Sistem Penghitung Manusia Menggunakan *Drone*

1st Harry Prasetya
Prodi S1 Teknik Komputer
Universitas Telkom
Bandung, Indonesia
hariprasetya@student.telkomuniversity.ac.id

2nd Casi Setianingsih, S.T., M.T.
Prodi S1 Teknik Komputer
Universitas Telkom
Bandung, Indonesia
setiacasie@telkomuniversity.ac.id

3rd Anggunmeka Luhur Prasasti, S.T., M.T.
Prodi S1 Teknik Komputer
Universitas Telkom
Bandung, Indonesia
anggunmeka@telkomuniversity.ac.id

Abstrak* — Perkembangan teknologi pengawasan dan keamanan dalam beberapa tahun terakhir telah menghasilkan kemajuan yang signifikan, terutama dalam hal pengenalan objek atau object detection. Namun, masih terdapat tantangan dalam menghitung jumlah manusia dengan akurat dalam berbagai kondisi pengawasan. Studi ini bertujuan untuk mengoptimalkan proses pengawasan dengan menghitung jumlah manusia di area tertentu menggunakan teknologi drone yang dilengkapi dengan kemampuan pengenalan objek. Pendekatan yang digunakan mencakup penggunaan metode YOLOv7 (You Only Look Once) untuk pendeteksian objek dan algoritma Centroid Tracker untuk pelacakan manusia secara dinamis. Pengujian dilakukan dengan mengambil data dari drone dalam berbagai kondisi, seperti ketinggian dan kecepatan berbeda. Hasil pengujian menunjukkan bahwa akurasi deteksi manusia dipengaruhi oleh berbagai faktor, termasuk kesalahan pendeteksian, double counting, dan kesulitan dalam mengenali objek asing. Pada skenario drone diam, sistem memiliki performa yang lebih baik dalam mengenali dan menghitung manusia dengan rata-rata akurasi 69,70%. Namun, performa akurasi sangat tergantung pada berbagai faktor, termasuk kondisi deteksi dan kemampuan sistem dalam mengenali objek secara tepat. Kesimpulan utama adalah bahwa program human counting ini memiliki potensi dalam meningkatkan efisiensi pengawasan dan keamanan dengan menghitung jumlah manusia, tetapi perlu peningkatan dan pengembangan lebih lanjut untuk mengatasi kendala-kendala yang dihadapi dalam pengenalan objek dan pelacakan manusia dalam berbagai skenario pengawasan.

***Kata kunci* — object detection, YOLO, Centroid Tracker, drone, penghitungan manusia**

I. PENDAHULUAN

Dalam beberapa tahun terakhir, perkembangan teknologi di bidang pengawasan dan keamanan telah mengalami kemajuan yang signifikan. Fitur-fitur penting dalam produk-produk pengawasan kini menjadi lebih penting dalam menjaga lingkungan yang aman dan terkendali. Salah satu aspek teknologi yang mendapat perhatian besar adalah pengenalan objek atau *object detection*. Penerapan teknologi ini memiliki potensi besar dalam mengoptimalkan proses pengawasan, terutama dalam hal menghitung jumlah manusia di suatu area.

Identifikasi jumlah manusia merupakan elemen kunci yang perlu diperhatikan. Saat ini, banyak metode pengawasan masih mengandalkan pendekatan manual oleh petugas keamanan. Sebagai contoh, dalam penggunaan *drone* untuk mendeteksi kerumunan orang, masih terbatas dalam kemampuan menghitung secara akurat jumlah orang di dalam kerumunan tersebut. Proses perhitungan manual terhadap kelompok orang memakan waktu lama dan bisa menyebabkan kesalahan seperti menghitung individu yang sama lebih dari sekali.

Untuk mengatasi tantangan ini, pendekatan yang diusulkan adalah memanfaatkan teknologi *drone* yang dilengkapi dengan kemampuan penghitungan manusia. *Drone* akan mampu melakukan perhitungan jumlah orang yang terdeteksi dalam jangkauan kamera. Pendekatan ini dapat mempercepat proses pengawasan, mengurangi kesalahan perhitungan, dan meningkatkan efisiensi operasional keamanan.

Dalam implementasinya, teknologi yang digunakan termasuk bahasa pemrograman Python, serta teknik *YOLO (You Only Look Once)* versi 7 untuk deteksi objek dan algoritma *Centroid Tracker* untuk menghitung manusia. Dengan menggabungkan teknologi *drone*, pengenalan objek dengan *YOLO*, dan penghitungan manusia menggunakan *Centroid Tracker*, diharapkan efektivitas dan efisiensi dalam pengawasan dan keamanan lingkungan dapat ditingkatkan. Pendekatan ini akan berkontribusi positif dalam menciptakan lingkungan yang lebih aman dan terkendali secara lebih optimal.

II. KAJIAN TEORI

A. *Object Detection*

Pada saat manusia melihat sebuah objek dalam gambar maka otak manusia akan dapat langsung mengenali objek, letak, beserta kondisi interaksi yang terjadi. Dengan

sistem visual manusia yang cepat dan akurat, memungkinkan manusia untuk dapat melakukan tugas-tugas kompleks. Namun bagaimana jika tugas-tugas kompleks tersebut dilakukan oleh sebuah komputer. Untuk itu dibutuhkan sebuah algoritma yang cepat dan akurat yang akan memungkinkan komputer dapat melakukan hal serupa hingga berpotensi untuk menyelesaikan tugas secara umum [1].

Object Detection merupakan metode dalam bidang *computer vision* yang memungkinkan kita untuk mengenali serta menemukan objek yang terdapat dalam gambar atau *video*. Dengan pendekatan ini, kita dapat mengidentifikasi jenis-jenis objek yang ada dan juga menentukan serta melacak posisi mereka dengan akurat, sambil memberikan label yang tepat. Teknik identifikasi dan lokalisasi objek ini memiliki potensi besar dalam menghitung jumlah objek yang hadir dalam suatu adegan visual, serta memetakan dan melacak lokasi dari objek-objek tersebut [2].

B. Pengenalan Objek YOLOv7

Metode *You Only Look Once (YOLO)* adalah sebuah pendekatan inovatif dalam deteksi objek. YOLO menganalisis seluruh gambar secara keseluruhan dalam satu tahap, mengizinkan jaringan saraf tiruan untuk langsung mendeteksi objek-objek yang ada dalam gambar tersebut. Pendekatan ini menghasilkan kecepatan dan akurasi yang tinggi dalam pendeteksian, bahkan mengungguli metode-metode lain dengan kecepatan hingga dua kali lipat [3].

C. Pelacakan Objek dengan *Centroid Tracker*

Algoritma pelacakan *centroid tracker* digunakan untuk melacak dan menghitung individu-individu dalam suatu rangkaian gambar bergerak. Algoritma ini memanfaatkan informasi *bounding box* dari deteksi objek sebelumnya. Dengan menggunakan titik pusat (*centroid*) dari setiap *bounding box*, algoritma ini menghitung jarak Euclidean antara centroid objek yang sudah terdeteksi pada bingkai sebelumnya dan *centroid* objek yang muncul pada bingkai berikutnya. Hal ini memungkinkan pengenalan individu-individu dan pergerakan objek secara akurat serta memberikan ID unik pada setiap objek yang terdeteksi [3].

D. Penggunaan MonaServer dan OBS

MonaServer merupakan sebuah server multimedia yang fleksibel dan mampu memberikan solusi *real-time*. Ini mendukung berbagai protokol seperti RTMFP, RTMP, RTMPE, WebSocket, dan HTTP [4]. Sementara itu, OBS (*Open Broadcaster Software*) adalah perangkat lunak sumber terbuka yang memungkinkan merekam video dan melakukan siaran langsung [5]. Dalam konteks pengawasan dan keamanan, keduanya dapat diintegrasikan untuk menyusun sistem yang mengambil *feed* video dari *drone* melalui MonaServer, mengolahnya menggunakan OBS, dan mengirimkan hasilnya ke aplikasi web.

III. METODE

A. Implementasi YOLO untuk Pengenalan Objek

Metode *YOLO* digunakan untuk pendeteksian objek dalam citra atau video. *YOLO* bekerja dengan menganalisis gambar secara menyeluruh dalam satu tahap, mengidentifikasi dan menghitung objek-objek yang ada. Implementasi ini melibatkan mengintegrasikan model *YOLO* dengan *input* gambar dari sumber *video*, menjalankan pendeteksian, dan menghasilkan *bounding box* serta label untuk objek-objek yang terdeteksi.

B. Pelacakan dan Penghitungan dengan *Centroid Tracker*

Algoritma *centroid tracker* digunakan untuk melacak pergerakan individu dalam video bergerak. Prosesnya melibatkan deteksi objek menggunakan YOLO, penghitungan titik pusat (*centroid*), dan penerapan mekanisme pelacakan menggunakan algoritma *centroid tracker*. Ini memungkinkan untuk menghitung jumlah manusia dalam area tertentu dan melacak pergerakan mereka dari bingkai ke bingkai.

C. Integrasi dengan MonaServer dan OBS

Integrasi MonaServer dan OBS melibatkan pengaturan MonaServer sebagai server RTMP untuk menerima *feed* video dari *drone* dan mengirimkannya ke OBS. OBS digunakan untuk memproses video, termasuk *overlay* dan elemen lainnya, serta mengirim hasilnya ke aplikasi web. Konfigurasi dilakukan pada kedua platform untuk memastikan aliran video yang lancar dan sesuai dengan kebutuhan.

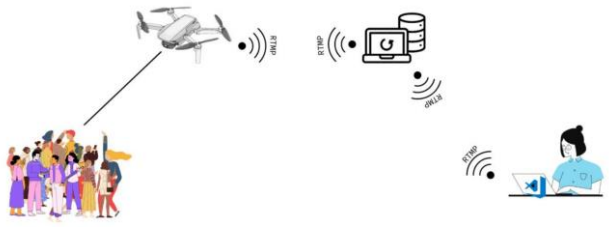
D. Implementasi dengan Python

Pengembangan solusi ini dilakukan menggunakan bahasa pemrograman Python. Python digunakan untuk menghubungkan, mengatur aliran data, serta mengintegrasikan berbagai komponen sistem. Pemrograman Python memungkinkan interaksi yang efisien antara algoritma deteksi, pelacakan, dan integrasi dengan platform MonaServer dan *OBS*.

Dengan menggabungkan pengenalan objek *YOLO*, pelacakan dengan *centroid tracker*, dan integrasi MonaServer dan *OBS*, serta implementasi dengan Python, solusi ini mampu menghasilkan sistem pengawasan dan perhitungan manusia yang efektif dan efisien.

IV. RANCANGAN SISTEM

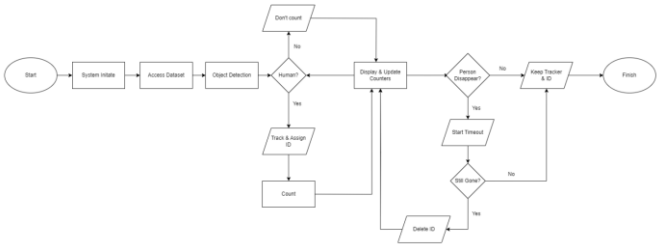
A. Arsitektur Utama



Gambar 1 Arsitektur Utama Sistem

Gambar 1 merupakan arsitektur utama sistem, dimana ada tiga komponen utama di dalamnya yaitu satu set komputer yang digunakan untuk menjalankan kode program human *counting*, RTMP yang digunakan untuk menerima *feed* video dari *drone* dan mengirimkannya ke OBS, serta *drone* sebagai media pengambilan video.

B. Flowchart Human Counting



Gambar 2 *Flowchart Human Counting*

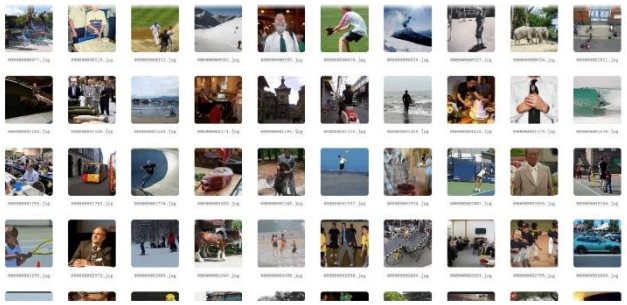
Gambar 2 menggambarkan mekanisme kerja dari penghitungan jumlah manusia (*human counting*). Sistem diawali dengan inisialisasi, kemudian mengakses *dataset* untuk mendapatkan referensi kelas objek "person" untuk deteksi objek. Jika sistem mendeteksi objek sebagai "person", langkah selanjutnya adalah melakukan perhitungan dan melaksanakan pelacakan (*tracking*). Di sisi lain, jika sistem tidak mendeteksi objek sebagai "person", maka tidak akan diterapkan tindakan seperti pembuatan bingkai batasan (*bounding box*) atau perhitungan.

Setelah proses deteksi dan pelacakan dilakukan, sistem akan mencatat jumlah total orang yang terhitung dan jumlah orang yang sedang terdeteksi dalam bingkai (*frame*) saat itu. Dalam kasus di mana seseorang menghilang dari bingkai, mekanisme *timeout* akan diaktifkan. *Timeout mechanism* berfungsi menghitung berapa lama seseorang tidak terdeteksi dalam bingkai. Jika seseorang menghilang selama periode yang melebihi batas yang ditentukan, ID yang sebelumnya diberikan akan dihapus, dan hitungan orang saat ini (yang terdeteksi dalam bingkai) akan diperbarui.

V. IMPLEMENTASI

A. Pengumpulan Gambar

Proses pengumpulan gambar dalam konteks human *counting* dilakukan dengan dua metode utama. Pertama, melibatkan pengambilan gambar secara langsung yang mengabadikan keberadaan manusia dalam berbagai situasi. Metode ini mencakup pengambilan foto manusia dalam lingkungan yang berbeda-beda. Selain itu, gambar yang sudah ada dalam penyimpanan perangkat, seperti foto *selfie* dan sejenisnya, juga digunakan sebagai bagian dari *dataset*. Kedua, untuk melengkapi *dataset*, gambar-gambar yang diambil dari sumber internet juga dimanfaatkan. Dalam proses ini, format gambar yang digunakan mencakup .jpg, .jpeg, dan .png. Semua gambar ini akan menjadi bagian penting dalam membangun *dataset* untuk tujuan penghitungan manusia.



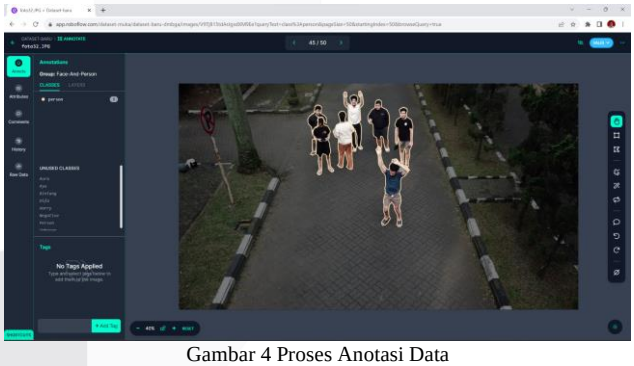
Gambar 3 Gambar Untuk *Dataset*

Gambar 3 merupakan contoh kumpulan gambar yang digunakan untuk *dataset* perhitungan manusia, gambar-gambar yang digunakan berkaitan dengan objek manusia yang diambil dari berbagai sudut pandang dan lingkungan yang berbeda.

B. Anotasi Data

Proses Anotasi Data dilakukan dengan memberikan label pada setiap data atau gambar mentah, dengan tujuan memberikan informasi tentang identifikasi suatu objek. Langkah ini memungkinkan komputer untuk memahami dan mempelajari cara membedakan antara objek-objek tersebut. Proses Anotasi Data juga memiliki peran penting dalam meningkatkan keterampilan algoritma pembelajaran mesin untuk mengatasi tantangan dunia nyata. Khususnya dalam konteks visi komputer, anotasi dataset memungkinkan model untuk memahami variasi dalam tampilan objek yang berbeda, termasuk posisi, ukuran, sudut pandang, dan kondisi pencahayaan. Dengan memberikan label yang akurat, algoritma dapat membedakan objek yang serupa namun memiliki perbedaan signifikan, serta mengidentifikasi konteks di sekitarnya.

Selain itu, dalam proses Anotasi Data terdapat berbagai jenis anotasi yang berkontribusi pada pengembangan model visi komputer yang lebih unggul dan dapat diandalkan. Beberapa jenis anotasi meliputi bingkai batasan (*bounding box*) untuk deteksi objek, segmentasi piksel untuk memisahkan objek dari latar belakang, dan titik *landmark* untuk mengidentifikasi fitur-fitur penting pada objek. Semua bentuk anotasi ini berperan dalam memperkuat kemampuan model untuk mendeteksi objek dan menghitung jumlah manusia dengan lebih baik.



Gambar 4 Proses Anotasi Data

Gambar 4 menunjukkan proses anotasi data yang dilakukan menggunakan platform pengolahan *dataset* Roboflow.

Roboflow merupakan sebuah kerangka kerja pengembangan dalam bidang *computer vision* yang bertujuan untuk meningkatkan proses pengumpulan data, pra-pemrosesan, dan teknik pelatihan model. Platform ini menawarkan akses kepada penggunanya terhadap berbagai kumpulan data publik yang telah disediakan, serta memberikan opsi bagi pengguna untuk mengunggah *dataset* khusus sesuai kebutuhan mereka. Selain itu, Roboflow juga dilengkapi dengan kemampuan untuk mengonversi data yang telah diberi label, memudahkan dalam proses pelatihan model [6].

C. Pemrograman

Penerapan penghitungan manusia dengan memanfaatkan mekanisme *timeout* dan pendekatan *Centroid Tracker* menghasilkan pendekatan yang efisien dan dinamis dalam melacak serta menghitung individu dengan akurasi yang tinggi dalam berbagai skenario. Pendekatan *Centroid Tracker* difokuskan pada menghubungkan objek yang terdeteksi dengan menghitung pusatnya, menghasilkan proses pelacakan yang lebih sederhana dan adaptif terhadap variasi pergerakan objek. Untuk meningkatkan akurasi, diterapkan pula mekanisme *timeout* yang berfungsi mencegah penghitungan berulang dalam periode tertentu.

Mekanisme *timeout* ini memastikan bahwa setelah individu terdeteksi dan dihitung, jika individu tersebut keluar dari bingkai, maka akan diaktifkan *timeout*. Dalam jangka waktu *timeout*, deteksi berikutnya dari individu yang sama akan diabaikan. Langkah ini membantu mencegah penghitungan berlebihan karena adanya deteksi berkelanjutan yang terus menerus. Dengan menggabungkan pendekatan *Centroid Tracker* dan mekanisme *timeout*, implementasi ini tidak hanya memberikan kemampuan pelacakan secara *real-time* tetapi juga mengurangi kesalahan akibat penghitungan ganda.

```
class CentroidTracker:
    def __init__(self, maxDisappeared=40):
        self.nextObjectID = 0
        self.objects = {}

  return go(f, seed, [])
}
```

Gambar 5 Kode Program Class CentroidTracker

Kode Program Class CentroidTracker, seperti yang terlihat dalam Gambar 5, menggambarkan struktur dan logika dari kelas CentroidTracker yang dirancang untuk pelacakan objek. Metode pembuat (__init__) ini memiliki parameter opsional "maxDisappeared", yang mengindikasikan jumlah maksimum frame berturut-turut di mana suatu objek dapat tidak terdeteksi sebelum dianggap "hilang" dan dihapus dari pelacakan. Secara *default*, nilai maxDisappeared diatur ke 40 *frame*.

Selanjutnya, variabel *instance* "self.nextObjectID" bertindak sebagai penanda untuk ID berikutnya yang akan diberikan kepada objek yang didaftarkan untuk pelacakan. Dimulai dari 0, nilai ini akan terus bertambah setiap kali ada objek baru yang akan diberi ID.

Pada bagian "self.objects", sebuah *dictionary* digunakan untuk menyimpan objek yang sedang dilacak. Key (kunci) yang digunakan adalah ID unik yang diberikan kepada masing-masing objek dalam *instance* kelas person. Informasi mengenai setiap objek akan diatur dan diperbarui di dalam *dictionary* ini, sehingga memungkinkan untuk pelacakan dan perhitungan jumlah manusia yang lebih akurat dalam berbagai situasi. Dengan komponen-komponen ini, kelas CentroidTracker dapat digunakan untuk melacak objek dan menghitung jumlah manusia dalam lingkungan yang dinamis.

```
def register(self, centroid):
    person = Person(self.nextObjectID, centroid)
    self.objects[self.nextObjectID] = person
    self.nextObjectID += 1
```

Gambar 6 Kode Program def register

Kode Program Fungsi "register", seperti yang terlihat dalam Gambar 6, menggambarkan langkah-langkah yang diperlukan untuk mendaftarkan objek baru dalam sistem pelacakan. Metode ini berperan dalam mengintegrasikan objek baru ke dalam alur pelacakan yang sedang berjalan.

Dalam metode "register(self, centroid):", parameter "centroid" yang diberikan mewakili pusat (*centroid*) dari objek yang telah terdeteksi. Pada tahap ini, langkah pertama yang dilakukan adalah menciptakan *instance* baru dari kelas "Person". Proses ini melibatkan pemberian ID kepada objek baru menggunakan nilai "nextObjectID" yang sedang berlaku. Dengan memanfaatkan ID ini, *instance* "Person" dibuat dengan mengambil ID dan nilai centroid yang diberikan.

Setelah pembuatan *instance* "Person", nilai "nextObjectID" ditingkatkan, sehingga memastikan bahwa ID berikutnya yang akan digunakan untuk objek berikutnya sudah disiapkan. Selanjutnya, objek "person" yang baru dibuat ini ditambahkan ke dalam *library* "objects", dengan ID-nya sebagai kunci. Ini memungkinkan sistem untuk melacak objek berdasarkan ID unik mereka dan melaksanakan *timeout* jika diperlukan.

Melalui serangkaian langkah ini, fungsi "register" memastikan bahwa objek baru dapat terintegrasi dengan sukses ke dalam sistem pelacakan yang ada, dan dapat dilacak serta dihitung sesuai kebutuhan.

```
def deregister(self, objectID):
    del self.objects[objectID]
```

Gambar 7 Kode Program Fungsi deregister

Kode Program Fungsi "deregister", seperti yang terlihat dalam Gambar 6, menunjukkan bagaimana sistem menghapus objek dari proses pelacakan ketika objek tersebut tidak lagi terdeteksi. Fungsi ini berfungsi untuk mengelola objek-objek yang telah berpindah dari pandangan kamera atau tidak terdeteksi dalam bingkai (frame) saat ini.

Dalam metode "deregister(self, objectID):", parameter "objectID" yang diberikan merupakan identifikasi unik dari objek yang akan dihapus dari sistem pelacakan. Langkah pertama yang diambil adalah menggunakan nilai "objectID" untuk mengidentifikasi dan menghapus entri yang sesuai dari library "objects".

Dengan menghapus entri objek yang tidak lagi diperlukan dari *library* "objects", sistem dapat menjaga efisiensi dan keakuratan pelacakan. Ini juga memungkinkan sistem untuk secara dinamis mengelola objek-objek yang mungkin keluar dari pandangan kamera atau berpindah dari bingkai saat ini. Dengan demikian, fungsi "deregister" merupakan langkah penting dalam menjaga integritas dan akurasi proses pelacakan objek

D. Pengujian

Program human *counting* dirancang dengan tujuan khusus untuk mendeteksi dan menghitung jumlah manusia yang terperangkap oleh bingkai kamera. Dalam rangka menguji performa sistem, serangkaian pengujian akan dilakukan untuk mengevaluasi akurasi penghitungan manusia dari berbagai ketinggian penerbangan serta sudut pandang. Proses pengujian akan mengikuti skema langkah-langkah berikut:

Pertama, dilakukan pengaturan dasar pengujian. Berbagai ketinggian penerbangan yang akan diuji ditentukan sebelumnya, termasuk ketinggian 3 meter, 5 meter, 7 meter, dan 10 meter. Selain itu, area pengujian dipersiapkan dengan mempertimbangkan area terbuka dan area yang memiliki hambatan visual, seperti pepohonan atau bangunan, guna menguji performa deteksi dalam kondisi yang berbeda.

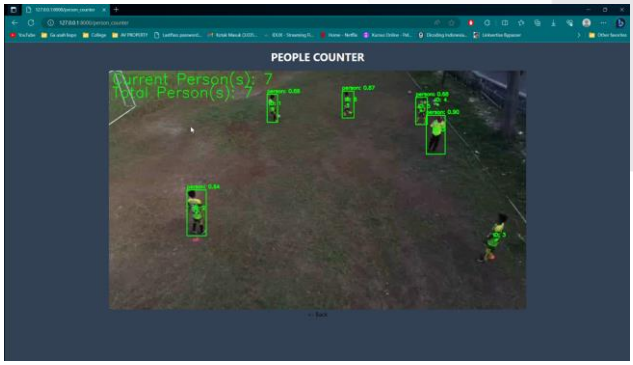
Kemudian, dilakukan pengujian penghitungan manusia. *Drone* diterbangkan pada setiap ketinggian yang telah ditentukan. Video *streaming* dari *drone* diambil dan program penghitungan manusia direkam saat dilakukan. Selama pengujian, jumlah orang yang benar-benar hadir dalam bingkai video dicatat secara manual sebagai data referensi yang valid.

Selanjutnya, dilakukan analisis performa. Metode pengenalan objek YOLOv7 digunakan dalam pengujian ini untuk mendeteksi dan menghitung manusia pada setiap video yang diambil. Hasil penghitungan dari algoritma dibandingkan dengan data referensi manual yang telah dicatat sebelumnya. Perbandingan ini bertujuan untuk mengevaluasi seberapa akurat sistem dalam menghitung jumlah manusia pada berbagai ketinggian.

Terakhir, dilakukan perhitungan akurasi pengenalan. Akurasi pengenalan dihitung untuk setiap ketinggian penerbangan dengan membandingkan jumlah orang yang terdeteksi secara akurat oleh algoritma dengan jumlah orang yang sebenarnya hadir yang telah dicatat secara manual. Hasil akurasi ini memberikan gambaran tentang sejauh mana sistem mampu menghitung jumlah manusia dengan tepat pada berbagai ketinggian yang diuji.

VI. HASIL DAN PEMBAHASAN

A. Hasil Akhir Sistem



Gambar 8 Tampilan Halaman Sistem *People Counter*

Tampilan untuk *display output* pada *people counter* memperlihatkan hasil *output* yang terhubung dengan program dan juga *drone*. *Drone* sebagai sumber yang terhubung dengan OBS akan ditampilkan POV-nya di dalam *page* ini. *Page* ini juga memperlihatkan *bounding box*, *confidence score* dan ID pada objek-objek yang terdeteksi berapa total orang yang terdeteksi dan juga berapa jumlah orang yang sedang berada di dalam *frame*.

B. Hasil Pengujian

Tabel 1 Hasil Uji Posisi *Drone* Bergerak

Height	Speed	Total Orang (*Real*)	Counter	Error Deteksi	Akurasi Deteksi
3m	0-2 m/s	7	15	53.33%	46.67%
	0-6 m/s	7	8	12.5%	87.5%
	0-10 m/s	7	15	53.33%	46.67%
5m	0-2 m/s	8	14	42.8%	57.2%
	0-6 m/s	7	13	46.15%	53.85%
	0-10 m/s	7	10	30%	70%
7m	0-2 m/s	8	9	11.11%	88.89%
	0-6 m/s	7	10	30%	70%
	0-10 m/s	7	8	12.5%	87.5%
10m	0-2 m/s	6	5	16.67%	83.33%
	0-6 m/s	6	5	16.67%	83.33%
	0-10 m/s	6	3	50%	50%

Tabel 1 merupakan hasil pengujian program human *counting* dengan skenario *drone* bergerak. Hasil uji akurasi deteksi dengan data pada Tabel tersebut mengindikasikan beberapa temuan penting terkait hubungan antara akurasi deteksi dan kondisi deteksi yang diberikan. Ketika akurasi deteksi berada pada kisaran sekitar 46.67% hingga 70% terdapat kasus *double counting* pada subjek yang sama, tampak bahwa sistem masih mengalami kesulitan dalam mengenali objek manusia secara akurat. Lebih lanjut, ketika akurasi deteksi mencapai 87.5% juga terjadi *double counting* pada subjek yang sama, namun menunjukkan performa yang lebih baik dalam mengenali objek, meskipun masih terdapat kendala yang perlu diatasi.

Namun, perlu diperhatikan bahwa pada akurasi deteksi sekitar 50%, terjadi beberapa orang yang tidak terdeteksi dan tidak terhitung. Situasi ini menggambarkan bahwa sistem belum mampu mengenali dan menghitung manusia dengan akurat dalam kondisi tersebut. Selanjutnya, saat akurasi deteksi mencapai 83.33% dan terdapat objek asing yang terdeteksi sebagai manusia, serta saat akurasi deteksi tetap pada tingkat yang sama dan terjadi deteksi objek asing yang sekilas, sistem mengalami kesulitan dalam mengenali objek manusia dan dalam beberapa kasus, terjadi kebingungan dengan objek lain.

Dari hasil data tersebut, dapat disimpulkan bahwa akurasi deteksi sangat dipengaruhi oleh berbagai faktor seperti *double counting*, kesalahan dalam mengenali objek asing, serta kesulitan dalam menghitung manusia dengan benar. Faktor-faktor tersebut memberikan dampak signifikan terhadap kemampuan sistem dalam mengenali dan menghitung manusia dalam berbagai kondisi deteksi.

Tabel 2 Hasil Uji Posisi *Drone* Diam

Height	Speed	Total Orang (*Real*)	Counter	Error Deteksi	Akurasi Deteksi
3m	30°	8	15	46.67%	53.33%
	45°	8	8	0%	100%
	90°	2	2	0%	100%
5m	30°	9	5	44,44%	55,56%
	45°	9	12	25%	75%
	90°	6	5	16.67%	83.33%
7m	30°	9	0	100%	0%
	45°	9	11	18.18%	81.82%
	90°	3	3	0%	100%
10m	30°	9	1	88,89%	11,11%
	45°	9	8	11.11%	88.89%
	90°	7	8	12.5%	87.5%

Tabel 2 merupakan hasil pengujian program human *counting* dengan skenario posisi *drone* diam di tempat atau tidak bergerak. Berdasarkan data hasil uji akurasi deteksi pada berbagai skenario ketinggian *drone*, dan kecepatan, total orang di dalam *frame*, dapat ditarik beberapa simpulan mengenai hubungan antara akurasi deteksi dan kondisi deteksi. Saat akurasi deteksi mencapai 100% dan terjadi beberapa *double counting* pada subjek yang sama, sistem mampu mendeteksi manusia dengan akurat, meskipun terdapat kesalahan *double counting*. Kasus di mana akurasi deteksi juga mencapai 100% dan banyak orang yang terdeteksi sesuai menunjukkan performa sistem yang baik dalam mengenali dan menghitung jumlah manusia dengan akurat. Namun, situasi di mana akurasi deteksi berada di kisaran 50% hingga 75% dan terdapat objek asing yang terdeteksi sebagai person, serta beberapa orang tidak terdeteksi, mengindikasikan tantangan sistem dalam mengenali objek manusia dan dapat menghasilkan kesalahan deteksi yang signifikan. Ketika akurasi deteksi mencapai 75% dan terjadi objek asing yang terdeteksi sebagai person, serta *double counting*, sistem mengalami kesulitan dalam mengenali objek sebenarnya dan kesalahan tersebut memengaruhi akurasi deteksi. Situasi dengan akurasi deteksi sekitar 83.33% dan terjadi *error* pendeteksian di mana 2 individu terdeteksi dalam satu *bounding box* menunjukkan kemampuan sistem dalam deteksi objek, namun masih terdapat beberapa kesalahan. Di sisi lain, ketika akurasi deteksi mencapai 0% dan terdapat objek asing yang terdeteksi sebagai person, sistem gagal mengenali objek manusia secara memadai. Performa sistem juga dipengaruhi oleh kesalahan *double counting*, seperti pada kasus dengan akurasi deteksi sekitar 81.82%. Situasi dengan akurasi deteksi rendah, seperti 11.11%, menunjukkan bahwa sistem mengalami kendala dalam mengenali objek manusia secara akurat. Akhirnya, akurasi deteksi sekitar 88.89% dan satu orang tidak terdeteksi serta tidak terhitung menunjukkan bahwa sistem mampu deteksi sebagian besar objek, tetapi masih terdapat kesalahan dalam mengenali individu tertentu. Dari data ini, terlihat bahwa akurasi deteksi sangat dipengaruhi oleh berbagai faktor, termasuk kesalahan pendeteksian, *double counting*, dan deteksi objek asing, yang secara signifikan mempengaruhi performa sistem dalam mengenali dan menghitung manusia dalam berbagai kondisi.

C. Analisis Hasil Pengujian

Proses pengujian Human *Counting* dilakukan dengan mengambil POV *drone* dari ketinggian 3m, 5m, 7m dan 10m, dengan setiap titik ketinggian dilakukan pengujian dari tiga titik sudut yaitu sudut 30°, 45° dan 90°. Berikut adalah nilai rata-rata akurasi dari masing-masing titik ketinggian :

1. Nilai rata-rata akurasi deteksi dan penghitungan manusia saat *drone* dalam keadaan bergerak.
   - Ketinggian 3 meter :
     $$\frac{46.67 + 87.5 + 46.67}{3} \text{X } 100\% = 60{,}28\%$$
   - Ketinggian 5 meter :
     $$\frac{57.2 + 53.85 + 70}{3} \text{X } 100\% = 60{,}35\%$$
   - Ketinggian 7 meter :
     $$\frac{88.89 + 70 + 87.5}{3} \text{X } 100\% = 82{,}13\%$$
   - Ketinggian 10 meter :
     $$\frac{88.89 + 70 + 87.5}{3} \text{X } 100\% = 82{,}13\%$$
2. Nilai rata-rata akurasi deteksi dan penghitungan manusia saat *drone* dalam keadaan diam.
   - Ketinggian 3 meter :
     $$\frac{53.33 + 100 + 100}{3} \text{X } 100\% = 84{,}4\%$$
   - Ketinggian 5 meter :
     $$\frac{55{,}56 + 75 + 83.33}{3} \text{X } 100\% = 71{,}29\%$$
   - Ketinggian 7 meter :
     $$\frac{0 + 81.82 + 100}{3} \text{X } 100\% = 60{,}6\%$$
   - Ketinggian 10 meter :
     $$\frac{11{,}11 + 88.89 + 87.5}{3} \text{X } 100\% = 62{,}5\%$$

VII. KESIMPULAN

Berdasarkan hasil pengujian dan analisis yang dilakukan terhadap program *human counting* dengan menggunakan teknologi pengenalan objek dan pelacakan manusia, dapat diambil beberapa kesimpulan penting. Program ini memiliki potensi besar dalam mengoptimalkan proses pengawasan dan keamanan dengan menghitung jumlah manusia di suatu area. Penggunaan metode *YOLOv7 (You Only Look Once)* untuk deteksi objek dan algoritma *Centroid Tracker* untuk pelacakan manusia secara dinamis membantu meningkatkan efisiensi operasional dan akurasi dalam mendeteksi serta menghitung manusia.

Pengujian yang dilakukan mengindikasikan bahwa akurasi deteksi sangat dipengaruhi oleh berbagai faktor,

termasuk kondisi deteksi seperti ketinggian *drone*, sudut pandang, dan kecepatan pergerakan. Saat akurasi deteksi mencapai tingkat yang tinggi, sistem mampu mendeteksi manusia secara akurat. Namun, terdapat situasi di mana akurasi deteksi menurun, dan hal ini dapat disebabkan oleh beberapa faktor, termasuk kesalahan *double counting* dan objek asing yang terdeteksi sebagai manusia. Penyebab utama terjadinya *double counting* adalah bahwa metode YOLO tidak memiliki kemampuan untuk mengenali setiap individu secara unik, sehingga sulit untuk membedakan individu yang sama atau berbeda dalam suatu bingkai. Kendala-kendala ini secara signifikan memengaruhi kemampuan sistem dalam menghitung jumlah manusia dengan benar.

Dalam skenario *drone* diam di tempat, sistem memiliki performa yang lebih baik dalam mengenali dan menghitung manusia dengan rata-rata akurasi 69,70%.

Dengan demikian, kesimpulan utama adalah bahwa program human *counting* ini memiliki potensi yang signifikan dalam meningkatkan efisiensi pengawasan dan keamanan dengan menghitung jumlah manusia. Namun, performa akurasi sangat tergantung pada berbagai faktor, termasuk kondisi deteksi dan kemampuan sistem dalam mengenali objek secara tepat. Pengembangan lebih lanjut dan peningkatan algoritma serta pengujian lebih komprehensif diperlukan untuk mengatasi kendala-kendala tersebut dan memastikan keakuratan yang lebih baik dalam menghitung manusia dalam berbagai skenario pengawasan.

REFERENSI

[1] T. A. Dompeipen, S. R. Sompie dan M. E. Najoan, "Computer Vision Implementation for Detection and Counting the Number of Humans," *Jurnal Teknik Informatika*, vol. 16, no. 1, pp. 65-76, 2021.

[2] H. Mulyo dan H. Kusumodestoni, "Object Detection pada CCTV untuk Smart City Kabupaten Kendal," *AMRI (Analisa, Metode, Rekayasa, Informatika)*, vol. 1, no. 2, pp. 121-124, 2022.

[3] F. Rachmawati dan D. Widhyaestoeti, "Deteksi Jumlah Kendaraan di Jalur SSA Kota Bogor Menggunakan Algoritma Deep Learning YOLO," *PROSIDING LPPM UIKA BOGOR*, 2020.

[4] M. Poux dan T. Jammet, "Overview - MonaServer," [Online]. Available: https://www.monaserver.ovh/. [Diakses 16 Agustus 2023].

[5] D. Rizana dan M. Huda, "Pelatihan pembuatan video pembelajaran menggunakan OBS studio," *COMMUNITY EMPOWERMENT*, vol. 6, no. 5, 2021.

[6] M. D. R. Pratama, B. Priyatna, S. S. Hilabi dan A. L. Hananto, "Deteksi Objek Kecelakaan Pada Kendaraan Roda Empat Menggunakan Algoritma YOLOv5," *Teknologi: Jurnal Ilmiah Sistem Informas*, vol. 12, no. 2, pp. 15-26, 2022.