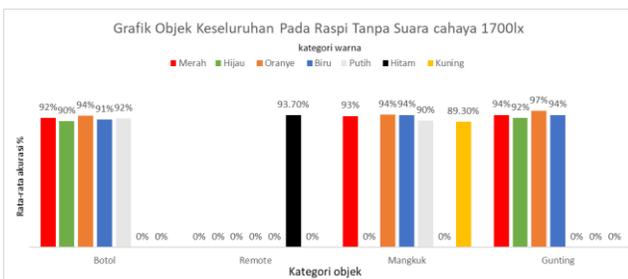
Gambar 4. 112 Grafik rentang akurasi terbaik dengan laptop dan suara pencahayaan 15-16lx

4.2.4.5 Akurasi terbaik raspi tanpa informasi suara pada penerangan sebesar 1700lx

Tabel 4. 54 Tabel rentang akurasi terbaik raspi pencahayaan 1700lx tanpa informasi suara

Nama	warna	jarak akurasi terbaik (cm)	Rata-rata akurasi di peroleh	Hasil
Botol	merah	20-30	92%	BERHASIL
	hijau	20-30	89.5%	BERHASIL
	oranye	20-30	93.5%	BERHASIL
Remote	biru	20-40	90.7%	BERHASIL
	putih	20-30	91.5%	BERHASIL
	hitam	20-40	93.7%	BERHASIL
Mangkuk	merah	20-50	93.25%	BERHASIL
	putih	20-30	90%	BERHASIL
	biru	20-30	94%	BERHASIL
gunting	oranye	20-40	94.3%	BERHASIL
	kuning	20-40	89.3%	BERHASIL
	merah	20-30	94%	BERHASIL
gunting	oranye	20-30	97%	BERHASIL
	biru	20-30	94%	BERHASIL
	hijau	20	92%	BERHASIL

Pada Tabel 4.54 dapat dilihat pada objek botol



rentang jarak terbaik berkisar 20-30 cm, pada remote berkisar 20-40 cm, pada mangkuk berkisar 20-30 cm dan pada gunting berkisar 20-30 cm.

Gambar 4. 15 Grafik rentang akurasi terbaik dengan raspi tanpa suara pencahayaan 1700lx

4.2.4.6 Akurasi terbaik raspi dengan informasi suara pada penerangan sebesar 1700lx

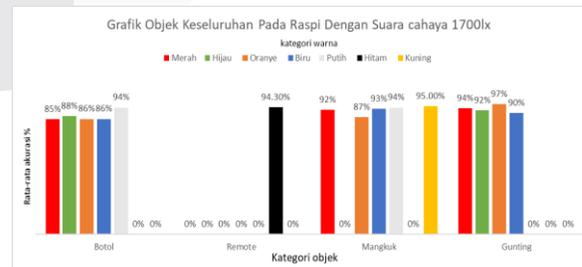
Tabel 4. 5 Tabel rentang akurasi terbaik raspi pencahayaan 1700lx dengan informasi suara

Nama	warna	jarak akurasi terbaik (cm)	Rata-rata akurasi di peroleh	Hasil	Suara
Botol	merah	20-40	85.3%	BERHASIL	BERHASIL
	hijau	20-50	87.75%	BERHASIL	BERHASIL
	oranye	20-30	85.5%	BERHASIL	BERHASIL
Remote	Biru	20-30	85.5%	BERHASIL	BERHASIL
	Putih	20-30	94%	BERHASIL	BERHASIL
	hitam	20-40	94.3%	BERHASIL	BERHASIL
Mangkuk	merah	20-40	92.3%	BERHASIL	BERHASIL
	putih	20-30	94%	BERHASIL	BERHASIL
	biru	20-40	93%	BERHASIL	BERHASIL
gunting	oranye	30-50	87%	BERHASIL	BERHASIL
	kuning	20	95%	BERHASIL	BERHASIL
	merah	20-30	93.5%	BERHASIL	BERHASIL
gunting	oranye	20-30	96.5%	BERHASIL	BERHASIL
	biru	20-30	90%	BERHASIL	BERHASIL
	hijau	20-30	92%	BERHASIL	BERHASIL

Pada Tabel 4.55 dapat dilihat pada objek botol rentang jarak terbaik berkisar 20-30 cm, pada remote berkisar 20-40 cm, pada mangkuk berkisar 20-40 cm dan pada gunting berkisar 20-30 cm.

Gambar 4. 114 Grafik rentang akurasi terbaik dengan raspi dengan suara pencahayaan 1700lx

4.2.4.7 Akurasi terbaik raspi tanpa informasi suara pada penerangan sebesar 15-16lx



informasi suara pada penerangan sebesar 15-16lx

Tabel 4. 56 Tabel rentang akurasi terbaik raspi dengan pencahayaan 15-16lx tanpa informasi suara

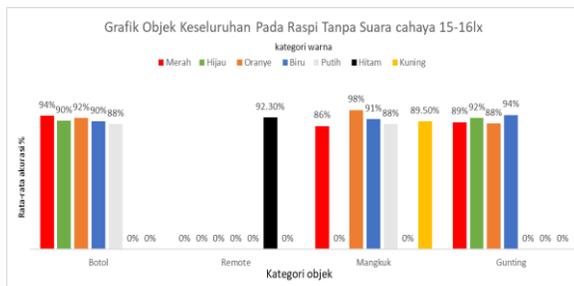
Nama	warna	jarak akurasi terbaik (cm)	Rata-rata akurasi di peroleh	Hasil
Botol	merah	20-30	93.5%	BERHASIL
	hijau	20-30	90%	BERHASIL
	oranye	20-30	92%	TIDAK BE
	biru	20-60	89.6%	TIDAK BE
	putih	20-30	87.5%	TIDAK BE
Remote	hitam	20-40	92.3%	BERHASIL
Mangkuk	merah	20-40	86.3%	TIDAK BE
	putih	20-30	87.5%	TIDAK BE
	Biru	20-30	91%	TIDAK BE
	oranye	20-30	97.5%	TIDAK BE
	kuning	20-30	89.5%	TIDAK BE
gunting	merah	20	89%	TIDAK BE
	oranye	20	88%	TIDAK BE
	biru	20	94%	TIDAK BE
	hijau	20	92%	TIDAK BE

Nama	warna	jarak akurasi terbaik (cm)	Rata-rata akurasi di peroleh	Hasil	Suara
Botol	merah	20-40	91%	BERHASIL	BERHASIL
	hijau	20	93%	BERHASIL	BERHASIL
	oranye	20-40	90%	TIDAK BERHASIL	BERHASIL
	biru	20-50	91.25%	TIDAK BERHASIL	BERHASIL
	putih	20,40-50	83.3%	TIDAK BERHASIL	BERHASIL
Remote	hitam	20-40	97%	BERHASIL	BERHASIL
Mangkuk	merah	20-40	86.7%	TIDAK BERHASIL	BERHASIL
	putih	20-40	91%	TIDAK BERHASIL	BERHASIL
	biru	20-30	95%	TIDAK BERHASIL	BERHASIL
	oranye	20-30	95%	TIDAK BERHASIL	BERHASIL
	kuning	20-30	89.5%	TIDAK BERHASIL	BERHASIL
gunting	merah	20-30	92.5%	BERHASIL	BERHASIL
	oranye	20	93%	BERHASIL	BERHASIL
	biru	20	94%	BERHASIL	BERHASIL
	hijau	20	94%	BERHASIL	BERHASIL

Pada Tabel 4.56 dapat dilihat pada objek botol rentang jarak terbaik berkisar 20-30 cm, pada remote berkisar 20-40 cm, pada mangkuk berkisar 20-30 cm dan pada gunting berkisar 20 cm, namun hasil warna yang dideteksi tidak begitu baik.

Pada Tabel 4.57 dapat dilihat pada objek botol rentang jarak terbaik berkisar 20-50 cm, pada remote berkisar 20-40 cm, pada mangkuk berkisar 20-30 cm dan pada gunting berkisar 20 cm, namun hasil warna yang dideteksi tidak begitu baik.

Gambar 4. 116 Grafik rentang akurasi terbaik dengan raspi dan suara pencahayaan 15-16lx



Gambar 4. 115 Grafik rentang akurasi terbaik dengan raspi tanpa suara pencahayaan 15-16lx

4.2.4.8 Akurasi terbaik raspi dengan informasi suara pada penerangan sebesar 15-16lx

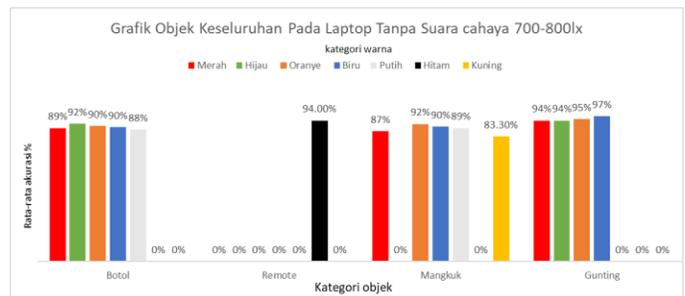
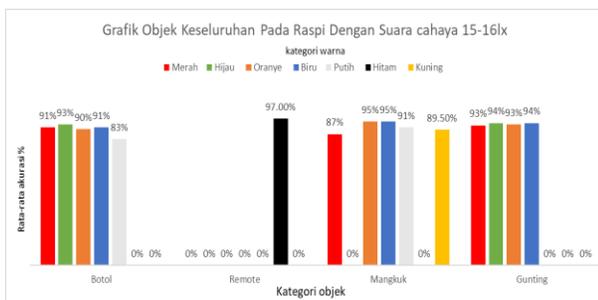
Tabel 4. 57 Tabel rentang akurasi terbaik raspi pada pencahayaan 15-16lx dengan informasi suara

4.2.4.9 Akurasi terbaik laptop tanpa informasi suara pada penerangan sebesar 700-800lx

Tabel 4. 58 Tabel rentang akurasi terbaik lapto pada pencahayaan 700-800lx tanpa suara

Nama	warna	jarak akurasi terbaik (cm)	Rata-rata akurasi diperoleh	Hasil
Botol	merah	20-50	89%	BERHASIL
	hijau	20-40	92%	BERHASIL
	oranye	20-50	90.25%	BERHASIL
	biru	20-50	89.50%	BERHASIL
	putih	20-50	88%	BERHASIL
Remote	hitam	20-40	94%	BERHASIL
Mangkuk	merah	20-50	87%	BERHASIL
	putih	20-50	89%	BERHASIL
	biru	20-40	90%	BERHASIL
	oranye	20-40	91.70%	BERHASIL
	kuning	20-40	89.30%	BERHASIL
gunting	merah	20-30	94%	BERHASIL
	oranye	20-30	95%	BERHASIL
	biru	20-30	97%	BERHASIL
	hijau	20-30	94%	BERHASIL

Pada Tabel 4.58 dapat dilihat pada objek botol rentang jarak terbaik berkisar 20-50 cm, pada remote 20-40 cm, pada mangkuk berkisar 20-50 cm dan pada gunting berkisar 20-30 cm.



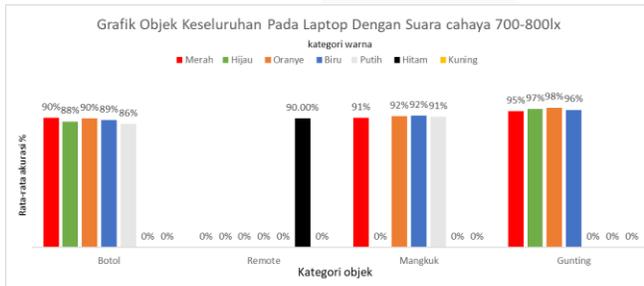
Gambar 4. 16 Grafik rentang akurasi terbaik dengan laptop tanpa suara pencahayaan 700-800lx

4.2.4.10 Akurasi terbaik laptop dengan informasi suara pada penerangan sebesar 700-800lx

Tabel 4. 59 Tabel rentang akurasi terbaik laptop pada pencahayaan 700-800lx dengan suara

Nama	warna	jarak akurasi terbaik (cm)	Rata-rata akurasi diperoleh	Hasil	Suar
Botol	merah	20-50	90.25%	BERHASIL	BERI
	hijau	20-60	87.80%	BERHASIL	BERI
	oranye	20-50	90%	BERHASIL	BERI
	biru	20-50	88.75%	BERHASIL	BERI
Remote	putih	20-40	86%	BERHASIL	BERI
	hitam	20-40	90%	BERHASIL	BERI
Mangkuk	merah	20-50	90.50%	BERHASIL	BERI
	putih	20-40	91%	BERHASIL	BERI
gunting	biru	20-40	92%	BERHASIL	BERI
	oranye	20-50	91.50%	BERHASIL	BERI
	merah	20-30	95%	BERHASIL	BERI
	oranye	20-30	97.50%	BERHASIL	BERI
gunting	biru	20-30	96%	BERHASIL	BERI
	hijau	20-30	96.50%	BERHASIL	BERI

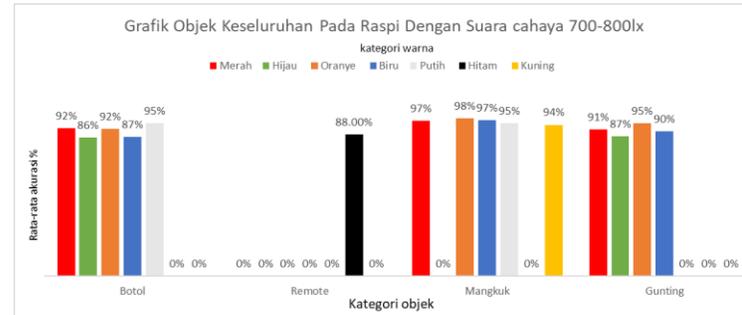
Pada Tabel 4.59 dapat dilihat pada objek botol rentang jarak terbaik berkisar 20-60 cm, pada remote 20-40 cm, pada mangkuk berkisar 20-50 cm dan pada gunting berkisar 20-30 cm.



Gambar 4. 118 Grafik rentang akurasi terbaik dengan laptop dan suara pencahayaan 700-800lx

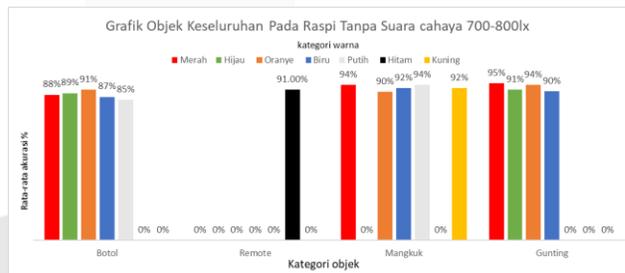
4.2.4.11 Akurasi terbaik raspi tanpa informasi suara pada penerangan sebesar 700-800lx

Tabel 4. 60 Tabel rentang akurasi terbaik raspi pada pencahayaan 700-800lx tanpa suara



Nama	warna	jarak akurasi terbaik (cm)	Rata-rata akurasi diperoleh	Hasil
Botol	merah	20-40	88%	BERHASIL
	hijau	20-50	89%	BERHASIL
	oranye	20-40	91%	TIDAK BERHASIL
	biru	20-40	86.70%	TIDAK BERHASIL
Remote	putih	20-30 dan 50-60	85%	TIDAK BERHASIL
	hitam	20-30	91%	BERHASIL
Mangkuk	merah	20-40	94%	BERHASIL
	putih	20-40	94%	TIDAK BERHASIL
	biru	20-30	92%	BERHASIL
	oranye	20-50	89.75%	TIDAK BERHASIL
gunting	kuning	20-40	92%	TIDAK BERHASIL
	merah	20	95%	BERHASIL
	oranye	20-30	94%	TIDAK BERHASIL
	biru	20	90%	TIDAK BERHASIL
gunting	hijau	20-30	91%	TIDAK BERHASIL

Pada Tabel 4.60 dapat dilihat pada objek botol rentang jarak terbaik berkisar 20-50 cm, pada remote 20-30 cm, pada mangkuk berkisar 20-50



Gambar 4. 119 Grafik rentang akurasi terbaik dengan raspi tanpa suara pencahayaan 700-800lx

4.2.4.12 Akurasi terbaik raspi dengan informasi suara pada penerangan sebesar 700-800lx

Tabel 4. 61 Tabel rentang akurasi terbaik raspi pada pencahayaan 700-800lx dengan suara

cm dan pada gunting berkisar 20-30 cm.

Nama	warna	jarak akurasi terbaik (cm)	Rata-rata akurasi di peroleh	Hasil
Botol	merah	20-40	92%	BERHASIL
	hijau	20-30	86%	BERHASIL
	oranye	20-30	91.50%	TIDAK BERHASIL
	biru	20 dan 50	86.50%	BERHASIL
Remote	putih	20-30	95%	TIDAK BERHASIL
	hitam	20-50	88%	BERHASIL
Mangkuk	merah	20-30	96.50%	BERHASIL
	putih	20-30	95%	TIDAK BERHASIL
	biru	20-30	97%	BERHASIL
	oranye	30	98%	BERHASIL
gunting	kuning	20-30	94%	TIDAK BERHASIL
	merah	20-30	91%	BERHASIL
	oranye	20-30	95%	TIDAK BERHASIL
	biru	20	90%	TIDAK BERHASIL
	hijau	30	87%	BERHASIL

Pada Tabel 4.61 dapat dilihat pada objek botol rentang jarak terbaik berkisar 20-50 cm, pada remote 20-50 cm, pada mangkuk berkisar 20-30 cm dan pada gunting berkisar 20-30 cm

Gambar 4. 17 Grafik rentang akurasi terbaik dengan raspi dengan suara pencahayaan 700-800lx

5. Kesimpulan dan Saran

5.1 Kesimpulan

Dari rangkaian pengujian yang dilakukan pada sistem sistem deteksi warna untuk membantu orang buta warna berbasis machine learning, maka di simpulkan bahwa

1. Akurasi deteksi dan warana deteksi yang baik terdapat pada rentangan jarak 20 -30 cm.
2. Sistem dapat mendeteksi warna dengan baik pada ruangan dengan pencahayaan terang sebesar 800-1700 lx dengan laptop.
3. Warna akan terdeteksi apabila objek benda terdeteksi, seperti pengujian terhadap objek gunting.
4. Cahaya mempengaruhi terhadap deteksi warna pada benda.

5. Sistem dapat memberikan informasi berupa suara pada jenis warna yang di deteksinya.

5.2 Saran

Untung pengembangan sistem lebih lanjut pada penelitian ini, disarankan untuk

1. Perlunya menghilangkan bayangan pada gambar agar kualitas deteksi warna dapat lebih akurat lagi.
2. Menggunakan raspi versi tertinggi agar dapat diminimalisir delay pada sistem.
3. Agar sistem kedepannya bisa memberikan informasi suara untuk nama objek dan warna sekaligus.

6. Daftar Pustaka

[1] “(1) (DOC) Makalah Buta warna | Ikzan Khalaf - Academia.edu.” [Online]. Available: https://www.academia.edu/11784979/Makalah_Buta_warna. [Accessed: 20-Oct-2019].

[2] R. Kurnia, “Penentuan Tingkat Buta Warna Berbasis His Dengan Banyak Warna Pada Citra Isihara,” *Semin. Nas. Apl. Teknol. Inf. 2009 (SNATI 2009)*, vol. IV, no. 1, pp. 54–77, 2009.

[3] T. Elektro, P. N. Malang, and J. S. Malang, “Jumlah penyandang buta warna di Indonesia semakin meningkat setiap tahun . Dari total penduduk yang berjumlah 255 juta jiwa , sebanyak 0 , 7 % terkena kelainan genetika yang penyandanganya tidak mampu membedakan tingkat gradasi suatu warna . Sedangkan dilu,” vol. 3, pp. 156–

- 167, 2017.
- [4] "Difference between Machine learning and Artificial Intelligence - GeeksforGeeks." [Online]. Available: <https://www.geeksforgeeks.org/difference-between-machine-learning-and-artificial-intelligence/>. [Accessed: 26-Oct-2019].
- [5] S. R. DEWI, "Deep Learning Object Detection Pada Video," 2018.
- [6] S. Z. Jasmine and A. Warman, "Rancang Bangun Alat Pendeteksi Warna Pada Kaleng Minuman Menggunakan Sensor TCS3200 Berbasis Arduino Uno," 2017.
- [7] it-jurnal.com, "biomas Chem Eng," *Biomass Chem Eng*, vol. 49, no. 23–6, pp. 3–16, 2015.
- [8] N. Safaat, "Nurzudin Safaat H (2012:9)," pp. 10–23, 2012.
- [9] A. Ahmad, "Mengenal Artificial Intelligence, Machine Learning, Neural Network, dan Deep Learning," *J. Teknol. Indones.*, no. October, p. 3, 2017.
- [10] "TensorFlow." [Online]. Available: <https://www.tensorflow.org/>. [Accessed: 20-Oct-2019].
- [11] "TensorFlow - Custom Object Detection API - JournalToday." [Online]. Available: <https://imamdigmi.github.io/post/tensorflow-custom-object-detection/>. [Accessed: 27-Oct-2019].
- [12] "(79) (DOC) TRANSFORMASI CITRA WARNA Apa itu Warna | natu geleng - Academia.edu." [Online]. Available: https://www.academia.edu/36672466/TRANSFORMASI_CITRA_WARNA_Apa_itu_Warna. [Accessed: 11-Apr-2020].
- [13] "(79) (DOC) PENGOLAHAN CITRA DIGITAL | agung tri setia - Academia.edu." [Online]. Available: https://www.academia.edu/11422880/PENGOLAHAN_CITRA_DIGITAL. [Accessed: 11-Apr-2020].
- [14] R. D. Liklikwatil, E. Noersasongko, and C. Supriyanto, "Optimasi K-Nearest Neighbor Dengan Particle Swarm Optimization Untuk Memprediksi Harga Komoditi Karet," *J. Sist. Inf. Dan Teknol. Inf.*, vol. 7, no. 2, pp. 172–182, 2018.
- [15] N. Hermaduanty and S. Kusumadewi, "Sistem Pendukung Keputusan Berbasis Sms Untuk Menentukan Status Gizi Dengan Metode K- Nearest Neighbor," *Semin. Nas. Apl. Teknol. Inf. ISSN 1907-5022*, vol. 2008, no. Snati, pp. 49–56, 2008.
- [16] "Buy a Raspberry Pi 3 Model B – Raspberry Pi." [Online]. Available: <https://www.raspberrypi.org/products/raspberry-pi-3-model-b/>. [Accessed: 09-Apr-2020].
- [17] "Camera Module - Raspberry Pi Documentation." [Online]. Available: <https://www.raspberrypi.org/documentation/hardware/camera/>. [Accessed: 09-Apr-2020].
- [18] W. Liu *et al.*, "SSD: Single shot multibox detector," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9905 LNCS, pp. 21–37, 2016, doi: 10.1007/978-3-319-46448-0_2.
- [19] F. Liantoni, "Klasifikasi Daun Dengan Perbaikan Fitur Citra Menggunakan Metode K-Nearest Neighbor," *J. Ultim.*,

vol. 7, no. 2, pp. 98–104, 2016, doi:
10.31937/ti.v7i2.356.

- [20] “Perbedaan Supervised Learning and Unsupervised Learning - Information Communication Technology.” [Online]. Available:
<https://www.uc.ac.id/ict/perbedaan-supervised-learning-and-unsupervised-learning/>. [Accessed: 21-Jun-2020].
- [21] “Accuracy (error rate) Definition | DeepAI.” [Online]. Available:
<https://deepai.org/machine-learning-glossary-and-terms/accuracy-error-rate>. [Accessed: 25-Jun-2020].
- [22] “ahmetozlu/vehicle_counting_tensorflow: ‘MORE THAN VEHICLE COUNTING!’ This project provides prediction for speed, color and size of the vehicles with TensorFlow Object Counting API.” [Online]. Available:
https://github.com/ahmetozlu/vehicle_counting_tensorflow. [Accessed: 19-May-2020].