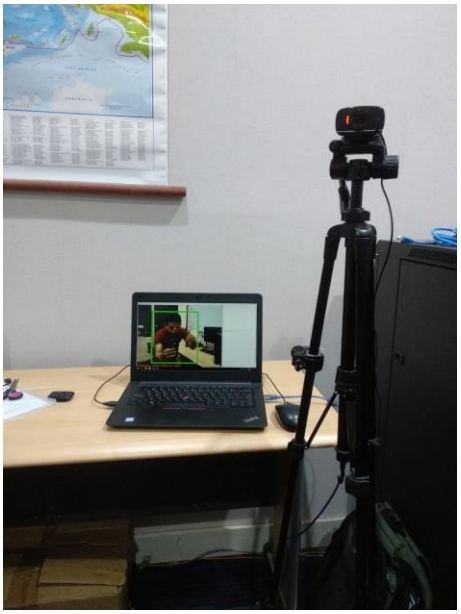
BAB V

IMPLEMENTASI DAN PENGUJIAN SISTEM

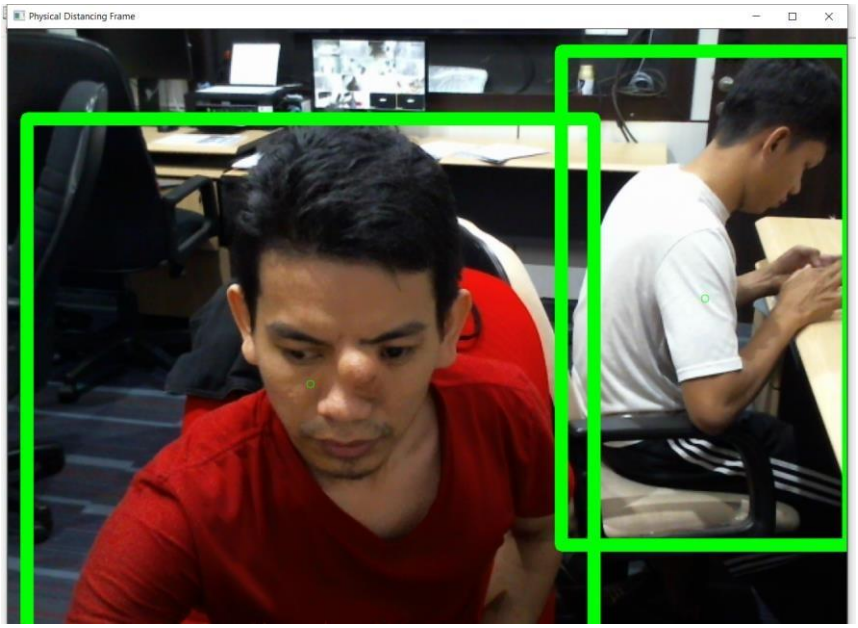
5.1 HASIL IMPLEMENTASI

Pada bab ini membahas tentang banyaknya hasil dari pengujian dan hasil penelitian Tugas Akhir ini. Dengan tujuan untuk mengetahui tingkat keberhasilan dari perancangan sistem yang telah diajukan dan dikerjakan. Pengujian dilakukan meliputi uji aplikasi menggunakan Metode YOLO, uji deteksi objek orang/manusia, dan pendeteksian pelanggaran jarak fisik dari setiap orang yang terdeteksi oleh kamera.

Gambar 5.1: Tampilan Hasil Rancangan Alat



Gambar 5.2: Tampilan Pengujian Alat Sedang Mendeteksi 1 Orang



Gambar 5.3: Tampilan Pengujian Alat Sedang Mendeteksi 2 Orang

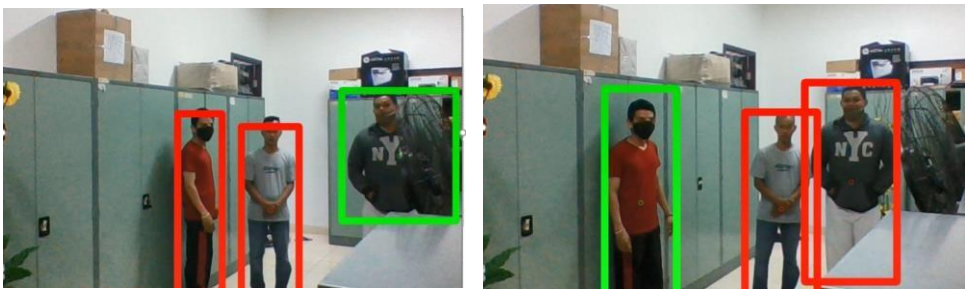
Gambar 5.1 merupakan hasil rancangan pendeteksi pelanggaran *physical distancing* yang telah dirancang oleh penulis. Gambar 5.2 dan 5.3 merupakan hasil pengujian deteksi objek dengan menggunakan metode YOLO.

5.1.1 Pengujian Pendeteksi Pelanggaran Physical Distancing

Gambar 5.4: Pengujian Jarak Aman Dengan Dua Orang

Gambar 5.5: Pengujian Jarak Aman Dengan Tiga Orang

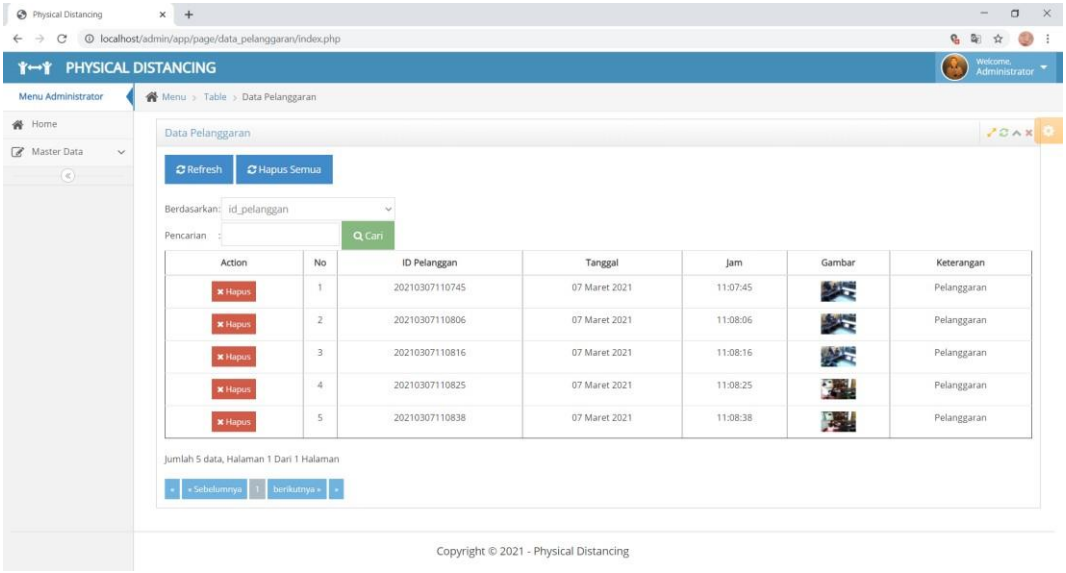
Gambar 5.6: Pengujian Jarak Tidak Aman Dengan Dua Orang



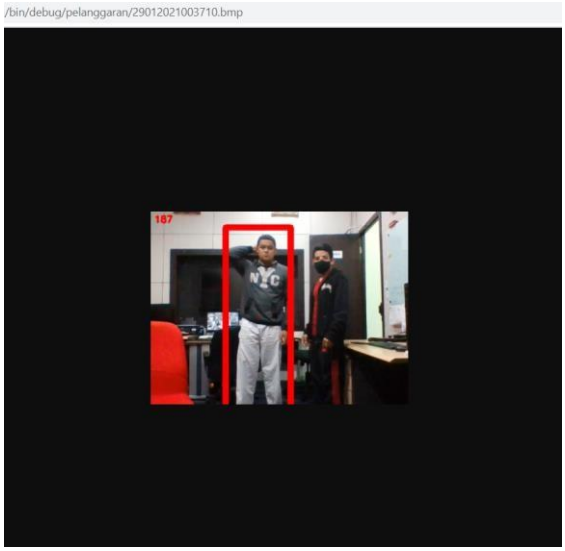
Gambar 5.7: Pengujian Jarak Tidak Aman Dengan Tiga Orang

Gambar 5.4 dan 5.5 merupakan hasil pengujian jarak aman dari dua orang maupun tiga orang, karena perhitungan posisi setiap koordinat masing-masing objek masih lebih besar dari nilai *Min_Jarak_Aman* yang diatur sebesar 200. Sedangkan untuk gambar 5.6 dan 5.7 adalah hasil pengujian jarak tidak aman dari dua orang maupun tiga orang, karena perhitungan posisi setiap koordinat masing-masing objek lebih kecil dari nilai *Min_Jarak_Aman*.

5.1.2 Tampilan Hasil Pelanggaran di Halaman *Website*



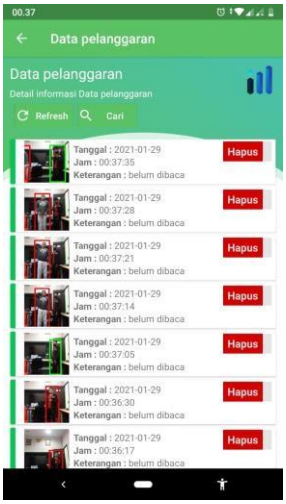
Gambar 5.8: Tampilan *Interface Website*



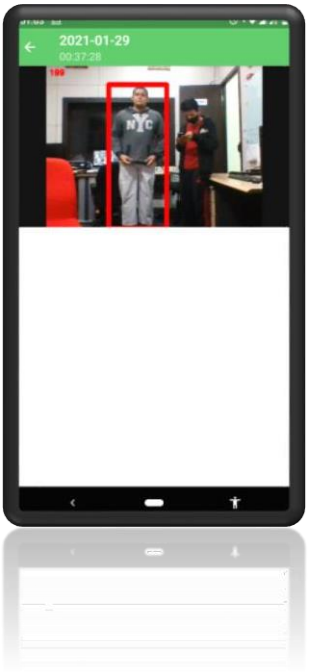
Gambar 5.9: Hasil *Capture* Pelaku Pelanggaran Di *Website*

Gambar 5.8 dan 5.9 merupakan interface berbasis *website* yang memiliki akses sebagai a*dministrator* yang dapat melihat hasil *capture* pelaku pelanggaran dan dapat mengubah *database* seperti menambahkan anggota untuk menjadi petugas keamanan kantin UNAMA agar dapat memonitoring pelanggaran yang terjadi melalui aplikasi android.

5.1.3 Tampilan Hasil Pelanggaran di Aplikasi Android



Gambar 5.10:Tampilan aplikasi android



Gambar 5.11: Hasil Capture Pelaku Pelanggaran di Aplikasi Android

Gambar 5.10 dan 5.11 merupakan interface berbasis android yang hanya memiliki akses sebagai anggota. Aplikasi android berfungsi untuk petugas keamanan kantin UNAMA agar dapat memonitoring pelanggaran *physical distancing* yang terjadi guna memberikan peringatan secara langsung kepada pelaku pelanggaran apabila menghiraukan suara peringatan yang telah berbunyi.

5.2 PENGUJIAN PERANGKAT LUNAK

5.2.1 Aplikasi Yang Digunakan Untuk Pengujian Sistem

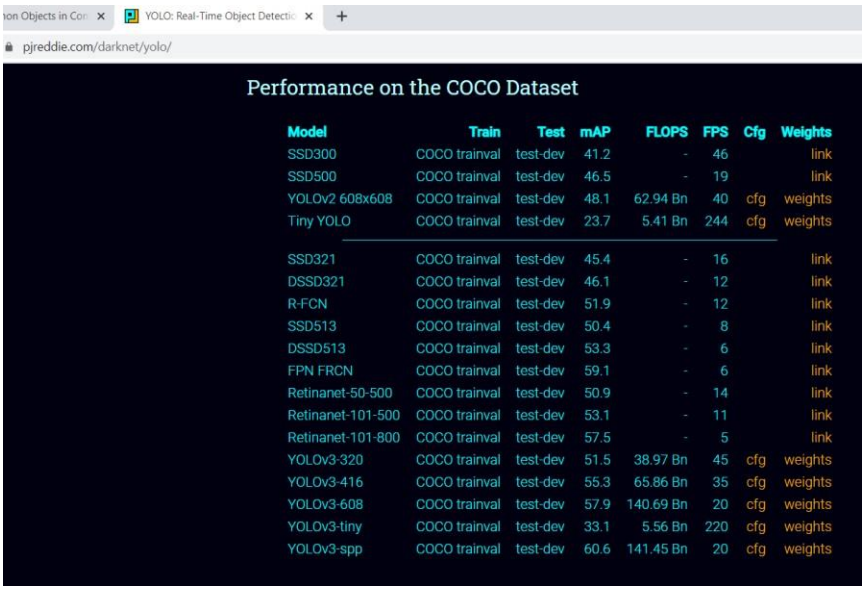
Untuk menguji aplikasi program maka diperlukan beberapa aplikasi yang sudah terinstall maupun tersedia sebagai berikut:

1. File YOLOV3 Config dan YOLOV3 Weight
2. Python 3 Idle dan beberapa library pendukung salah satunya OpenCV
3. XAMPP

5.2.2 Tahapan Pengujian Sistem

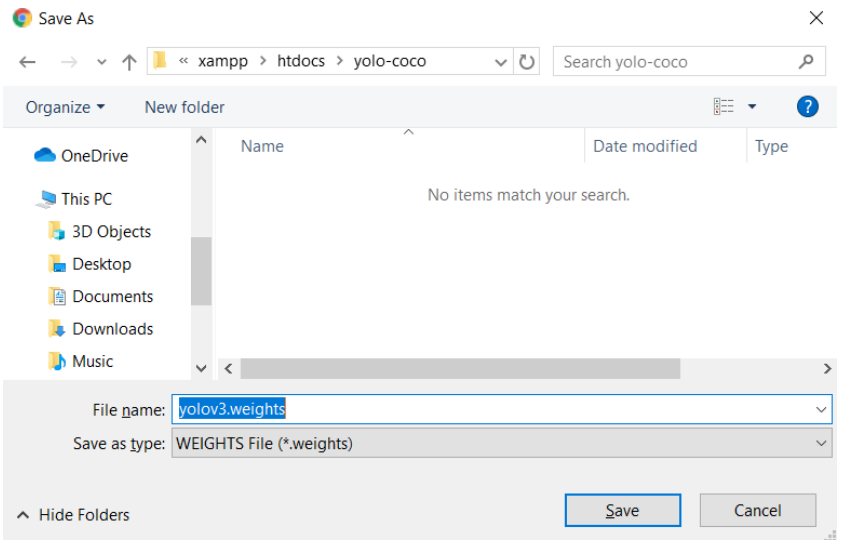
Langkah-langkah yang dilakukan untuk pengujian ini adalah sebagai berikut:

1. Langkah pertama yang harus di lakukan ialah menyiapkan file Model YOLOv3 dan *File Configuration* YOLOv3. File-file YOLOv3 ini bisa di Unduh di situs resmi YOLO, https://pjreddie.com/darknet/yolo/

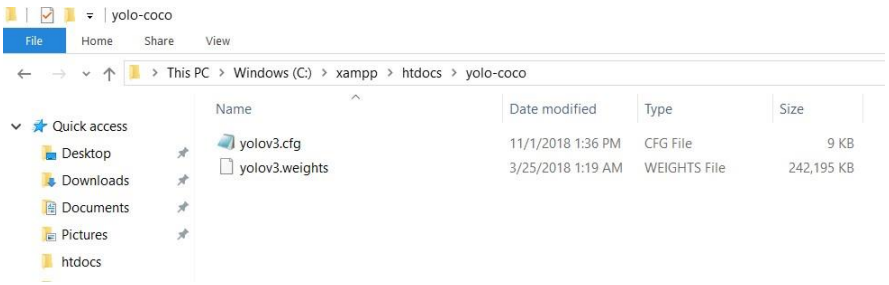


Gambar 5.12: Download File YOLOv3

2. Download file *weights* dan *cfg*. Pada pengujian ini menggunakan versi YOLOv3 416, arahkan lokasi penyimpanan kedua file tersebut ke folder *root server* localhost *x:\xampp\htdocs* tambah folder baru dengan nama yolo-coco

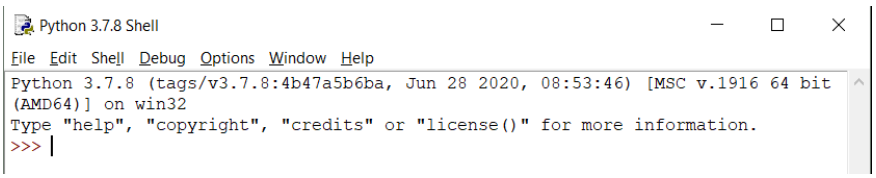


Gambar 5.13: Simpan File YOLOv3



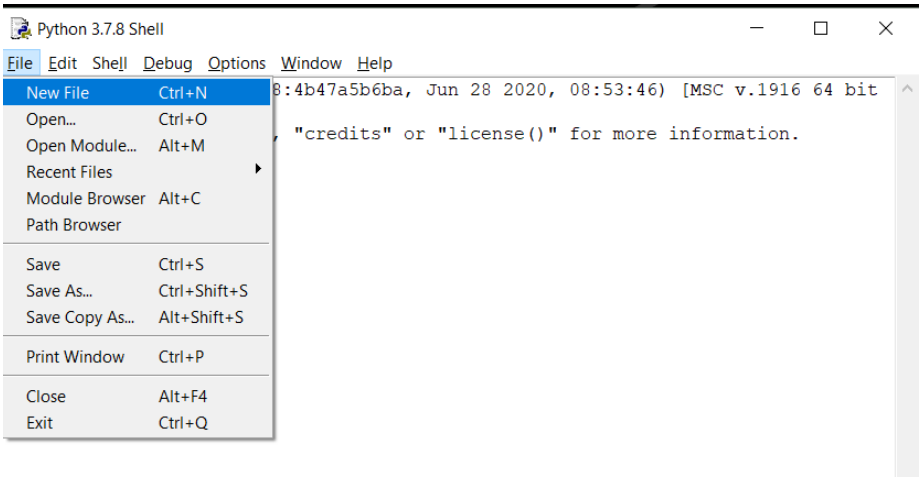
Gambar 5.14: File YOLOv3 Yang Telah Didownload

3. Setelah selesai download File YOLOv3, lalu download *dataset* label COCO di link *https://github.com/pjreddie/darknet/blob/master/data/coco.names* dan simpan file *coco.names* di folder yang sama dengan file YOLOv3.
4. Langkah selanjutnya ialah membuat program menggunakan bahasa pemrograman *Python.* Jalankan IDLE python 3



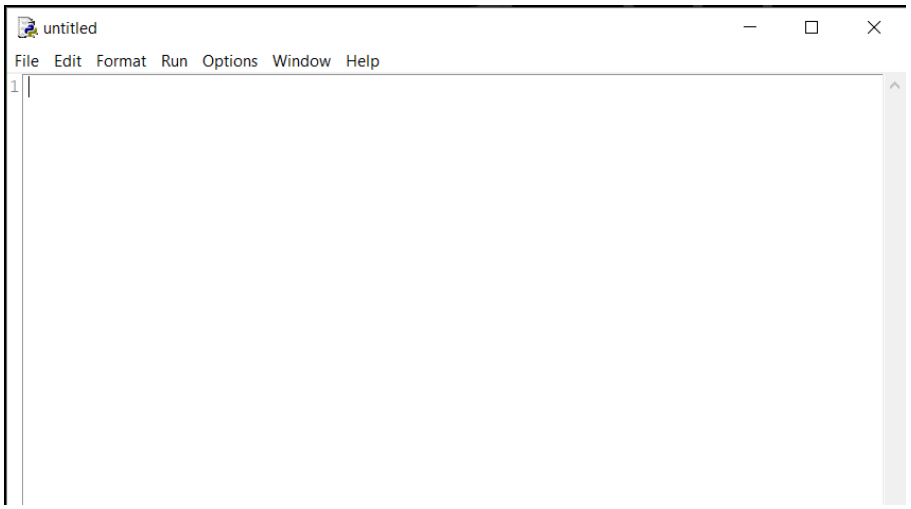
Gambar 5.15: Tampilan Awal Idle Python

5. Buka jendela file baru dengan klik File kemudian New File



Gambar 5.16: Membuat File Python Baru

6. Pada jendela baru yang muncul ini digunakan untuk menulis coding program sampai selesai



Gambar 5.17: Jendela Baru Python

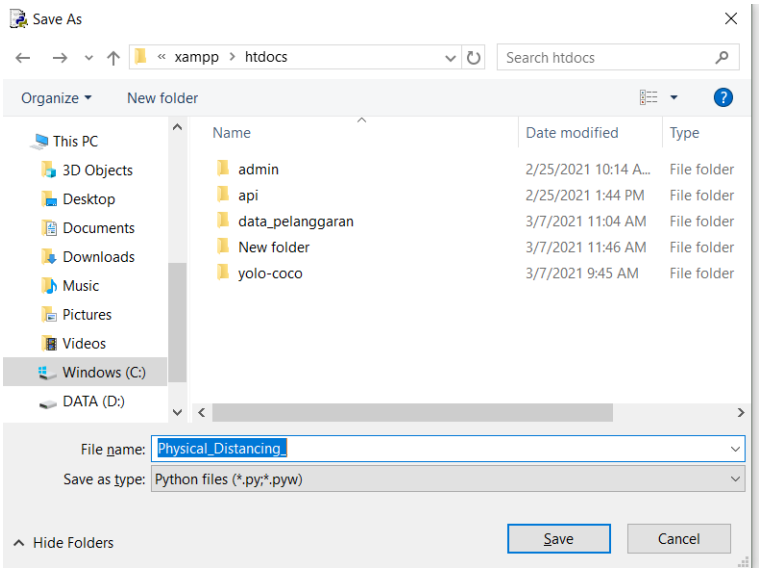
untitled

File Edit Format Run Options Window Help

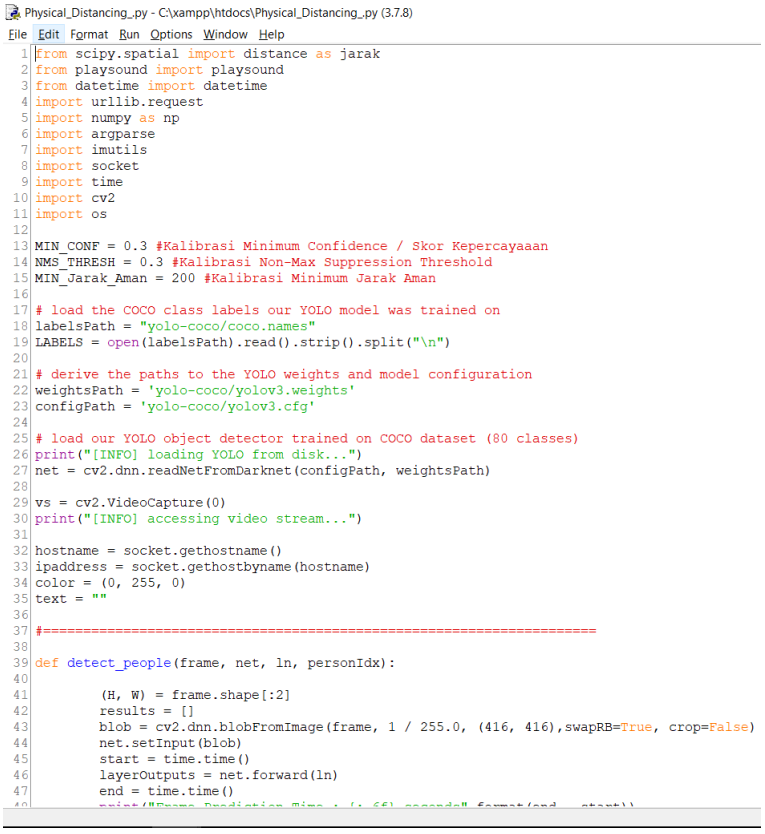
```
from scipy.spatial import distance as jarak
from playsound import playsound
from datetime import datetime
import urllib.request
import numpy as np
import argparse
import imutils
import socket
import time
import cv2
import os

MIN_CONF = 0.3 #Kalibrasi Minimum Confidence / Skor Kepercayaaan
NMS_THRESH = 0.3 #Kalibrasi Non-Max Suppression Threshold
MIN_Jarak_Aman = 200 #Kalibrasi Minimum Jarak Aman

# load the COCO class labels our YOLO model was trained on
labelsPath = "yolo-coco/coco.names"
LABELS = open(labelsPath).read().strip().split("\n")

# derive the paths to the YOLO weights and model configuration
weightsPath = 'yolo-coco/yolov3.weights'
configPath = 'yolo-coco/yolov3.cfg'

# load our YOLO object detector trained on COCO dataset (80 classes)
print("[INFO] loading YOLO from disk...")
net = cv2.dnn.readNetFromDarknet(configPath, weightsPath)

vs = cv2.VideoCapture(0)
print("[INFO] accessing video stream...")

hostname = socket.gethostname()
ipaddress = socket.gethostbyname(hostname)
color = (0, 255, 0)
text = ""

#========================================================================

def detect_people(frame, net, ln, personIdx):

        (H, W) = frame.shape[:2]
        results = []
        blob = cv2.dnn.blobFromImage(frame, 1 / 255.0, (416, 416),swapRB=True, crop=False)
        net.setInput(blob)
        start = time.time()
        layerOutputs = net.forward(ln)
        end = time.time()
```

Gambar 5.18: Penulisan Coding Program

7. Setelah selesai menuliskan coding program, selanjutnya simpan file coding program yang telah kita buat dan arahkan lokasi penyimpanan pada folder *root server* localhost *x:\xampp\htdocs* dengan nama file *Physical_Distancing_.py* perhatikan ekstensi file harus *.py* (*python*).

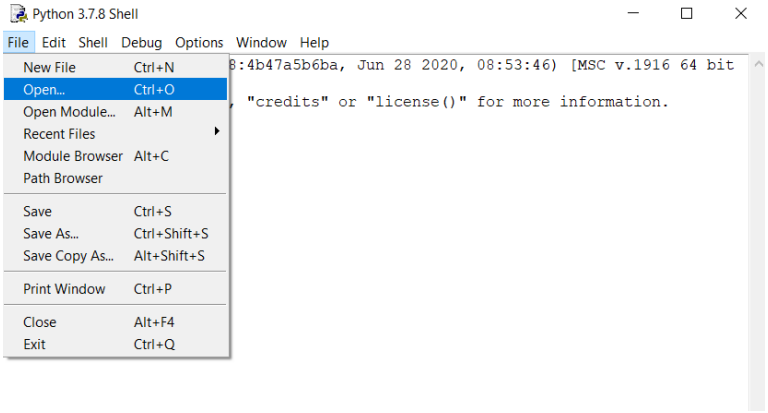


Gambar 5.19: Menyimpan File Coding Program Python



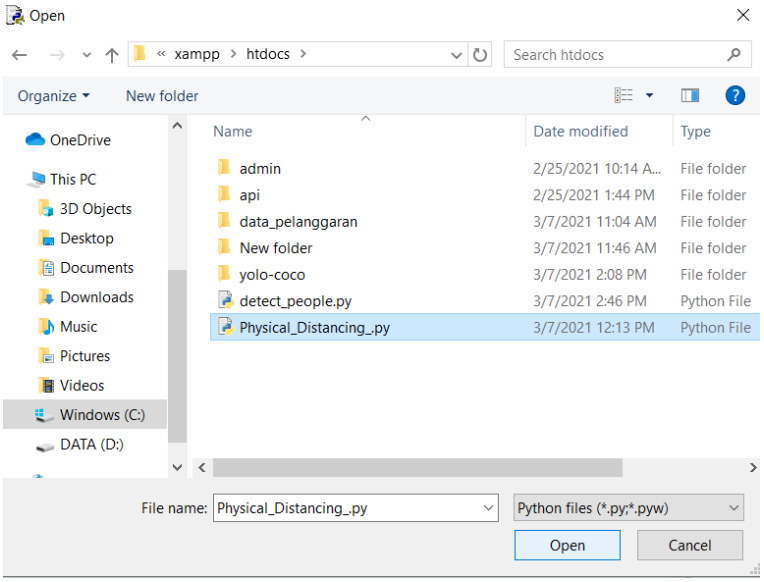
Gambar 5.20: Tampilan Setelah Coding Program Disimpan

Berikut merupakan cara untuk membuka progam yang telah tersimpan atau menjalankan program yang telah dibuat, file harus berformat .py agar bisa dibuka pada *Idle Python*, pada tampilan awal *python* klik file kemudian pilih *open* :



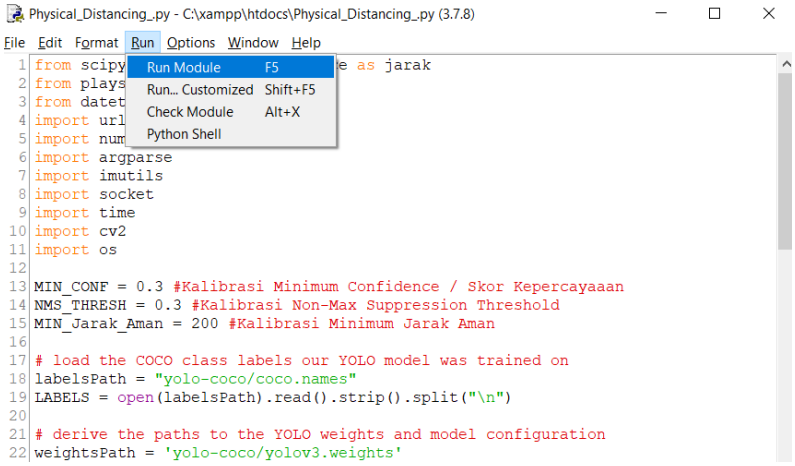
Gambar 5.21: Tampilan Membuka *Project Python*

Kemudian pilih file *project python* yang ingin di buka atau yang ingin dijalankan. *File project* yang dapat dibuka hanya yang berekstensi *.py*. Setelah file dipilih klik tombol *Open*.



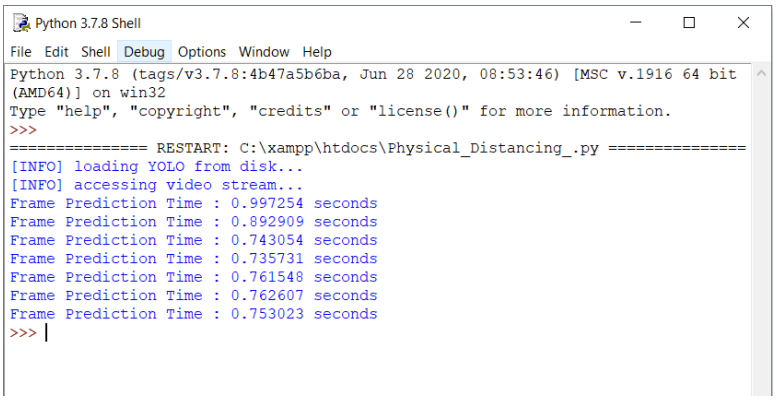
Gambar 5.22: Tampilan Memilih *File Python*

Setelah *project python* terbuka, selanjutnya dapat diedit terlebih dahulu atau langsung di jalankan dengan cara pilih tab menu Run, lalu pilih Run Module. Cara lain untuk menjalankan *project python* juga dapat menggunakan tombol shorcut F5.

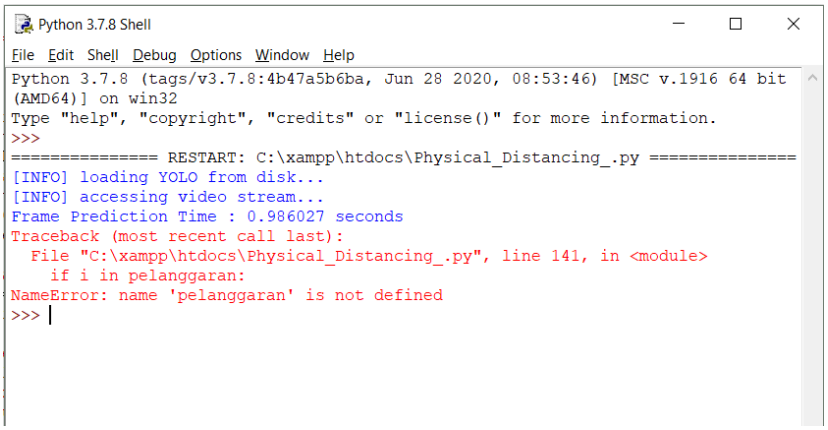


Gambar 5.23:Tampilan Menjalankan *Project Python*

Setelah *project python* di jalankan *Idle Python* akan secara otomatis menampilkan *python shell* yang mana *python shell* ini berfungsi sebagai *interface* yang menampilkan proses yang sedang berjalan dari hasil program yang telah kita buat. *Python shell* juga akan menampilkan peringatan apabila terdapat error pada program yang kita jalankan.



Gambar 5.24:Tampilan Program Tidak Terdapat *Error*



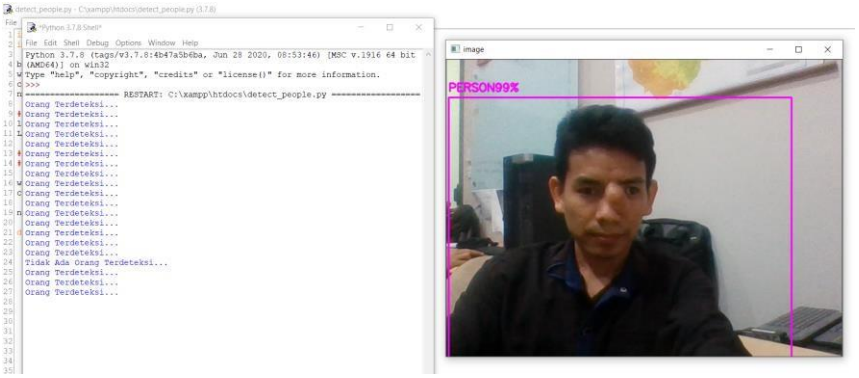
Gambar 5.25:Tampilan Program Ketika Terdapat *Error*

5.2.3 Pengujian Program *detect_people.py*

Pengujian ini bertujuan untuk menguji apakah program yang telah dibuat sesuai dengan yang di inginkan, kegunaan preproses ini adalah untuk mengetahui apakah program dapat mendeteksi keberadaan manusia.

```
def detect_people(outputs, img):
    hT, wT, cT = img.shape
    bbox = []
    classIds = []
    confs = []

    for output in outputs:
        for det in output:
            scores = det[5:]
            classId = np.argmax(scores)
            confidence = scores[classId]
            if confidence > confThreshold:
                w,h = int(det[2]*wT), int(det[3]*hT)
                x,y = int((det[0]*wT)-w/2), int((det[1]*hT)-h/2)
                bbox.append([x,y,w,h])
                classIds.append(classId)
                confs.append(float(confidence))


    indices = cv2.dnn.NMSBoxes(bbox,confs, confThreshold, nmsThreshold)

    for i in indices:
        i = i[0]
        box = bbox[i]
        x,y,w,h = box[0], box[1], box[2], box[3]
        cv2.rectangle(img,(x,y),(x+w,y+h),(255,0,255),2)
        cv2.putText(img,f'{LABELS[classIds[i]].upper()}{int(confs[i]*100)}%',
                    (x,y-10),cv2.FONT_HERSHEY_SIMPLEX,0.6,(255,0,255),2)

        if LABELS[classIds[i]] == "person":
            print("Orang Terdeteksi...")

        else:
            print("Tidak Ada Orang Terdeteksi...")
```

Gambar 5.26: Potongan Coding Program *detect_people.py*



Gambar 5.27: Hasil dari Program *detect_people.py*

5.2.4 Pengujian Program *Physical_Distancing_.py*

Pengujian ini bertujuan untuk menguji apakah program yang dibuat dengan menggunakan metode YOLO *Object Detection* sesuai dengan yang diinginkan yaitu dapat mendeteksi pelanggaran *physical distancing* (jarak fisik) setiap orang yang terdeteksi oleh kamera.

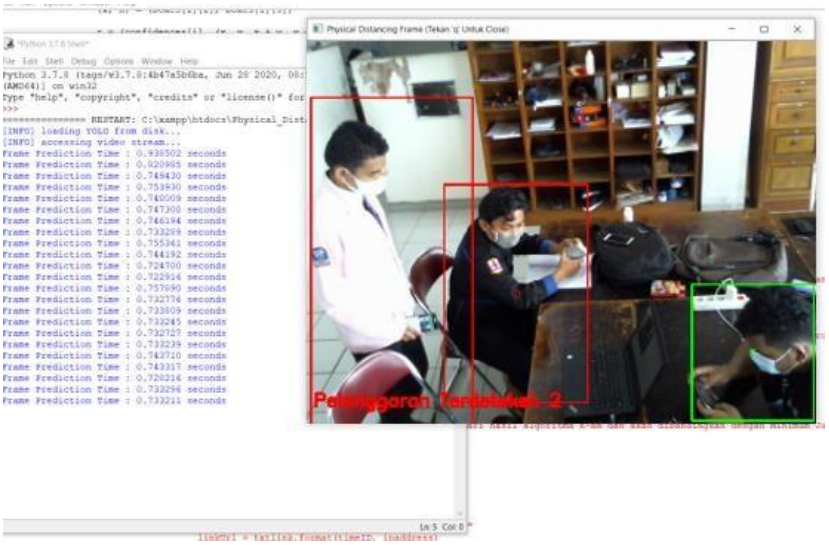
1. Langkah awal dalam pengujian program physical distancing adalah melakukan kalibrasi *MIN_CONF*, *NMS_THRESH*, dan *MIN_Jarak_Aman* (nilai 100-200 untuk jarak aman kurang lebih 1 meter) yang bisa diatur pada program agar mendapatkan hasil yang diinginkan.

Physical_Distancing_.py - C:\xampp\htdocs\Physical_Distancing_.py (3.7.8)

e Edit Format Run Options Window Help

```
from scipy.spatial import distance as jarak
from playsound import playsound
from datetime import datetime
import urllib.request
import numpy as np
import argparse
import imutils
import socket
import time
import cv2
import os

MIN_CONF = 0.3 #Kalibrasi Minimum Confidence / Skor Kepercayaaan
NMS_THRESH = 0.3 #Kalibrasi Non-Max Suppression Threshold
MIN_Jarak_Aman = 200 #Kalibrasi Minimum Jarak Aman
```

Gambar 5.28: Tampilan Kalibrasi Pendeteksian



Gambar 5.29: Hasil Pendeteksian Pelanggaran *Physical Distancing*

5.3 PENGUJIAN PERANGKAT KERAS

Pengujian perangkat keras ini dilakukan untuk mengetahui apakah perangkat yang digunakan seperti *webcam* dan *speaker* sudah berjalan dengan semestinya sesuai dengan perintah yang di program, apakah *webcam* dapat mendeteksi objek manusia atau tidak, apakah *speaker* akan memberikan peringatan jika *webcam* mendeteksi adanya pelanggaran jaga jarak, disini juga akan diuji waktu respon *webcam* terhadap objek dalam beberapa kali percobaan.

5.3.1 Pengujian Deteksi Objek Manusia Dan Waktu Respon

Pengujian ini menguji sistem apakah sistem dapat mendeteksi objek manusia pada jarak tertentu serta waktu respon perpindahan objek dari setiap percobaan. Pengujian dapat dilihat pada tabel 5.1:

Tabel 5.1: Pengujian Deteksi Objek Manusia Dan Waktu Respon

Banyak percobaan	Akurasi deteksi objek manusia (%)			Rata-rata waktu respon perpindahan objek (detik)
	1m – 5m	5m – 10m	10m – 20m	
1	100	100	90	0.721
2	100	90	80	0.838
3	100	100	100	0.663
4	100	100	90	0.825
5	100	90	90	0.846
6	100	100	90	0.789
7	100	90	80	0.707
8	100	100	100	0.770
9	100	100	90	0.629
10	100	90	80	0.830
Rata - rata	**100**	**96**	**89**	**0.761**

Pada Tabel 5.1 dapat dilihat rata – rata perbandingan dari hasil 10x percobaan menunjukkan bahwa hanya pada jarak 1 – 5 meter tingkat akurasi mencapai 100% untuk pendeteksian objek manusia. Dan waktu respon perpindahan jarak objek rata – rata mencapai 0,761 detik.

5.3.2 Pengujian *Speaker* Saat Mendeteksi Ada Pelanggaran

Pengujian ini menguji apakah sistem memberikan peringatan berupa *output* suara jika sistem mendeteksi ada pelanggaran *physical distancing* . Pengujian dapat dilihat pada tabel 5.2:

Tabel 5.2: Pengujian Output Suara

No	Percobaan	Aman	Pelanggaran	Respon Speaker	Kesimpulan
1	1	YA	-	TIDAK	Baik
2	2	YA	-	TIDAK	Baik
3	3	-	YA	YA	Baik
4	4	-	YA	YA	Baik
5	5	YA	-	TIDAK	Baik
6	6	-	YA	YA	Baik
7	7	-	YA	YA	Baik
8	8	-	YA	YA	Baik
9	9	-	YA	YA	Baik
10	10	-	YA	YA	Baik
11	11	YA	-	TIDAK	Baik
12	12	YA	-	TIDAK	Baik

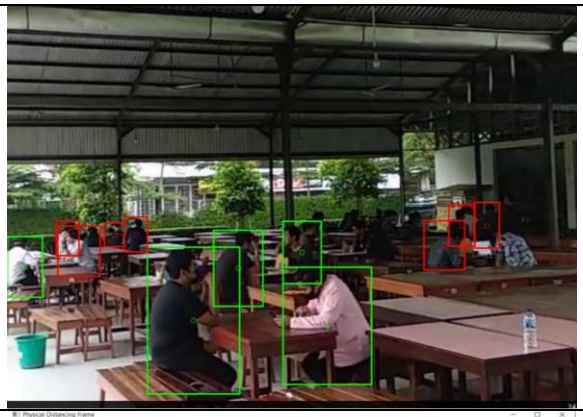
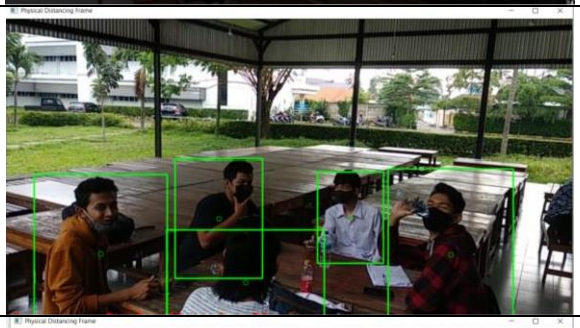
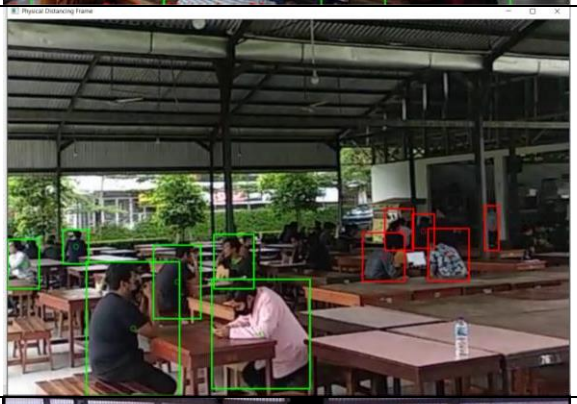
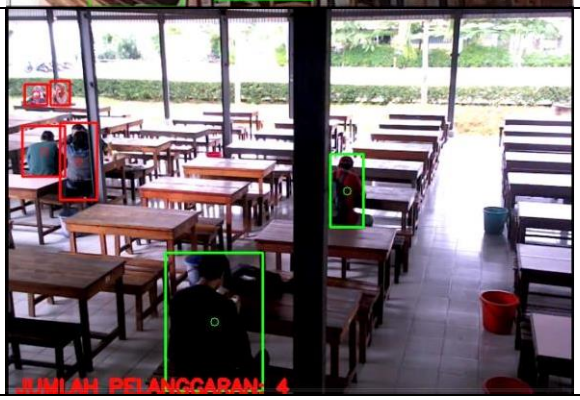
Dari hasil pengujian pada tabel 5.2 diatas telah dilakukan pengujian sebanyak 12X percobaan dimana mendapatkan hasil yang sangat baik. Speaker hanya akan berbunyi ketika terdeteksi adanya pelanggaran dan diam ketika terdeteksi aman.

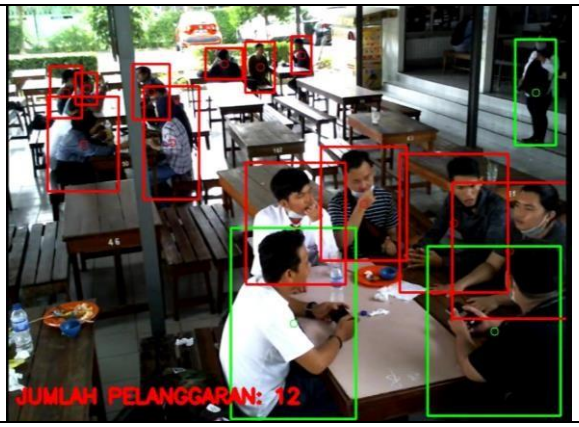
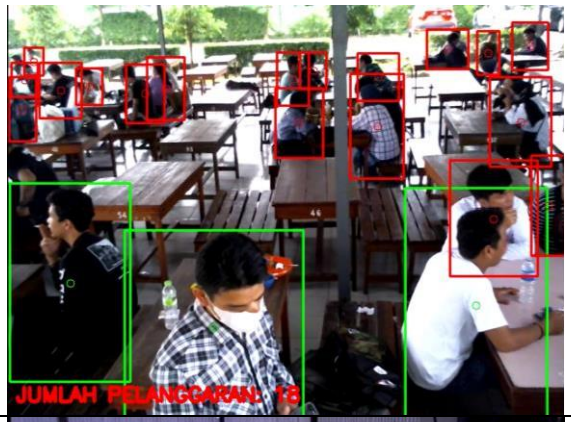
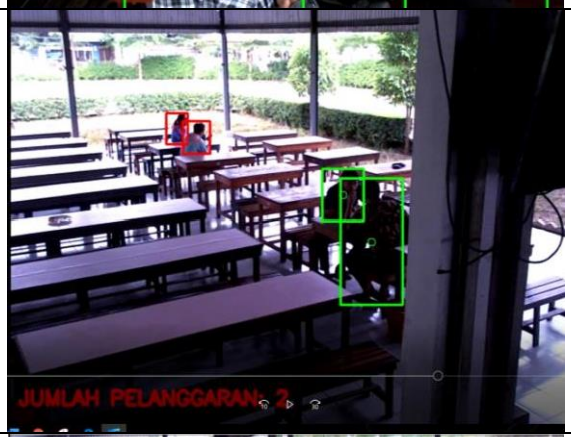
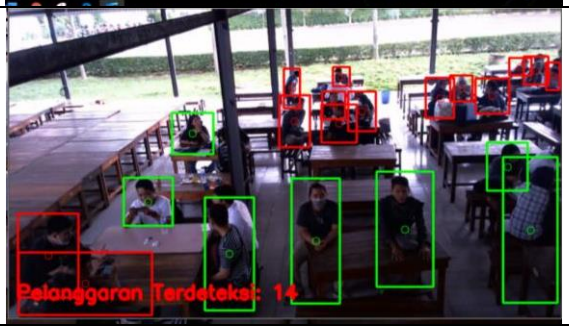
5.3.3 Hasil Pengujian Deteksi Pelanggaran di Kantin UNAMA

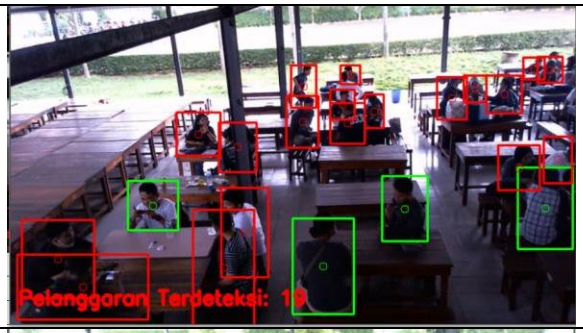
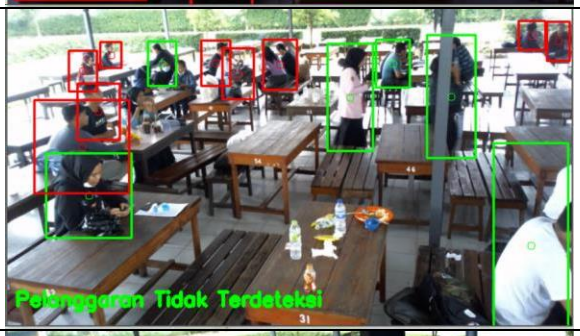
Berdasarkan hasil dari pengujian pada video *webcam* yang diletakkan di kantin kampus UNAMA. Dapat ditunjukan pada tabel 5.3 hasil pengujian. Ada 8 sudut pandang kamera yang berbeda, dimana terdapat pula kualitas video yang berbeda-beda salah satunya pada pencahayaan. Kualitas video yang dihasilkan dari *webcam*, dapat mempengaruhi tabel hasil pengujian dalam mendeteksi objek manusia. Sebagai contoh, jika warna baju yang digunakan oleh manusia itu sama dengan warna background yang berada dibelakangnya. Maka, sulit untuk mengidentifikasi objek manusia itu sendiri. Begitu juga, dengan objek manusia yang terhalangi oleh objek lain pada *webcam*. Serta objek manusia yang duduk dilokasi yang minim pencahayaaan akan sulit terindentifikasi.

Pada tabel 5.3 di bawah ini dapat dilihat, objek manusia yang terbaca dengan jelas akan terbounding. Sedangkan objek manusia yang tidak terbaca secara jelas tidak terbounding. Objek manusia yang terbouding ini akan diproses dengan jarak antar bounding yang terdekat. Apabila jarak kurang dari angka *MIN_Jarak_Aman* yang telah di kalibrasi (nilai 100-200 untuk jarak aman kurang lebih 1 meter), maka bounding tersebut akan berwarna merah. Dan value yang dihasilkan akan terhitung sebagai pelanggaran *physical distancing*. Dalam pengujian yang dilakukan ini menggunakan kalibrasi *MIN_Jarak _Aman = 100, MIN_CONF = 0.3 dan NMS_THRESH = 0.3.*

Tabel 5.3: Hasil Sample Dari Potongan Video Webcam di Kantin UNAMA

No	Hasil Pendeteksian	Tidak Terdeteksi	Terdeteksi Aman	Terdeteksi Pelanggaran
1		3	5	7
2		0	5	0
3		4	6	5
4		1	2	4

5		3	3	12
6		4	4	15
7		0	2	2
8		3	7	14

9		0	4	19
10		3	6	9
11		2	3	7
12		3	1	20

Tabel 5.4: Hasil Persentase Pengujian

Sample Dari Potongan Video	Persentase Pendeteksian Objek Manusia	Persentase Pendeteksian Jarak Pelanggaran
1	86.6	71.4
2	100	100
3	73.3	40
4	85.7	100
5	83.3	83.3
6	82.6	73.3
7	100	0
8	87.5	85.8
9	100	73.6
10	83.3	77.7
11	83.3	75
12	87.5	80
Rata-rata	**87.75**	**71.67**

Berdasarkan hasil persentase pada Tabel 5.4 dapat dilihat bahwa sistem dapat mendeteksi objek manusia dan dapat mendeteksi jarak pelanggaran antar objek manusia yang terdeteksi berdasarkan hasil 12 sample gambar yang diambil dari potongan video *webcam* di kantin UNAMA pada Tabel 5.3, didapatkan rata-rata error terhitung sebesar 12.25% dan 28.33%. Dengan pengujian inilah, bahwa sistem yang telah dirancang mendapatkan hasil akurasi sebesar 87.75% untuk mendeteksi objek manusia dan hasil akurasi sebesar 71.67% untuk mendeteksi jarak pelanggaran antar objek manusia.

5.4 ANALISA SISTEM SECARA KESELURUHAN

Untuk mendeteksi apabila terjadi kesalahan setelah uji coba, maka perlu dilakukan analisa secara keseluruhan mulai dari pengujian pada *webcam, speaker, website,* aplikasi android maupun program. Dari seluruh proses yang telah dilakukan, baik pengujian perangkat keras maupun perangkat lunak, dapat dikatakan bahwa alat ini dapat berfungsi sebagaimana yang penulis inginkan.

Namun masih ada beberapa masalah dan kekurangan pada sistem yang telah dirancang, sistem hanya bisa mendapati ukuran jarak yang akurat bila objek terdeteksi secara horizontal jika objek terdeteksi secara vertical dihadapan camera maka objek akan terdeteksi pelanggaran. Objek yang berkerumun masih sulit untuk terdeteksi per objek.

Analisa dilakukan untuk menunjukan bahwa sistem yang dirancang ini dapat bekerja sesuai dengan tujuan dari pembuatan. Analisa ini dilakukan dengan cara menguji sistem yang telah di program pada kondisi yang sebenarnya, proses pengujian dilakukan dengan cara sebagai berikut :

1. Memproses semua program mulai dari Pre-proses, kemudian *training* menggunakan metode YOLOv3 kemudian dijalankan dengan memasangkan *webcam* pada laptop melalui kabel USB.
2. Lalu jalankan program, *webcam* akan mulai mendeteksi keberadaan objek.
3. Selanjutnya *Webcam* akan mendeteksi secara *real time,* dan sistem akan melakukan *object tracking* dengan menampilkan *bounding box* pada objek manusia yang terdeteksi. Titik koodinat yang terdapat ditengah *bounding box* diproses menggunakan teori *Euclidean Distance*, apabila nilai *Distance*

yang di dapat lebih kecil dari nilai pengaturan *MIN_Jarak_Aman* yang telah diatur pada variabel program, maka sistem akan mendeteksi adanya pelanggaran.

4. Kemudian sistem akan memberikan peringatan berupa suara melalui speaker, lalu sistem akan men-*capture* pelaku pelanggaran tersebut dan hasil *capture* disimpan difolder *root server* localhost *x:\xampp\htdocs\data_pelanggaran*.
5. Terakhir sistem akan melakukan *request* ke URL *website physical distancing* untuk mengupdate *database* pelanggaran secara realtime. Apabila database telah terupdate, maka petugas keamanan kantin UNAMA dapat melihat ada pelanggaran *Physical Distancing* yang terjadi melalui *Web Physical Distancing* ataupun aplikasi *Physical Distancing Android*.