JCONES Vol. 11, No. 2 (2022) Hal: 57-62

Journal of Control and Network Systems

Situs Jurnal : http://jurnal.dinamika.ac.id/index.php/jcone

Sistem Deteksi Pelanggaran Jarak *Social Distancing* Dengan Transformasi *Bird's Eye View* Menggunakan Yolo-V3

Moamar Zahir Payudan [1] Heri Pratikno [2] Yosefine Triwidyastuti [3] Pauladie Susanto [4]
Program Studi/Jurusan Teknik Komputer
Universitas Dinamika.
Jl. Raya Kedung Baruk 98 Surabaya, 60298
Email: 1) 18410200056@dinamika.ac.id 2) heri@dinamika.ac.id 3) yosefine@dinamika.ac.id 4) pauladie@dinamika.ac.id

Abstrak: Salah satu protokol menjaga jarak merupakan upaya pencegahan virus COVID-19 yang masih sering diabaikan masyarakat. Berdasarkan permasalahan tersebut diatas, maka penulis membuat sistem yang mampu mendeteksi pelanggaran jarak *social distancing* dengan transformasi *bird's eye view.* Proses deteksi pelanggaran jarak *social distancing* menggunakan *input* video dengan metode YOLO-V3. Pelacakan objek manusia menggunakan kamera dengan *perspective view images* 45º kemudian di transformasi menjadi *bird's eye view* 90º terhadap *ground plane.* Proses tranformasi ini dilakukan untuk mengetahui efektifitas program *perspective view images* 45º dan program *bird's eye view* 90º dalam mendeteksi pelanggaran jarak. Tingkat akurasi hasil dari penelitian ini dalam mendeteksi pelanggaran jarak berdasar visualisasi *perspective view images 45º* sebesar 81.9%, sedangkan 18.1% pelanggaran jarak *social distancing* tidak terdeteksi, dan dalam transformasi sistem *bird's eye view 90º* memiliki tingkat akurasi sebesar 70.1%, sedangkan 29.9% pelanggaran jarak *social distancing* tidak terdeteksi. Hal ini menunjukkan tingkat akurasi dari program *Perspective view Images* 45º lebih tinggi dibandingkan program *Perspective Bird's Eye View* 90º. Tingginya nilai *error* pada program *Perspective Bird's Eye View* 90º dipengaruhi oleh sistem yang mendeteksi objek selain manusia, sehingga sistem menghitung dan memproses dengan jumlah data pelanggaran yang lebih banyak maupun lebih sedikit dibandingkan dengan data asli atau data perhitungan manual.

Kata Kunci: YOLO-V3, *Social Distancing*, *bird eye view*, COVID-19

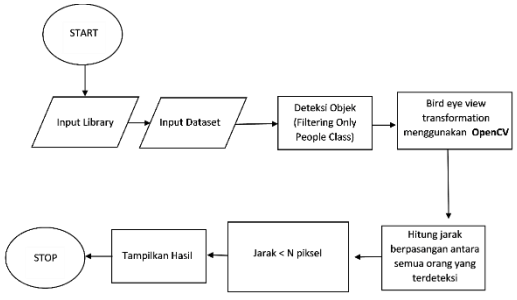
Protokol menjaga jarak merupakan upaya pencegahan virus Covid-19 yang masih sering di abaikan masyarakat saat beraktivitas, Menurut Kementerian Kesehatan – RI perkembangan kasus COVID-19 pada Bulan Maret 2022 mencapai 155.977 kasus aktif dan 154.343 kasus meninggal dunia. Dari permasalahan ini, penulis membuat sistem Pelanggaran Jarak *Social Distancing* Dengan Transformasi *Bird's Eye View* menggunakan *input* video dengan metode YOLO-V3. Proses Deteksi Jarak Kerumunan digunakan dalam rangka peningkatan dan penegakan protokol Kesehatan selama masa pandemi. Sistem deteksi pada proposal ini dapat dilakukan secara *real-time* kamera maupun deteksi dari *input* video menggunakan *Deep Learning* dengan *Python OpenCV* dan *Tensorflow*. YOLO-V3 merupakan metode atau teknologi jaringan syaraf pintar yang berfungsi untuk melakukan deteksi secara *real-time* dan dapat digunakan untuk pendeteksian objek, dengan kemampuan pendeteksian yang cepat dan akurat hingga dua kali lipat dibandingkan beberapa metode lain. YOLO-V3 memiliki arsitektur yang sederhana yaitu jaringan saraf *convolutional*. dengan metode YOLO-V3 pelacakan objek manusia dari *Perspective view Images 45º* dan *bird eye view 90º* terhadap *Ground plane* yang mempunyai keunggulan dan memungkinkan untuk mendapatkan representasi jarak antar manusia yang lebih akurat dengan *OpenCV* dan *Tensorflow*.

Pada penelitian sebelumnya, Fadhilah Mileanasari, Fia Anisa, Mita Sofi Abdillah dan Novendra Setyawan (2020) membuat sistem menggunakan YOLO-V3 sebagai deteksi objek, gambar, dan pelacakan objek manusia dari satu *Perspective view Images* 45º terhadap *Ground Plane*, dengan menggunakan *input* video CCTV Dinas Perhubungan dan dari *website live* CCTV

Kota Yogyakarta, yang mendeteksi jarak antar manusia, dengan hasil akurasi sebesar 63.91% untuk mendeteksi objek manusia dan hasil akurasi sebesar 84.69% untuk mendeteksi jarak antar manusia. Penelitian dari Oktaviani Ella Karlina dan Dina Indarti (2019) membuat sistem Pengenalan Objek Makanan Cepat Saji Pada Video Dan *Real Time Webcam* Menggunakan Metode *You Look Only Once* (Yolo), dengan hasil pengujian yang dilakukan menggunakan video dan *real time webcam*, objek pada citra makanan cepat saji berhasil dikenali dengan akurasi 63% sampai 100%, sedangkan penelitian dari Mawaddah Harahap, Juni Elfrida, Pasrah Agusman, Mario Rafae, Rahul Abram, Kiki Andrianto (2019) membuat Sistem Cerdas Pemantauan Arus Lalu Lintas Dengan YOLO (*You Only Look Once v3*), dengan hasil dapat mengklasifikasikan kendaraan dengan mAP(*mean Average Precision*) pada CCTV *Fix* yang paling tertinggi yaitu 97%, sedangkan pada CCTV PTZ adalah 99%.

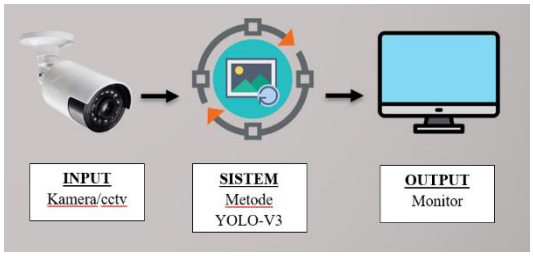
METODE PENELITIAN

Dataset yang digunakan dalam Deteksi Pelanggaran Jarak *Social Distancing* ini adalah dataset yang dapat diakses oleh semua orang melalui website *cocodataset.org*, dimana dataset ini memiliki jumlah 120.000 gambar dengan total 880.000 objek berlabel, sedangkan gambar manusia terdapat 66.808 gambar. Model-model ini di Training untuk memuat 80 jenis objek berbeda dalam kumpulan data ini.



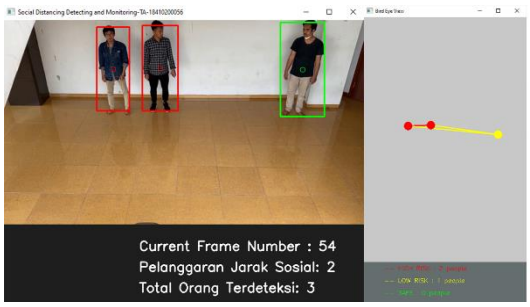
Gambar 1. Flowchart sistem deteksi

Algoritma program Sistem Deteksi Pelanggaran Jarak *Social Distancing* yang diajukan pada penelitian ini. Proses dimulai dari Start, lalu meng*input*kan *library*. Selanjutnya menginputkan dataset yang telah disediakan yaitu YOLO-COCO dataset. Setelah itu proses Deteksi Objek (*Filtering Only People Class*) menggunakan *OpenCV People Counter*, dan ketika objek sudah didapat proses dilanjutkan untuk mencari 4 titik sudut denah dan diterapkan transformasi perspektif menggunakan *OpenCV getPerspective Transform*. Kemuadian proses hitung jarak berpasangan antara semua orang yang terdeteksi dimulai dengan melihat nilai jarak lebih kecil atau lebih besar dari N (*pixel*), jika nilai jarak lebih kecil dari nilai N (*pixel*), maka warna kotak pembatas (*bounding box*) berubah menjadi merah yang artinya terdeteksi pelanggaran *Social Distancing*. Pada saat menampilkan hasil muncul tampilan layar dari *Perspective view Images 45º* dan *bird eye view 90º* dengan bidang bumi (*ground plane*).



Gambar 2. Model perancangan *Hardware*

Berdasarkan gambar 2 model perancangan *hardware* dapat diketahui bahwa *input* data berasal dari kamera atau CCTV. *Input* tersebut diproses oleh sistem dengan menggunakan metode YOLO-V3, dengan tujuan memproses video yang berasal dari kamera atau CCTV tersebut untuk mendeteksi object manusia. Object manusia yang sudah terdeteksi oleh sistem diproses kembali oleh sistem untuk mendeteksi jarak antara manusia pada video yang di uji, sehingga sistem mampu mendeteksi pelanggaran jarak *Social Distancing*. *Output* yang ditampilkan pada monitor adalah hasil perhitungan dan visualisasi deteksi jumlah pelanggaran *Social Distancing* dari sistem *Perspective view Images* 45º dan *Bird's Eye View 90º*.



Gambar 3. Tampilan *output* pada layar *monitor*

Dilihat dari gambar 3 tampilan *output* pada layar monitor di atas, ditampilkan visualisasi sistem, perhitungan jumlah orang yang melakukan pelanggaran jarak *Social Distancing* dan jumlah

frame dalam video. Proses deteksi jarak dilakukan dengan mendeteksi keberadaan orang pada setiap *frame.* Ketika orang tidak melakukan pelanggaran jarak maka *bounding box* berwarna hijau atau kuning, tetapi jika orang melakukan pelanggaran jarak, maka *bounding box* berubah warna menjadi merah, semua proses tersebut kemudian di transformasikan dalam bentuk sistem *Bird's Eye View 90º*.

```
# ambang batas penekanan non-maksima suprecion
NMS_THRESH = 0.3

# probabilitas minimum untuk menyaring deteksi sumur
MIN_CONF = 0.3

# jarak aman minimum (dalam piksel)
# antara dua orang yang terdeteksi
MIN_DISTANCE = 150

# jalur string ke model
MODEL_PATH = 'yolo-coco'

# apakah menggunakan gpu (dengan CUDA) atau tidak
USE_GPU = False

# ukuran bingkai keluaran
SIZE_FRAME = 720
```

Gambar 4. Konfigurasi *pixel*

Dalam program *perspective view Images 45º* terdapat *function* yang menghitung jarak antar orang berdasarkan *pixel* sebesar 50, 75 dan 150 *pixel* terhadap jarak sesungguhnya yaitu 2 meter, sehingga jika jarak antar orang kurang dari *pixel* yang telah ditentukan atau kurang dari 2 meter, maka *bounding box* berubah menjadi warna merah dengan indikator tampilan jumlah pelanggaran. Dalam menentukan *pixel* dapat dipengaruhi jarak antar kamera dengan *ground plane* atau bidang datar, semakin dekat jarak kamera dengan *ground plane* maka semakin besar nilai *pixel* yang dipakai.

```
if len(results) >= 2:
        # ekstrak semua centroid dari hasil dan hitung
        # Jarak Euclidean antara semua pasangan centroid
        centroids = np.array([r[2] for r in results])
        D = dist.cdist(centroids, centroids, metric="euclidean")

        # loop di atas segitiga atas dari matriks jarak
        for i in range(0, D.shape[0]):
            for j in range(i + 1, D.shape[1]):
                # periksa untuk melihat apakah jarak antara keduanya
                # pasangan centroid kurang dari nomor yang
dikonfigurasi
                # piksel
                if D[i, j] < config.MIN_DISTANCE:
                    # perbarui set pelanggaran kami dengan indeks
                    # pasangan centroid
                    violate.add(i)
                    violate.add(j)
```

Gambar 5. Program menghitung jarak antar orang

```
def get_distances(boxes1, bottom_points, distance_w, distance_h):

    distance_mat = []
    bxs = []

    for i in range(len(bottom_points)):
        for j in range(len(bottom_points)):
            if i != j:
                dist = cal_dis(bottom_points[i], bottom_points[j], distance_w, distance_h)
                #dist = int((dis*180/1100)/distance)
                if dist <= 150:
                    closeness = 0
                    distance_mat.append([bottom_points[i], bottom_points[j], closeness])
                    bxs.append([boxes1[i], boxes1[j], closeness])
                elif dist > 150 and dist <=180:
                    closeness = 1
                    distance_mat.append([bottom_points[i], bottom_points[j], closeness])
                    bxs.append([boxes1[i], boxes1[j], closeness])
                else:
                    closeness = 2
                    distance_mat.append([bottom_points[i], bottom_points[j], closeness])
                    bxs.append([boxes1[i], boxes1[j], closeness])

    return distance_mat, bxs
```

Gambar 6. Konfigurasi *Bird's Eye View 90º*

Dalam program transformasi *Bird's Eye View 90º* terdapat *function* yang menghitung jarak antar *centroid* atau titik yang telah ditransformasi tersebut, kemudian sistem menghitung jarak antar titik berdasarkan *pixel* sebesar 180 dan 1100 *pixel* terhadap jarak sesungguhnya yaitu 2 meter, sehingga jika jarak antar *centroid* kurang dari *pixel* yang ditentukan atau kurang dari 2 meter, maka *centroid* atau titik tersebut berubah menjadi warna merah dengan indikator tampilan jumlah pelanggaran. Dalam menentukan *pixel* dapat dipengaruhi jarak antar kamera dengan *ground plane* atau bidang datar, semakin dekat jarak kamera dengan *ground plane*, maka semakin besar nilai *pixel* yang dipakai.

Pengujian Deteksi Pelanggaran

Pengujian dari sistem deteksi yang meliputi dua parameter uji, yaitu: pengujian visualisasi deteksi pelanggaran jarak dari *Perspective view Images* 45º dan Tranformasi *Bird's Eye View 90º*, dan pengujian akurasi.

Tujuan Pengujian

Tujuan pengujian ini adalah untuk mengetahui apakah sistem dapat mendeteksi orang yang melakukan pelanggaran jarak *Social Distancing* dalam suatu *frame* sebuah video.

Prosedur Pengujian

Menjalankan program yang sudah dibuat menggunakan metode YOLO-V3 untuk mendeteksi orang dan pelanggaran menggunakan program *Visual Studio Code.* Ketika sekumpulan orang berada dalam video, maka setiap orang tersebut dideteksi oleh sistem melalui tampilan *bounding box*. Setiap detik perubahan *frame* pada video menghitung jarak *bounding box* yang berdekatan, jika jarak kurang dari nilai *pixel* yang ditentukan maka *bounding box* berubah menjadi

merah, nilai *pixel* ditentukan berdasarkan jarak antara kamera dengan *ground plane*.

Tranformasi *Bird's Eye View 90º* menggunakan *OpenCV getPerspective Transform* dengan menjalankan program yang berbeda, dimana 4 titik poin *frame* yang diuji ditentukan secara manual dan sistem otomatis memproses transformasi tersebut sampai video selesai, kemudian menghitung tingkat akurasi data yaitu mencari nilai *error* (%) seperti yang terdapat pada rumus (1), kemudian mencari nilai rata - rata *error* (%) yang terdapat pada rumus (2), sehingga untuk menentukan tingkat akurasi data menggunakan rumus (3).

$$Error\,(\%) = \frac{Error}{\text{PM}} \text{ X } 100\% \qquad (1)$$

$$\text{Rata – Rata } Error\,(\%) = \frac{\sum error\ (\%)}{\text{BP}} \qquad (2)$$

$$\text{Akurasi} = 100\% - \text{Rata – Rata } error\ (\%) \qquad (3)$$

UV: Uji Video ke-
BP: Banyaknya Percobaan
DW: Durasi Waktu
PM: Perhitungan Manual
TS: Terdeteksi Sistem
E1: Error
E2: Error (%)

HASIL DAN PEMBAHASAN

Tabel 1. Hasil uji *Perspective view Images 45º*

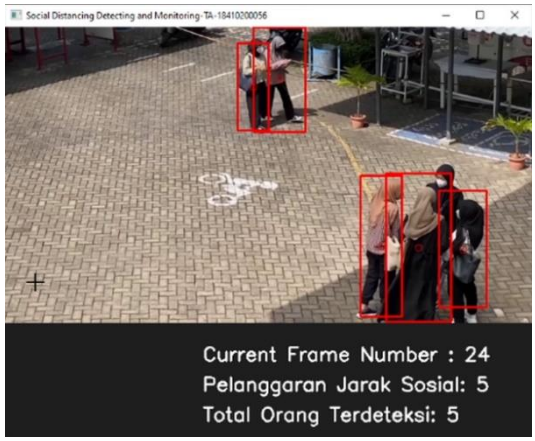
UV	BP	DW	PM	TS	E1	E2
1	1		3	3	0	0
	2	4	3	3	0	0
	3		3	3	0	0
2	1		9	10	1	11.11
	2	4	9	10	1	11.11
	3		9	10	1	11.11
3	1		8	13	5	62.5
	2	4	8	13	5	62.5
	3		8	13	5	62.5
4	1		5	5	0	0
	2	4	5	5	0	0
	3		5	5	0	0
5	1		6	5	1	16.66
	2	4	6	5	1	16.66
	3		6	5	1	16.66
Jumlah			93	108	21	270.81

$$\text{Rata – Rata } Error\,(\%) = \frac{270.81}{15}$$
$$= 18.1\%$$

$$\text{Akurasi} = 100\% - 18.1\%$$
$$= 81.9\%$$

Berdasarkan hasil pengujian pada tabel 1 hasil uji deteksi pelanggaran jarak *social distancing* dari program *Perspective view Images* 45º menunjukkan bahwa sistem dapat mendeteksi pelanggaran jarak *social distancing* dengan tingkat akurasi sebesar 81.9%, sedangkan 18.1% pelanggaran jarak *social distancing* tidak terdeteksi oleh sistem atau *error*.

Dari gambar 7 hasil deteksi *Perspective view Images* 45º sampel uji video ke-5 dalam *frame* ke-24, dapat dilihat bahwa sistem dapat mendeteksi orang yang melakukan pelanggaran sebanyak 5 orang, sedangkan pada perhitungan secara manual jumlah pelanggaran jarak sebanyak 6 orang, sehingga terdapat 1 orang yang melakukan pelanggaran tidak terdeteksi oleh sistem. Pada video tersebut, sistem menggunakan perbandingan skala 75 *pixel*: 2 meter.



Gambar 7. Visualisasi view Images 45º

Pada pengujian akurasi dari *Bird's Eye View 90º* ini juga dilakukan dengan 5 Video yang menampilkan aktivitas di tempat umum, dimana masing-masing video diuji sebanyak 3 kali percobaan, pengujian pada deteksi dari program *Bird's Eye View 90º* ini dihitung berdasarkan setiap *frame* pada video sampai video berakhir dengan perhitungan deteksi pelanggaran yang dihitung berdasarkan jumlah orang yang melanggar. Kemudian hasil deteksi dari visualisasi *Bird's Eye View 90º* dibandingkan dengan hasil perhitungan

secara manual di setiap detiknya sampai masing-masing video yang di uji selesai.

Tabel 2. Hasil uji *Perspective Bird's Eye View 90º*

UV	BP	DW	PM	TS	E1	E2
1	1		3	3	0	0
	2	4	3	3	0	0
	3		3	3	0	0
2	1		9	11	2	22.22
	2	4	9	12	3	33.33
	3		9	11	2	22.22
3	1		8	13	5	62.5
	2	4	8	14	6	75
	3		8	13	5	62.5
4	1		5	7	2	40
	2	4	5	7	2	40
	3		5	7	2	40
5	1		6	6	0	16.66
	2	4	6	6	0	16.66
	3		6	6	0	16.66
Jumlah			93	122	29	447.75

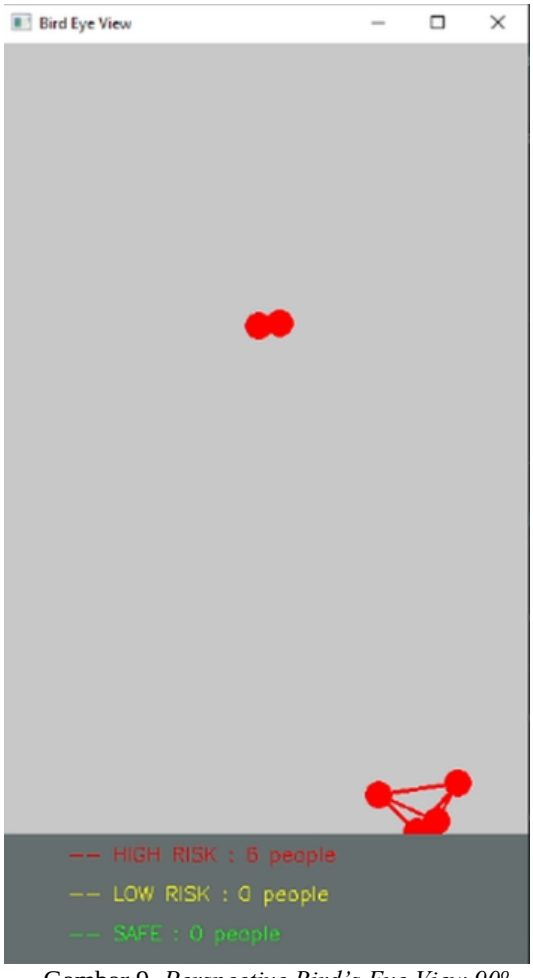
$$\text{Rata – Rata } \textit{Error}\,(\%) = \frac{447.75}{15} = 29.9\%$$

Akurasi = 100% - 29.9%
= 70.1%

Berdasarkan hasil pengujian pada tabel 2 hasil uji deteksi pelanggaran jarak *Social Distancing* dari *Perspective Bird's Eye View* 90º menunjukkan bahwa sistem dapat mendeteksi pelanggaran jarak *social distancing* dengan tingkat akurasi sebesar 70.1%, sedangkan 29.9% pelanggaran jarak *social distancing* tidak terdeteksi oleh sistem atau *error*. Hal ini menunjukkan bahwa tingkat akurasi dari program *Perspective view Images* 45º lebih tinggi dibandingkan program *Perspective Bird's Eye View* 90º, tingginya nilai *error* pada program *Perspective Bird's Eye View* 90º dipengaruhi oleh sistem yang mendeteksi objek selain manusia, sehingga sistem menghitung dan memproses data pelanggaran *social distancing* dengan jumlah yang lebih banyak dibandingkan dengan data asli atau data perhitungan manual.

Gambar 8. Proses 4 titik poin transformasi

Dari gambar 8 pada proses menentukan 4 titik poin transformasi dilakukan pada program *Bird's Eye View 90º*. proses ini dilakukan untuk menentukan lokasi objek kerumunan yang diuji pada video, sehingga proses transformasi terhadap sistem *Bird's Eye View 90º* berdasarkan lokasi 4 titik poin yang sudah ditentukan.



Gambar 9. *Perspective Bird's Eye View 90º*

KESIMPULAN

Dari hasil pengujian yang dilakukan oleh penulis untuk penelitian ini, didapatkan beberapa kesimpulan sebagai berikut:

1. Jarak deteksi pelanggaran *social distancing* pada program *perspective view images 45º* menggunakan 50, 75 dan 150 *pixel*, sedangkan pada program *bird's eye view 90º* menggunakan 180 dan 1100 *pixel*.
2. Jarak antara kamera dan *ground plane* mempengaruhi jarak deteksi pelanggaran *social distancing*, jika jarak antara kamera dan *ground plane* lebih jauh maka menggunakan 50 *pixel*, sedangkan jika jarak antara kamera dengan *ground plane* lebih dekat, maka menggunakan 75 *pixel* hingga 150 *pixel*.
3. Tingkat akurasi hasil dari penelitian ini dalam mendeteksi pelanggaran jarak *social distancing* berdasar visualisasi *perspective view images* 45º sebesar 81.9%, sedangkan 18.1% pelanggaran jarak *social distancing* tidak terdeteksi, dan dalam transformasi sistem *bird's eye view* 90º memiliki tingkat akurasi sebesar 70.1%, sedangkan 29.9% pelanggaran jarak *social distancing* tidak terdeteksi. Hal ini menunjukkan bahwa tingkat akurasi dari program *Perspective view Images* 45º lebih tinggi dibandingkan program *Perspective Bird's Eye View* 90º. Tingginya nilai *error* pada program *Perspective Bird's Eye View* 90º dipengaruhi oleh sistem yang mendeteksi objek selain manusia, sehingga sistem menghitung dan memproses dengan jumlah data pelanggaran *social distancing* lebih banyak maupun lebih sedikit dibandingkan dengan data asli atau data perhitungan manual.

DAFTAR PUSTAKA

Aditya Yanuar.r. 2018. *YOLO (you only look once)*. machinelearning.mipa.ugm.ac.id

Andre Oliver. 2022. *Deep Learning: Definisi, Jenis, Contoh Penerapan, dan Manfaatnya*. glints.com

Anirudh Rao. 2021. *OpenCV Python Tutorial: Computer Vision With OpenCV In Python*. edureka.co

Fadhilah Mileanasari. 2020. *Monitoring Of Physical Distance For Covid-19 Public Health Using You Only Look Once*. Jurnal. Universitas Muhammadiyah Malang: Malang

Lin-Bo Luo, In-Sung Koh, Kyeong-Yuk Min, Jun Wang, Jong-Wha Chong. 2010. *Low-cost Implementation of Bird's-eye View System for Camera-on-vehicle*. Jurnal. DOI: 10.1109/ICCE.2010.5418845 · Source: IEEE Xplore

Lusiana Mustinda 2020. *Perbedaan Social Distancing, Physical Distancing hingga PSBB*. news.detik.com: Jakarta

Mawaddah Harahap. 2019. *Sistem Cerdas Pemantauan Arus Lalu Lintas Dengan YOLO (You Only Look Once v3)*. Jurnal. Universitas Prima Indonesia: Medan

Oktaviani Ella Karlina. 2019. *Pengenalan Objek Makanan Cepat Saji Pada Video Dan Real Time Webcam Menggunakan Metode You Look Only Once (Yolo)*. Jurnal. Fakultas Teknologi Industri Universitas Gunadarma : Depok

Sellina Nuril Laili. (2022). Sistem Deteksi Kapasitas Orang Di Dalam Ruangan Menggunakan Metode *Faster R-Cnn*.

Uri Almog. 2016. *jaringan deteksi YOLO dan versinya 1, 2 dan terutama 3*. ichi.pro

Venkata Krishna Jonnalagadda. 2019. *Object Detection YOLO v1 , v2, v3*. medium.com

Wede. 2020. *Belajar Data Science : Apa yang dimaksud dengan Tensorflow dan Bagaimana Penggunaannya?*. dqlab.id: Jakarta