Implementasi *Face Recognition* Berbasis *Deep Neural Network* Sebagai Sistem Kendali Pada *Quadcopter*

Implementation Of Face Recognition Based On Deep Neural Network As Control System On Quadcopter

1st Bangga Adi Septyanto
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
banggaadi@student.telkomuniversity.ac.id

2nd Suryo Adhi Wibowo
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
suryoadhiwibowo@telkomuniversity.ac.id

3rd Casi Setianingsih
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
casisetianingsih@telkomuniversity.ac.id

Abstrak—*Artifcial Intelligence* (AI) adalah bidang yang dikembangkan untuk mempelajari dan menyerupai kecerdasan manusia ke dalam sebuah sistem komputer. Salah satu hasil dari pengembangan AI khususnya pada bidang *deep neural network* adalah teknologi *face recognition*. *Face recognition* merupakan suatu sistem yang digunakan untuk mengenali wajah manusia yang terdapat pada suatu citra. Tugas akhir ini dilakukan implementasi teknologi *face recognition* terhadap *unmanned aerial vehicle* (UAV) untuk beberapa kepentingan seperti pengawasan, pencarian orang, dan pemantauan jarak jauh. Dalam tugas akhir ini penerapan *face recognition* diimplementasikan menggunakan algoritma YOLOv5 sebagai *object detection* serta pembuatan algoritma *object tracking*. Untuk mengetahui performansi terbaik, penelitian ini akan dilakukan pengujian terhadap 3 model, yaitu YOLOv5n, YOLOv5s dan YOLOv5m. Untuk mencari model terbaik akan dievaluasi menggunakan parameter *Mean Average Precision* (mAP), jarak efektif deteksi dan kecepatan inferensi model. Hasil perbandingan menunjukkan model YOLOv5n memiliki kecepatan inferensi terbaik yaitu 5ms. Namun, model ini memiliki nilai mAP terendah yaitu 79,8%. Sedangkan, mAP tertinggi dengan nilai 84,8% dicapai oleh model YOLOv5m dengan kecepatan inferensi sebesar 13ms. Model YOLOv5s memiliki selisih kecepatan inferensi yang kecil dengan YOLOv5n dengan 6ms dan nilai mAP sebesar 82,4%.

Kata Kunci— *Face Recognition, Unmanned Aerial Vehicle, YOLOv5, Deep Neural Network.*

Abstract—Artifcial Intelligence (AI) is a feld developed to study and mimic human intelligence into a computer system. One of the results of AI development, especially in the feld of deep neural networks is face recognition technology. Face recognition is a system that It is used to recognize human faces in an image. In this fnal project, the implementation of face recognition technology on the unmanned aerial vehicle (UAV) for several purposes such as surveillance, people search, and remote monitoring. In this fnal project, the application of face recognition is implemented using the YOLOv5 algorithm as object detection and making algorithm for object tracking. To fnd out the best performance, this research will test 3 models, namely YOLOv5n, YOLOv5s, and YOLOv5m. To fnd the best model will be evaluated using the Mean Average Precision (mAP) parameters, the effective detection distance, and the model inference speed. The comparison results show that the YOLOv5n model has the best inference speed of 5ms, however, this model has the lowest mAP value of 79,8%. Meanwhile, YOLOv5m has the best mAP value of 84,8% with an inference speed of 13ms. And the YOLOv5s model has a large difference in inference speed small with YOLOv5n with 7ms and mAP value of 82,4%.

Keywords: Face Recognition, Unmanned Aeiral Vehicle, YOLOv5, Deep Neural Network.

I. PENDAHULUAN

Seiring dengan perkembangan teknologi, penerapan dan pemanfaatan *Unmanned Aerial Vehicle* (UAV) menjadi salah satu alternatif untuk membantu pekerjaan manusia dalam kehidupan setiap harinya. UAV merupakan pesawat terbang yang dapat beroperasi tanpa awak yang dapat dikendalikan secara

jarak jauh atau otomatis [1]. Banyak pemanfaatan yang didapat dari penggunaan UAV dikarenakan mereka dapat menjangkau lokasi yang sulit dijangkau oleh manusia. Selain itu, UAV juga dapat mengambil gambar dan video melalui sudut pandang burung atau *bird's-eye view*. UAV yang dilengkapi dengan kamera dapat digunakan untuk perihal pengawasan, pemantauan jarak jauh, fotograf dan perfilman. Penerapan teknologi *Artifcial Intelligence* pada UAV dapat meningkatkan potensi yang dimiliki UAV untuk melakukan tugas yang diberikan.

Artifcial Intelligence (AI) adalah bidang yang dikembangkan untuk mempelajari pola dari data mentah kedalam sebuah sistem komputer [2]. Salah satu bidang pada AI yang sedang berkembang pesat adalah *face recognition*. Identitas yang unik dan memiliki ciri khas masing-masing pada setiap orang didefnisikan dengan wajah manusia oleh karena itu, implementasi *face recognition* kepada perangkat UAV akan menjadi suatu keuunggulan karena dapat melakukan deteksi pada wajah manusia [3]. *Face recognition* adalah cabang dari bidang *object detection* yang dapat mengidentifikasi beda wajah manusia. Kemampuan dari model *face recognition* yang dibuat menjadi kunci utama untuk membuat UAV dapat mengenali wajah yang terdapat pada suatu kumpulan manusia. Namun, terdapat beberapa rintangan yang dialami oleh peneliti dalam pengembangan model *face recognition*, seperti jarak optimal untuk pendeteksian, kecepatan komputasi model, dan kompleksitas pengenalan wajah itu sendiri [4].

Penelitian terhadap penggunaan *face recognition* pada UAV telah dilakukan oleh Yu Fan, Yiyue Luo, dan Xianjun Chen. Penelitian tersebut membandingkan beberapa algoritma *deep learning* dengan *improved* YOLOv3. Metode yang diajukan pada penelitian tersebut adalah dengan mengaplikasikan *dense block module* untuk mengurangi kompleksitas jaringan, mengurangi pekerjaan memori model jaringan, dan memastikan akurasi yang lebih tinggi. Pada penelitian tersebut didapat hasil akurasi *improved* YOLOv3 sebesar 86,74% dengan kecepatan inferensi 63,54 ms, sementara untuk algoritma orisinalnya yaitu YOLOv3 didapat hasil akurasi sebesar 80,22% dengan kecepatan inferensi 85,69 ms [5]. Namun, pengaplikasian model *deep learning* terhadap *quadcopter* memerlukan waktu komputasi yang singkat guna menjamin pendeteksian secara *real time*.

Berdasarkan beberapa penelitian yang sudah dijelaskan, pelaksanaan pengenalan wajah dapat dilakukan dengan beragam metode. Dalam tugas Akhir ini akan mengaplikasikan algoritma *You Only Look Once* (YOLO) versi lima yang dikembangkan oleh ultralytics sebagai pendeteksi wajah. YOLOv5 merupakan sebuah algoritma deteksi objek yang memiliki akurasi tinggi dan komputasi inferensi yang cepat. Selain itu, YOLOv5 juga memiliki keunggulan pada sisi *deployment* karena model yang dihasilkan lebih ringan dan memiliki ukuran yang lebih kecil jika dibandingkan dengan versi sebelumnya. Algoritma ini dibagi menjadi lima model yang berbeda yaitu YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l dan YOLOv5x. Akurasi dan kecepatan inferensi pada algoritma YOLOv5 bergantung kepada jenis model yang digunakan karena pada setiap model memiliki ukuran parameter dan kompleksitas *layer* yang berbeda-beda.

Terdapat dua faktor penting dalam tugas akhir ini, yaitu akurasi dan kecepatan inferensi. Pengaplikasian model harus dilakukan secara *real-time* sehingga membutuhkan model yang ringan. Oleh karena itu, pada tugas akhir ini dilakukan pengaplikasian tiga model YOLOv5 yaitu YOLOv5n, YOLOv5s dan YOLOv5m untuk mencari model dengan performa kecepatan inferensi serta akurasi terbaik. Serta, pada tugas akhir ini akan mengintegrasikan fitur *object tracking* sebagai sistem kendali pada UAV.

II. KAJIAN TEORI

A. *Quadcopter*

Quadcopter merupakan salah satu jenis *Unmanned Aerial Vehicle* (UAV) yang memiliki empat lengan dengan baling-baling, dua baling-baling berputar searah jarum jam dan sisanya berputar melawan arah jarum jam [6]. Sebuah *quadcopter* memerlukan sebuah komputer untuk memberikan perintah yang dapat membuat perubahan kecepatan putar pada baling-baling sehingga dapat bergerak sesuai perintah. Kemajuan pada bidang elektronik memungkinkan pada *quadcopter* untuk beroperasi tanpa awak dan dapat diberikan penambahan fitur seperti *microprocessor*, kamera, dan bahkan *Global Positioning System* (GPS) sehingga *quadcopter* menjadi alat yang sangat efektif untuk digunakan pada bidang fotografi dan videografi.

B. *Face Recognition*

Face Recognition atau Pengenalan wajah merupakan sebuah sistem yang dibuat untuk mengenali wajah manusia. Pengenalan wajah melibatkan perbandingan dari banyak wajah yang terdapat pada

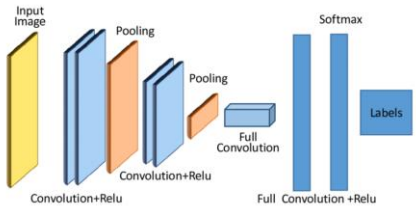
suatu citra dan mengidentifikasi wajah tersebut dengan wajah yang terdapat pada *dataset* [3]. Sistem ini dapat dimanfaatkan untuk mendeteksi dan membedakan wajah manusia pada kamera sehingga dapat membantu tugas manusia sehari-hari. Performa dari pengenalan wajah bergantung pada beberapa faktor seperti iluminasi, ekspresi, umur, gaya rambut, aksesoris, intensitas cahaya dan pergerakan. Maka dari itu proses pelatihan wajah harus memiliki *dataset* yang bervariasi pada setiap kelasnya guna memaksimalkan performa identifikasi wajah. Keberadaan *dataset* besar yang telah dianotasi dan alat dengan kemampuan komputer yang kuat menjadikan perkembangan pesat untuk teknologi *face recognition*. Implementasi teknologi ini dapat digunakan pada perangkat sehari-hari seperti kamera CCTV (Closed Circuit Television), *quadcopter*, dan *autonomous vehicles*.

C. *Face Tracking*

Face tracking merupakan sebuah teknologi yang dapat mendeteksi dan melacak keberadaan wajah manusia. Teknologi ini dapat digunakan pada video digital secara langsung ataupun secara jarak jauh [7]. *Face tracking* bekerja dengan menggunakan kamera atau video dan mendeteksi wajah yang terdapat pada video tersebut. Ketika wajah telah terdeteksi maka teknologi tersebut dapat menentukan dan mengikuti setiap pergerakan yang dilakukan pada wajah yang terdeteksi. Penggunaan *face tracking* bergantung kepada beberapa faktor yaitu kualitas citra pada video, *frame rate* video tersebut dan pencahayaan yang buruk. Pengimplementasian teknologi *face recognition* pada *face tracking* akan membuat penggunaan teknologi ini lebih efektif dan dapat dipergunakan untuk membantu pekerjaan manusia secara cerdas.

D. *Convolutional Neural Network* (CNN)

Convolutional Neural Network (CNN) merupakan salah satu bentuk dari ANN *Artifcial Neural Network* yang menggunakan *deep learning* untuk melakukan tugasnya. CNN adalah jaringan saraf yang dirancang untuk memproses data dua arah. CNN digunakan untuk analisis citra visual, mendeteksi, dan mengenali objek dalam citra, yang merupakan vektor berdimensi tinggi yang melibatkan sejumlah besar parameter untuk mengkarakterisasi jaringan. CNN biasanya digunakan untuk *image* dan *video recognition* serta dapat digunakan juga untuk *natural language processing* (NLP) [9]. Pada CNN terdapat dua proses, yaitu proses ekstraksi fitur dan proses klasifikasi fitur. Pada proses ekstraksi fitur terdapat *convolution layer, pooling layer*, dan *Rectifed Linear Unit* (ReLU) sebagai fungsi aktivasi. Kemudian pada proses klasifikasi fitur terdapat *fully connected layer* dan *softmax* sebagai fungsi aktivasi. Semakin banyak jaringan atau *convolution layer*, akan membuat gambar semakin kecil dan dalam. Gambar 1 menunjukan *layer* yang digunakan pada CNN [10].



GAMBAR 1
ARSITEKTUR CNN

1. *Convolutional Layer*

Convolutional layer adalah komponen utama dari CNN. *Convolutional layer* menggunakan kernel konvolusi sebagai *filter* untuk bergeser pada citra. Nilai dari setiap piksel yang dilewati oleh *filter* akan di digandakan dan ditambahkan kepada hasil konvolusi [11]. Persamaan dari proses konvolusi didapatkan dari Persamaan 1.

$$h(x) = f(x) * g(x) \quad (1)$$

dimana f(x) dengan g(x) adalah citra dari *input* dan *filter*. Penggunaan *stride* dan *padding* menjadi salah satu faktor dari hasil konvolusi. *Stride* sendiri merupakan jumlah pergeseran filter terhadap piksel pada suatu citra dan *padding* merupakan parameter yang digunakan untuk menambah jumlah piksel yang terdapat pada setiap sisi pada suatu citra sehingga pergeseran *filter* dapat dilakukan dengan optimal.

2. *Pooling Layer*

Pooling layer digunakan untuk mengurangi dimensi yang dihasilkan dari *convolution layer*. Ada tiga jenis teknik *pooling*, yaitu *general pooling, overlapping pooling and Spatial Pyramid Pooling* (SPP). Namun, yang paling sering digunakan ialah *general pooling* yang memiliki dua teknik, yaitu *max pooling* dan *average*. *Max pooling* merupakan teknik pooling yang mengambil nilai maksimum pada *filter* yang bergeser. Sementara pada *average pooling*, menggunakan nilai rata-rata nilai yang terdapat pada *filter*.

3. *Fully Connected Layer*

Fully-Connected Layer berfungsi untuk mengklasifikasi data masukan. *Feature map* yang dihasilkan dari *Convolution Layer* masih berbentuk multidimensi, sehingga harus dilakukan proses *flatten* atau *reshape feature map* untuk mentransformasikan data menjadi satu dimensi agar bisa dijadikan *input* pada *Fully-Connected Layer* dan dapat diklasifikasi secara linear. Saat proses *flatten*, nilai *input* matriks dari *layer* sebelumnya akan diubah menjadi vector. Dalam proses ini, ditambahkan metode *dropout* untuk menghindari *overftting* dengan cara menonaktifkan beberapa *edge* yang terhubung di setiap *neuron* [12].

E. You Only Look Once (YOLO)

You Only Look Once (YOLO) merupakan sistem deteksi objek secara *real time* yang unggul pada bidangnya. Awalnya YOLO pertama kali dipublikasi oleh Joseph redmon, YOLO secara cepat dapat menyaingi algoritma *deep learning* lainnya [13]. YOLO terus dikembangkan oleh pencipta aslinya sampai dengan versi YOLOv3, setelah itu perkembangan YOLO dilanjutkan oleh Alexey Bochkovsk untuk versi YOLOv4 dan Glenn Jocher pada YOLOv5. Algoritma YOLO bekerja dengan membagi nilai masukan dari sebuah citra menjadi sistem jaringan S × S dan setiap jaringan tersebut menjadi faktor utama untuk melakukan deteksi objek. Pada setiap sel didalam jaringan tersebut bertugas untuk memprediksi objek dan memberi tanda untuk setiap objek yang terdeteksi.

1. YOLOv5

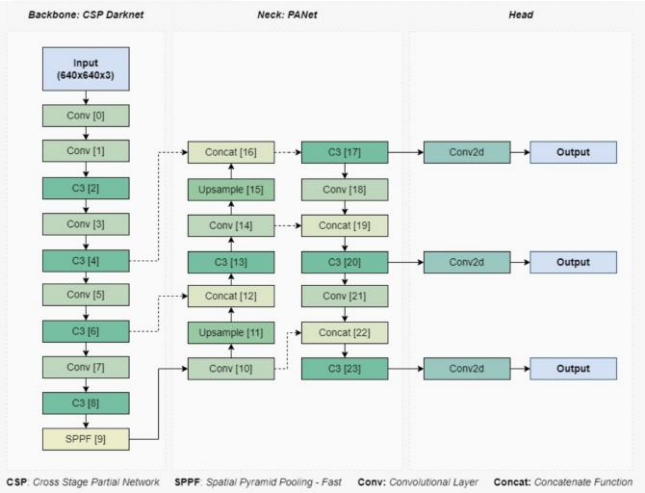
YOLOv5 merupakan versi terbaru dari model YOLO. Versi ini dikembangkan oleh perusahaan Ultralytics dan dirilis pada tahun 2020. YOLOv5 memiliki lima model utama yaitu YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l dan YOLOv5x. Keempat model tersebut memiliki karakteristik yang berbeda dari segi kecepatan inferensi, ukuran parameter, dan keakuratan model [14].

Pada proses deteksi objek, objek tersebut akan ditandai dengan *bounding box*. Keakuratan dari algoritma tersebut akan ditandai dengan *confidence score*, Semakin tinggi nilai *confidence score* hasil prediksi semakin akurat [13]. Untuk setiap *bounding box* yang diproduksi pada setiap objek yang terdeteksi terdapat beberapa parameter penting yaitu parameter x, y, w, h, dan *confidence score*. Nilai x dan y merupakan nilai koordinat *bounding box* yang telah diproduksi. Nilai w dan h merupakan lebar dan panjang dari *bounding box* tersebut dan *confidence score* merupakan nilai keakuratan objek yang telah terdeteksi.

2. Arsitektur YOLOv5

Arsitektur Yolov5 terbagi menjadi empat bagian yaitu *input*, *backbone*, *neck* dan *output*. Bagian *input* pada biasanya terdapat proses awal untuk datanya, seperti *mosaic data augmentation* dan *adaptive image flling*. Jaringan *backbone* menggunakan *cross-stage partial network* (CSP) dan *spatial pyramid pooling* (SPP) yang berfungsi untuk ekstraksi *feature maps* dari berbagai macam ukuran citra dengan beberapa konvolusi dan *pooling*. *BottleneckCSP* digunakan untuk mengurangi waktu komputasi dan kecepatan inferensi, sedangkan SPP digunakan untuk mengambil ekstraksi fitur dari beberapa skala yang berbeda untuk dijadikan *feature map*. Hal ini dapat mempengaruhi akurasi model. Pada bagian *neck* menggunakan struktur dari *feature pyramid network* (FPN) dan *persistence apparent network* (PAN). Kedua struktur ini dapat membuat ekstraksi fitur yang didapat dari *backbone* lebih efsien.Pada bagian *output* digunakan untuk memprediksi objek dari berbagai ukuran di *feature maps* [16].

YOLOv5 memiliki lima arsitektur utama dengan nama YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l dan YOLOv5x. Perbedaan dari tiap arsitekturnya adalah jumlah ekstraksi fitur dan kernel konvolusi pada jaringan konvolusi tersebut. Perbedaan tersebut disebabkan oleh nilai *compound scaling* pada tiap arsitekturnya. Struktur arsitektur Yolov5 ditunjukan pada Gambar 2.



GAMBAR 2
ARSITEKTUR CNN

F. Intersection Over Union (IOU)

Intersection over union (IOU) adalah sebuah istilah yang digunakan untuk mengukur nilai tumpang tindih antara dua kotak. Pada bidang deteksi objek IOU

digunakan untuk mengevaluasi *bounding box* prediksi dan *bounding box* yang sebenarnya. Hal ini menentukan seberapa benar prediksi yang dilakukan saat melakukan deteksi objek. Nilai IOU dapat dihitung dengan Persamaan 2.

$$IOU = \frac{AreaOfIntersection}{AreaOfUnion} \quad (2)$$

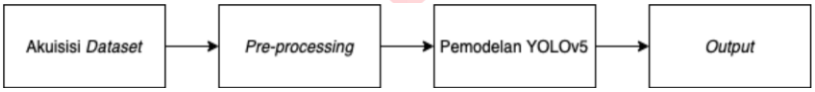
Pada Persamaan 2 menunjukan bahwa perolehan nilai IOU didapat dari cakupan antara kedua *bounding box* prediksi dan *ground truth* yang saling tumpeng tindih (*Area Of Intersection*) dibagi total luas dari kedua *bounding box* tersebut. Tujuan utama dari penggunaan IOU adalah untuk meningkatkan prediksinya sampai kedua *bounding box* tertumpang tindih secara sempurna, yaitu IOU antara dua kotak menjadi sama dengan 1.

III. METODE

Pada bagian ini berisi diagram blok penelitian yang dilakukan dan perancangan model menggunakan YOLOv5. Keluaran dari penelitian ini adalah pilihan model terbaik dari beberapa model yang dilakukan uji coba. Parameter performansi yang digunakan pada penelitian ini adalah *Mean Average Precision* (mAP), kecepatan inferensi dan jarak optimal deteksi.

A. Diagram Blok Penelitian

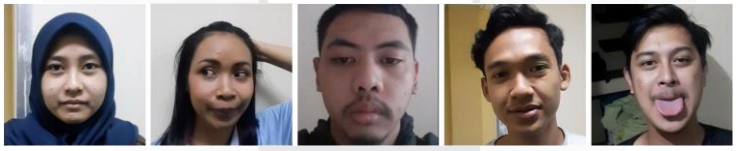
Penelitian ini menggunakan *deep neural network* dengan algoritma YOLOv5 untuk melakukan *face recognition*. Tahap pertama pada tugas akhir ini adalah akuisisi data. Akuisisi *dataset* bertujuan untuk mengumpulkan data berupa citra wajah dari beberapa variasi sudut pandang. Pengumpulan data dilakukan menggunakan kamera *smartphone*. Tahap selanjutnya yaitu *pre-processing* yang bertujuan untuk menyesuaikan *dataset* sebagai *input* untuk *training model*. Tahap ketiga yaitu pemodelan menggunakan YOLOv5 untuk melakukan deteksi wajah. Skema umum diagram blok sistem diilustrasikan pada Gambar 3.



GAMBAR 3
DIAGRAM BLOK SISTEM

1. Dataset

Dataset yang digunakan oleh sistem ini adalah citra dengan format *.png*. Proses akuisisi *dataset* dilakukan dengan pengambilan video menggunakan kamera *smartphone* dan mengambil setiap *frame* pada video tersebut untuk dijadikan satuan citra. Video diambil dari berbagai sudut sehingga tampak wajah yang menyeluruh. Setelah itu, dilakukan proses *labelling* sesuai dengan nama kelas yang diberikan. *Dataset* ini memiliki lima kelas dengan nama ”Bangga”, ”Dena”, ”Dewi”, ”Jaka”, dan ”Kurnivan”, dan dan didapati total 1728 citra pada seluruh kelasnya. Dari keseluruhan, *dataset* dibagi menjadi dua bagian yaitu *training set* (80%) , *validation set* (20%). Beberapa gambar yang terdapat di *dataset* dapat dilihat pada Gambar 4.



GAMBAR 4
CONTOH *DATASET* TIAP KELAS

2. Pre-processing

Pre-processing adalah proses awal dilakukannya perbaikan suatu citra untuk menyesuaikan *dataset* sebagai *input* untuk *training model*. Tahap ini akan diterapkan untuk seluruh dataset yang disiapkan sebelumnya. Metode *pre processing* yang digunakan pada tugas akhir ini adalah *resize* dan *auto-orient*. *Resize* bekerja dengan mengubah ukuran citra menjadi lebih besar atau lebih kecil dari ukuran sebelumnya. Proses ini dilakukan untuk mengubah dimensi pada suatu citra menjadi dimensi yang telah ditentukan, sehingga citra pada *dataset* menjadi proporsional. Pada proses ini citra ditransformasi sehingga memiliki jumlah dimensi 640 × 640. *Auto-orient* dilakukan untuk menjamin citra yang diunggah menjadi *dataset* sesuai dengan orientasi citra orisinalnya. Proses ini mengembalikan format *Exchangeable Image File* (EXIF) *dataset* yang telah diproses seperti format awal.

3. *Train* Model

Untuk memastikan pengaplikasian model *machine learning* ke *quadcopter* berjalan dengan optimal, ada faktor yang harus diperhatikan yaitu kecepatan inferensi pada video. Kecepatan menjadi salah satu faktor yang

penting dikarenakan *quadcopter* akan bergerak secara konstan sampai *quadcopter* menerima perintah untuk mendarat dari *device*. Maka dari itu, peristiwa *delay* harus dihindari. Pada tahap ini *dataset* dilatih dengan menggunakan algoritma YOLOv5 dengan beberapa model yang berbeda. Pada setiap pelatihan yang dilakukan, model tersebut diterapkan parameter yang sama. Konfgurasi parameter ditunjukan pada Tabel 1.

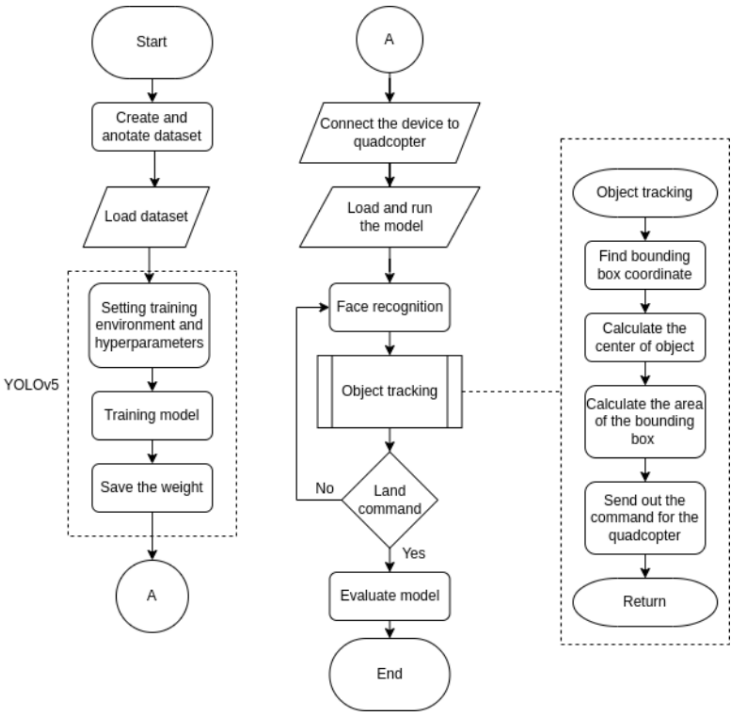
Tabel 1 Konfigurasi parameter *training* YOLOv5

Parameter	Nilai
Optimizer	*Stochastic Gradient Descent (SGD)*
Learning Rate	0,01
Epoch	150
Batch Size	16
Input Size	640

Pada tugas akhir ini dilakukan uji coba terhadap tiga model YOLOv5 yaitu YOLOv5n, YOLOv5s dan YOLOv5m. Uji coba ini bertujuan untuk mencari model YOLOv5 yang paling efektif untuk diimplementasikan pada *quadcopter*. Setiap model YOLOv5 memiliki ukuran parameter yang berbeda-beda sehingga menghasilkan perbedaan pada nilai mAP dan kecepatan inferensi. Tiap model YOLOv5 telah memiliki *pre-trained checkpoint* untuk digunakan saat proses *training*.

B. Implementasi Model Pada *Quadcopter*

Pada tahap ini dilakukan pengimplementasian model yang telah dilatih. Jenis *quadcopter* yang digunakan pada tugas akhir ini adalah Djitello. Jenis *quadcopter* ini memiliki kamera yang dapat dihubungkan melalui *wif*. Implementasi ditujukan untuk menambahkan fitur pengenalan wajah dan *object tracking*. Proses implementasi model dilakukan dengan menghubungkan *quadcopter* pada *device* yang sedang menjalankan inferensi YOLOv5 secara *real-time*. Lalu, perintah akan dikirimkan melalui program yang sudah didesain. Implementasi sistem ini ditunjukan seperti diagram alir pada Gambar 5.



GAMBAR 5
DIAGRAM ALIR IMPLEMENTASI SISTEM

1. *Face Recognition*

Pada tahap ini dilakukan penyambungan antara *device* dan *quadcopter*, penyambungan ini dilakukan dengan menggunakan jaringan *wif* yang dipancarkan oleh *quadcopter* dan disambungkan kepada *device*. Setelah itu, *device* akan menjalankan algoritma YOLOv5 dengan menggunakan hasil pelatihan *dataset* yang telah dilakukan dan menyambungkannya melewati *ip address* agar *device* dapat mengakses kamera dari *quadcopter* tersebut. *Input* berupa video *live stream* yang didapat dari kamera *quadcopter*, *input* ini digunakan untuk proses pengenalan wajah. Proses tersebut bergantung pada kondisi wajah yang tertangkap oleh kamera. Maka dari itu, dibutuhkan kondisi ruang dengan intensitas cahaya tinggi, kualitas kamera yang baik dan jarak yang optimal.

Pada proses *face recognition*, *device* dapat mengenali dan mengidentifikasi wajah sesuai *dataset* yang terlihat pada kamera. *Output* deteksi oleh YOLOv5 berupa *bounding box* yang mengelilingi objek terdeteksi serta *confidence score*. Pada setiap *bounding box* memiliki nilai ”xyxy” yang berisikan letak dan koordinat *bounding box* tersebut. Setiap objek yang terdeteksi memiliki *confidence score* yang menunjukan seberapa baik objek tersebut terdeteksi.

2. Object Tracking

Pada tahap ini dilakukan *object tracking* dengan menggunakan algoritma yang telah dibuat dan diintegrasikan pada YOLOv5. Cara kerja dari algoritma *tracking* tersebut adalah dengan membuat titik tengah pada *bounding box* yang telah mendeteksi wajah. Titik tengah tersebut menentukan perintah maneuver yang diberikan kepada *quadcopter*. Perintah manuver yang diberikan berdasarkan posisi titik tengah adalah rotasi dan perintah naik dan turun. Hal pertama yang dilakukan oleh algoritma ini adalah mencari koordinat wajah didalam *bounding box*. Pencarian nilai tersebut dilakukan dengan Algoritma 1.

Variabel ”det” merupakan *array* hasil deteksi yang telah dilakukan oleh algoritma YOLOv5. Variabel ”x” dan ”y” diambil dari dua array pertama pada variabel koordinat ”xyxy”. Sedangkan, untuk mencari nilai variabel ”w” dan ”h” diharuskan untuk mengkonversi format ”xyxy” terlebih dahulu ke format ”xywh”, lalu ketika sudah didapat format tersebut maka nilai variabel ”w” dan ”h” terdapat pada array ketiga dan keempat. Nilai ”xywh” tersebut dihitung untuk mendapatkan nilai titik tengah *bounding box* dan disimpan pada variabel ”cx” dan ”cy”, hal ini dilakukan untuk menemukan titik tengah koordinat objek yang terdeteksi dan berdasarkan dari letak koordinat tersebut algoritma dapat memberi nilai manuver kekiri atau kekanan. Setelah mendapat nilai manuver dan ukuran dari *bounding box* tersebut akan dilakukan perhitungan nilai variabel ”w” dan ”h” untuk mendapatkan luas *bounding box*. Perhitungan luas tersebut digunakan untuk mengetahui posisi wajah yang terdeteksi, Semakin besar ukuran *bounding box* akan mengindikasikan bahwa objek yang terdeteksi memiliki jarak yang dekat dengan *quadcopter* sehingga device akan memberikan perintah mundur untuk *quadcopter* dan sebaliknya. nilai ”cx” dan ”cy” akan disimpan berupa *array* pada variable ”myFaceListC” dan nilai luas dari *bounding box* akan disimpan pada variable ”myFaceListArea”. Setelah nilai manuver dan ukuran *bounding box* telah diketahui maka perintah untuk menggerakan *quadcopter* dilakukan sesuai dengan posisi wajah pada kamera.

Result: Mencari Koordinat Wajah
initialization;
$myFaceListC \leftarrow []$;
$myFaceListArea \leftarrow []$;
for $xyxy$ **to** det **do**
  x $\leftarrow xyxy[0]$
  y $\leftarrow xyxy[1]$
  w $\leftarrow xyxy[2]$
  h $\leftarrow xyxy[4]$
  cx $\leftarrow x + w$
  cy $\leftarrow y + h$
  area $\leftarrow w * h$
  myFaceListC.append $\leftarrow ([cx, cy])$
  myFaceListArea.append $\leftarrow [area]$
end
if $length.myFaceListArea \neq 0$ **then**
  $i \leftarrow myFaceListArea.index$
  return $[myFaceListC[i],\ myFaceListArea[i] \leftarrow info]$
end
else
  return $[[0,\ 0],\ 0] \leftarrow info$
end

Algorithm 1: Cara algoritma mencari koordinat wajah.

Setelah mendapatkan nilai koordinat *bounding box* wajah pada *array* ”myFaceListC” dan ”myFaceListArea”, selanjutnya adalah menghitung dan memberikan perintah agar *drone* bergerak sesuai koordinat. Sebelum algoritma ini dapat memberikan perintah dilakukan dahulu inisialisasi variabel, yaitu variable ”FbRange” sebagai penentu jarak minimum dan maksimum, ”fb” sebagai perintah maju dan mundur, ”speed” sebagai perintah rotasi, dan ”pid” sebagai penentu titik tengah pada *frame* inferensi. Nilai minimum ditunjukan oleh variabel ”fbRange” pada *array* ke 0 dan nilai maksimum ditunjukan pada *array* ke 1. Setelah mendapatkan nilai ”speed”, jika nilai tersebut lebih dari 0 maka akan dibulatkan ke 100 dan sebaliknya, jika nilai kurang dari 0 maka akan dibulatkan ke negatif 100. Pembulatan ini dilakukan untuk mengatur seberapa banyak *quadcopter* melakukan rotasi. Setelah mendapat nilai yang diperlukan maka algoritma *object tracking* akan memberikan perintah pergerakan untuk *quadcopter* sesuai dengan nilai variable ”fb” dan ”speed” yang didapat dari perhitungan sebelumnya. Algoritma *Object tracking* ditunjukan pada Algoritma 2.

$$\text{Recall} = \frac{TP}{TP + FN} \quad (4)$$

$$\text{mAP} = \frac{1}{N}\sum_{i=1}^{N} \text{AP}_i \times 100\% \quad (5)$$

Result: Memberikan perintah pada *drone* berdasarkan koordinat t
tengah
initialization;
$fbRange \leftarrow [15000, 20000]$;
$area \leftarrow info[1]$;
$x, y \leftarrow info[0]$;
$fb \leftarrow 0$;
$speed \leftarrow 0$;
$pid \leftarrow [0.4, 0.4, 0]$;
$speed \leftarrow pid[0]/2 * error + pid[1] * (error - pError)/2$;
$speed \leftarrow int(np.clip(speed, -100, 100))$;
if $area \geq fbRange[1]$ **then**
  Output: Menjauh ;
  $fb \leftarrow -20$;
end
else if $area \leq fbRange[0]$ **then**
  Output: Mendekati ;
  $fb \leftarrow 20$;
end
else if $area \leq fbRange[0] area \neq 0$ **then**
  Output: Wajah tidak terdeteksi ;
  $fb \leftarrow 20$;
end
$send_rc_control \leftarrow (0, fb, 0, speed)$;
return *error*
;

Algorithm 2: Cara algoritma memberikan perintah untuk *drone*.

$$AP = \sum_{n} [R_n - R_{n-1}]P_n \quad (6)$$

3. Performance Evaluation Metrics

Performance evaluation metrics merupakan satuan yang digunakan untuk menguji kualitas model *machine learning* yang telah dilatih [18]. Setelah proses pengujian terdapat nilai-nilai yang dapat dicari dengan membandingkan data latih dan data validasi sehingga menghasilkan tabel *confusion matrix*. Pada tugas akhir ini *performance evaluation metrics* yang dianalisa adalah mAP.

a. Precision

Precision merupakan penggambaran tingkat keakuratan data yang diminta dengan hasil prediksi yang diberikan oleh model. Maka, nilai *precision* dapat didefnisikan sebagai rasio prediksi benar positif dibandingkan dengan keseluruhan hasil yang diprediksi

$$\text{Precision} = \frac{TP}{TP + FP} \quad (3)$$

positif. Dari semua kelas positif yang telah di prediksi dengan benar, berapa banyak data yang benar benar positif. Nilai dari *precision* didapat dari Persamaan 3. Pada persamaan tersebut terdapat TP sebagai *True Positive* dibagi dengan TP ditambah FP sebagai *False Positive*.

b. Recall

Recall merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan data yang benar positif. Persamaan 4 dilakukan untuk mencari nilai *recall*. Maka dari itu, data yang positif merupakan hasil prediksi dari model yang dibuat dan divalidasi dengan data uji coba. Nilai *recall* dapat dihitung dengan Persamaan 4.

.

dengan TP merupakan *True Positive* dan FN merupakan *False Negative* [19].

c. Mean Average Precision

Mean Average Precision adalah nilai yang diambil dari hasil rata-rata *precision* dan di rata-rata kan dengan jumlah kelas. Nilai ini digunakan untuk mengevaluasi hasil pelatihan objek deteksi pada YOLOv5. Secara matematis, mAP dapat didefnisikan dengan Persamaan 5.

dengan N merupakan jumlah kelas, i merupakan iterasi dan AP merupakan nilai rata-rata dari *precision*. Nilai AP dapat dihitung dengan menggunakan Persamaan 6.

dengan *Rn* dan *Pn* merupakan *recall* dan *precision* pada saat *threshold* ke-n.

d. Jarak Optimal Berdasarkan *Confidence Score*

Confidence score menunjukan probabilitas dari objek yang dideteksi terlaksana dengan baik dengan memberikan nilai sesuai dengan persentasenya Istilah

ini menunjukan seberapa besar kemungkinan adanya objek didalam *bounding box*, Jika tidak ada objek maka nilai *confidence score* akan menunjukan nilai nol. Secara umum, *confidence score* akan menunjukan nilai yang lebih tinggi jika memiliki ambang batas IOU yang lebih ketat. *Confidence score* dapat dihitung dengan Persamaan 7.

$$ConfidenceScore = \Pr(class) \times IOU \times 100\% \quad (7)$$

dengan variabel Pr(class) adalah nilai probabilitas adanya sebuah objek perkelas dan IOU merupakan nilai *Intersection Over Union*.

e. Kecepatan Inferensi

Penilaian kecepatan inferensi dilakukan dengan menggunakan satuan *frame per second* (FPS). FPS adalah adalah jumlah *frame* yang berganti pada setiap detik. Parameter ini akan tampil secara automatis pada saat program dijalankan. Satuan ini digunakan untuk mengetahui seberapa cepat kemampuan model saat melakukan proses deteksi objek [20].

D. Spesifikasi Perangkat

Pada tugas akhir ini digunakan beberapa perangkat keras yaitu Dji Tello Drone dan *personal computer* (*device*). Dji Tello Drone merupakan jenis *quadcopter* yang memiliki kamera yang dapat diakses via *wif* sehingga memungkinkan untuk menjalankan program melalui *personal computer*. Laptop digunakan untuk perancangan sistem dan *training dataset* menggunakan server DGX A100. Perangkat lunak yang digunakan untuk proses *training* adalah *Jupyter Notebook*. Spesifkasi perangkat keras yang digunakan sebagai berikut:

1. Dji Tello Drone
    a. Photo : 5MP (2592x1936)
    b. Video : HD 720p30
    c. Dimensi : 98 x 92,5 x 41 mm
    d. Kapasitas baterai 1109 mAh dan tegangan 3,8 Volt
2. *Personal Computer*
    a. Processor: Intel® Core™ i7-9700K CPU @ 3.60 GHZ ×16
    b. RAM: 16 GB DDR4
    c. Graphics: NVIDIA GeForce RTX 2060
    d. OS: Ubuntu 20.04.2 LTS 64-bit
3. Nvidia DGX A100
    a. CPU: Dual AMD Rome 7742, 128 cores total, 2.25 GHz (base), 3.4 GHz (max boost)
    b. GPUs: 8x NVIDIA A100 Tensor Core GPUs
    c. GPU Memory: 320 GB total
    d. System Power Usage: 6.5kW max
    e. OS: Ubuntu 20.04.2 LTS 64-bit

IV. HASIL DAN PEMBAHASAN

Pada bagian ini dilakukan pengujian dan analisis terhadap sistem menggunakan algoritma YOLOv5 yang diimplementasikan pada *quadcopter*. Bagian ini juga memaparkan hasil percobaan dengan menggunakan parameter performansi mAP, jarak optimal deteksi, dan kecepatan inferensi. Pengujian ini menggunakan tiga ukuran model yang berbeda diantaranya: YOLOv5n, YOLOv5s, dan YOLOv5m.

A. Analisis mAP

Pada tugas akhir ini dilakukan proses pelatihan *dataset* dengan menggunakan tiga ukuran model YOLOv5. Pada percobaan ini perolehan nilai mAP@.95 tertinggi didapat pada model YOLOv5m dengan nilai 84,8% dan nilai mAP.95 terendah didapat pada model YOLOv5n dengan nilai 79,8%. Sementara, model YOLOv5s memiliki nilai mAP@.95 sebesar 82,4%. Ketiga model ini memiliki selisih nilai sebesar 2% pada setiap tingkatannya. Perbandingan nilai mAP@.95 dapat dilihat pada Gambar 6.

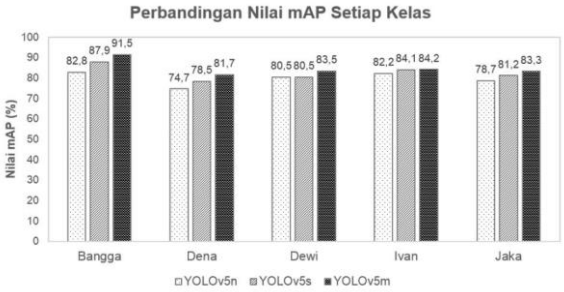
GAMBAR 6
PERBANDINGAN NILAI MAP

Perbedaan nilai disebabkan oleh bedanya lebar *layer* dan kedalaman kanal pada setiap model. Hal tersebut mempengaruhi komputasi model pada saat proses pelatihan sehingga *layer* dan kanal yang diproses memiliki ukuran yang berbeda pada setiap modelnya.

YOLOv5n memiliki lebar *layer* dan kedalaman kanal yang lebih kecil jika dibandingkan dengan kedua model lainnya. Maka dari itu, proses pelatihan tidak memiliki tingkat ketelitian yang tinggi.

1. Analisis Nilai mAP Setiap Kelas

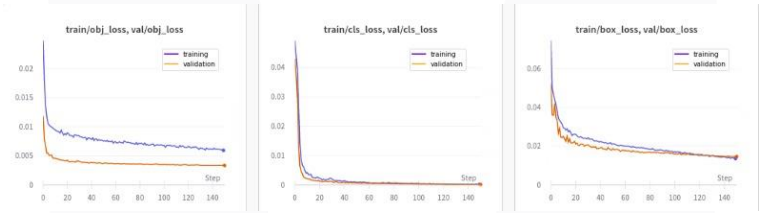
Pada Gambar 7 menunjukan hasil mAP dari setiap kelas untuk model YOLOv5n, YOLOv5s, dan YOLOv5m. Berdasarkan hasil tersebut dapat dilihat bahwa pada semua kelas dengan nilai mAP tertinggi terdapat pada model YOLOv5m dengan perolehan nilai mAP tertinggi terdapat pada kelas Bangga dengan nilai 91,5%. Sementara, kelas dengan perolehan nilai mAP terendah didapat pada kelas Dena dan model YOLOv5n dengan nilai mAP 74,7%. Hal ini disebabkan oleh perbedaan kondisi yang dimiliki *dataset* pada setiap kelasnya. Kondisi yang dapat mempengaruhi kualitas yaitu pencahayaan, penggunaan aksesoris, dan sudut pengambilan gambar.



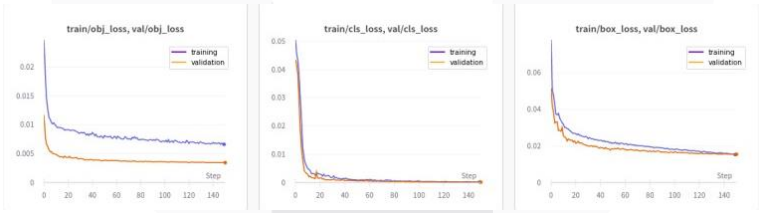
Gambar 7 Perbandingan Nilai mAP Tiap Kelas

4.2 Analisis Grafik *Loss*

Fungsi *loss* merupakan salah satu cara untuk mengevaluasi performa model objek deteksi. Nilai *loss* yang tinggi menandakan model tersebut mengalami banyak kesalahan saat melakukan predikisi. YOLOv5 menggunakan 3 nilai *loss* yaitu *box loss*, *classifcation loss*, dan *object loss. Box loss* menunjukan kemampuan model untuk memberikan *bounding box* terhadap objek yang telah terdeteksi dengan melakukan perbandingan dengan *ground truth*. *Classifcation loss* menunjukan seberapa baik model dalam memprediksi kelas. *Object loss* adalah probabilitas terdapat objek dalam sebuah gambar. Hasil dan analisis grafik *loss* dapat dilihat pada Gambar 8.



(a) YOLOv5n



(b) YOLOv5s

(c) YOLOv5m

GAMBAR 8
VISUALISASI GRAFIK *LOSS* PADA MODEL

Berdasarkan Gambar 8 pada kolom pertama, didapatkan hasil terhadap *objectness loss* pada tiga pelatihan model YOLOv5n, YOLOv5s dan YOLOv5m. Hasil pada dua model YOLOv5n dan YOLOv5s

didapatkan nilai yang sama, sehingga kedua model tersebut menunjukkan nilai *training loss* dan *validation loss* memiliki nilai yang cukup signifkan dibandingkan dengan grafik *loss* pada *box loss* dan *classifcation loss*. Selanjutnya hasil pada model YOLOv5m terdapat selisih yang signifkan pada nilai *training loss* dan *validation loss*, namun selisih tersebut terus berkurang seiring bertambahnya *epoch.*

Berdasarkan Gambar 8 terhadap *classifcation loss* yang diilustrasikan pada grafik terhadap tiga pelatihan model YOLOv5n, YOLOv5s dan YOLOv5m. Pada classifcation loss ini grafik memberikan hasil pada tiga pelatihan model yaitu memiliki perubahan dan nilai yang sama. Pada *epoch* 0 sampai 40 memiliki ketidakstabilan nilai *training* dan *validation loss* sehingga pada *epoch* selanjutnya terdapat penurun nilai yang stabil serta selisih nilai yang kecil.

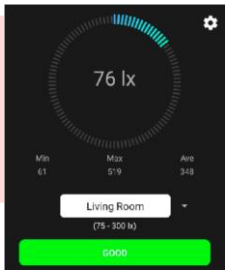
Berdasarkan Gambar 8 pada kolom ketiga, didapatkan hasil dari grafik *box loss* pada pengujian model. Terdapat tiga pengujian model yaitu YOLOv5n, YOLOv5s dan YOLOv5m. Pada tiga pengujian model ini, grafik menggambarkan hasil yang sama, namun pada model YOLOv5n memberikan nilai *training loss* dan *validation loss* yang sama. Grafik *box loss* yang dihasilkan oleh pengujian model YOLOv5m awalnya memiliki selisih antara *validation loss* dan *training loss*, namun pada *epoch* 100 sampai 150 dihasilkan *validation loss* telah melampaui nilai *training loss*. Sementara itu pada pengujian model YOLOv5s dihasilkan *epoch* 130 sampai 150 menunjukkan *validation loss* telah melampaui nilai *training loss*.

C. Analisis Jarak Optimal

Pada bagian ini dibandingkan jarak optimal deteksi dari tiga model yang digunakan dan diuji berdasarkan *confidence score* yang dihasilkan pada saat proses inferensi. Perbandingan ini dilakukan dengan melakukan inferensi video dari sisi pandang dan kamera *quadcopter*, lalu mengukur performa deteksi dari jarak yang berbeda. Jarak diukur dari rentang 2 meter sampai dengan 8 meter. Percobaan dilakukan dengan menggunakan 2 kondisi intensitas cahaya yang berbeda, intensitas cahaya dihitung dengan menggunakan aplikasi *light meter* pada *smartphone*.

1. Analisis Deteksi Jarak pada Waktu Sore Hari

Pada percobaan ini dilakukan pengujian deteksi jarak optimal pada waktu sore hari. Gambar 9 menunjukan nilai intensitas cahaya yang didapat dari aplikasi *lightmeter*. Pada percobaan ini memiliki nilai intensitas cahaya sebesar 76lx.



GAMBAR 9
NILAI INTENSITAS CAHAYA

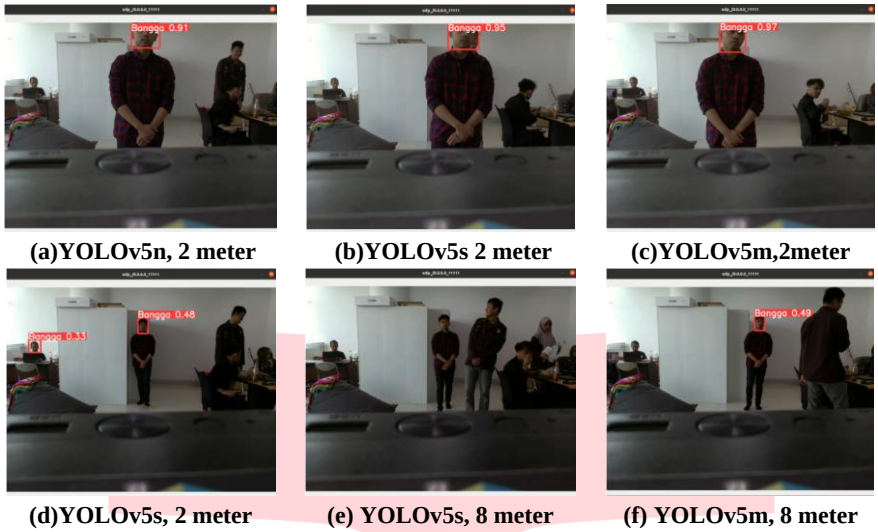
Confidence score yang dihasilkan menunjukan seberapa yakin model terhadap objek yang dideteksi. Dalam per-coban ini, jarak menjadi pengaruh penting pada kemampuan model untuk mengenali wajah yang tertangkap kamera. Pada jarak 2 meter, ketiga model memiliki performansi yang tinggi dengan *confidence score* diatas 90%. Pada jarak 4 meter, model YOLOv5n memiliki penurunan performansi, sedangkan model lain memiliki nilai yang konsisten. Pada jarak 6 meter dan 8 meter, penurunan performansi terjadi pada model YOLOv5n dan YOLOv5s secara signifkan dengan *confidence score* dibawah 50%. Sementara performansi model YOLOv5m pada jarak 6 meter menunjukkan hasil yang baik. Namun, pada jarak 8 meter terjadi penurunan performansi dengan *confidence score* dibawah 50%.

TABEL 2
NILAI *CONFIDENCE SCORE* TERHADAP JARAK

Model	*Confidence Score (2 meter)*	*Confidence Score (4 meter)*	*Confidence Score (6 meter)*	*Confidence Score (8 meter)*
YOLOv5n	91%	80%	49%	48%
YOLOv5s	95%	94%	57%	0%
YOLOv5m	95%	92%	90%	49%

Gambar 10 menunjukan perbandingan performansi setiap model terhadap jarak. Model YOLOv5n pada jarak 8 meter terdapat kesalahan deteksi wajah yang tidak termasuk di kelas *dataset* yang telah dilatih. Model YOLOv5s pada jarak 8 meter tidak

terdapat wajah yang terdeteksi. Hal ini menandakan bahwa kedua model tersebut tidak memiliki performansi yang baik untuk melakukan deteksi jarak jauh. Sementara, model YOLOv5m menunjukan performa yang baik terhadap jarak dekat maupun jauh.



(a)YOLOv5n, 2 meter **(b)YOLOv5s 2 meter** **(c)YOLOv5m,2meter**

(d)YOLOv5s, 2 meter **(e) YOLOv5s, 8 meter** **(f) YOLOv5m, 8 meter**

GAMBAR 10
HASIL PERBANDINGAN PERFORMANSI MODEL TERHADAP JARAK

2. Analisis Deteksi Jarak pada Waktu Siang Hari

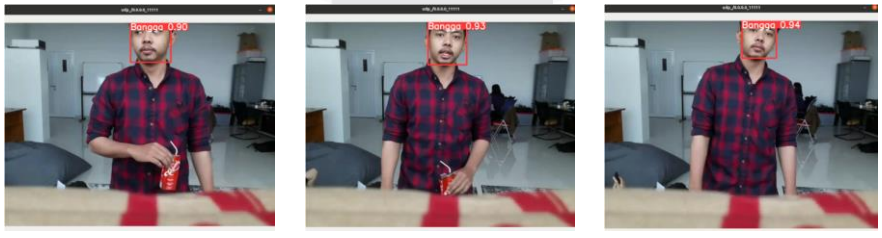
Pada percobaan ini dilakukan pengujian deteksi jarak optimal pada waktu siang hari. Gambar 11 menunjukan nilai intensitas cahaya yang didapat dari aplikasi*lightmeter*. Pada waktu pengujian didapat nilai *light meter* sebesar 396lx sehingga terdapat perubahan intensitas cahaya yang signifkan.

GAMBAR 11
NILAI INTENSITAS CAHAYA

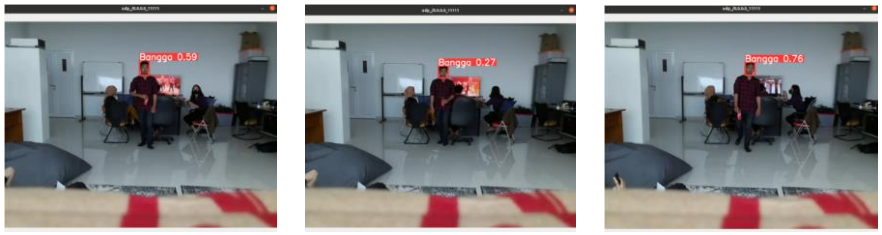
Pada Tabel 3 menunjukan hasil inferensi video untuk menguji *confidence score* yang didapat pada setiap jarak. Pada percobaan ini YOLOv5n mengalami penurunan performansi pada jarak 4 meter dan nilai kembali naik pada meter 6 dan 8 sehingga nilai yang dihasilkan tidak konsisten. Model YOLOv5s mengalami penurunan setelah 4 meter dan YOLOv5m memiliki konsistensi *confidence score* yang baik karena setiap meternya tidak memiliki nilai kurang dari 50%.

TABEL 3
NILAI *CONFIDENCE SCORE* TERHADAP JARAK

Model	*Confidence Score (2 meter)*	*Confidence Score (4 meter)*	*Confidence Score (6 meter)*	*Confidence Score (8 meter)*
YOLOv5n	90%	28%	56%	59%
YOLOv5s	93%	79%	55%	27%
YOLOv5m	94%	87%	77%	76%



(a)YOLOv5n, 2 meter **(b)YOLOv5s 2 meter** **(c)YOLOv5m,2meter**



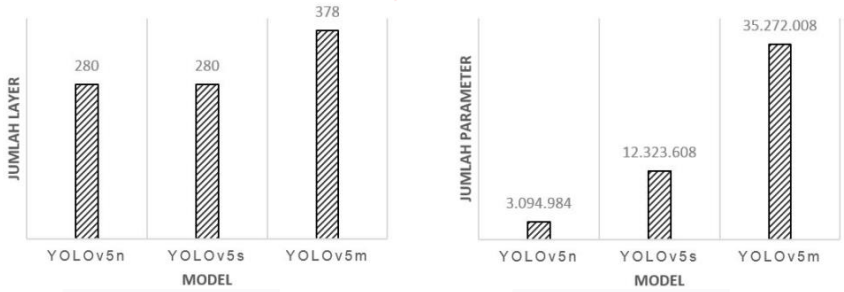
(d)YOLOv5s, 2 meter **(e) YOLOv5s, 8 meter** **(f) YOLOv5m, 8 meter**

GAMBAR 12
HASIL PERBANDINGAN PERFORMANSI MODEL TERHADAP JARAK

D. Analisis Kecepatan Inferensi

Pada analisis ini dibandingkan kecepatan inferensi yang dimiliki oleh ketiga model yang digunakan. Kecepatan inferensi didapat dengan menghitung berapa jumlah *frame* yang diproses oleh model pada setiap detik secara *real-time*. Kecepatan inferensi yang dihasilkan model menjadi kunci utama dalam kemampuan *drone* untuk melakukan *object tracking*.

Perbedaan kecepatan dari setiap model disebabkan oleh ukuran parameter yang berbeda. Model YOLOv5n dan YOLOv5s memiliki kedalaman kanal yang sama, namun model YOLOv5s memiliki *layer* yang lebih lebar dibandingkan dengan YOLOv5n. Sedangkan model YOLOv5m memiliki lebar *layer* dan kedalaman kanal yang lebih besar dari kedua model tersebut, yaitu YOLOv5n dan YOLOv5s. Perbedaan ukuran parameter ini menentukan jumlah *layer* dan jumlah parameter yang digunakan model, sehingga mempengaruhi waktu komputasi yang dilakukan untuk melakukan prediksi.



(a) (b)

GAMBAR 13
HASIL PERBANDINGAN JUMLAH *LAYER* DAN PARAMETER MODEL

Perbedaan jumlah *layer* dan parameter ditunjukan pada Gambar 13a dan Gambar 13b. Model YOLOv5n dan YOLOv5s memiliki banyak *layer* yang sama dengan jumlah 280. Sedangkan, model YOLOv5m memiliki jumlah *layer* sebanyak 378 sehingga memiliki selisih sebanyak 98 *layer* dari dua model sebelumnya. Parameter yang digunakan juga memiliki perbedaan di setiap modelnya. Model YOLOv5n memiliki 3094984 parameter, model YOLOv5s memiliki 12323608 parameter, sedangkan model YOLOv5m memiliki jumlah parameter 35272008. Kecepatan inferensi yang dihasilkan oleh setiap model dapat dilihat pada Tabel 4.

TABEL 4
KECEPATAN INFERENSI RATA-RATA MODEL DALAM SATUAN *MILLISECOND.*

Model	Kecepatan Inferensi (ms)
YOLOv5n	5
YOLOv5s	6
YOLOv5m	13

Kecepatan inferensi yang dihasilkan model dipengaruhi oleh kompleksitas *layer* yang dimiliki model dan kualitas *Graphic Processing Unit* (GPU) yang digunakan *device* untuk melakukan inferensi. Model YOLOv5n memiliki kecepatan inferensi yang konstan dengan nilai 5 ms per *frame*. Model YOLOv5s dan model YOLOv5n memiliki selisih yang kecil dengan nilai 6 ms per *frame*. Sedangkan, model YOLOv5m memiliki kecepatan sebesar 13 ms per *frame.*

V. KESIMPULAN

A. Kesimpulan

Pada tugas akhir ini, implementasi *face recognition* sebagai sistem kendali pada *quadcopter* dengan algoritma YOLOv5 dilakukan menggunakan 3 ukuran model. Model yang digunakan adalah YOLOv5n, YOLOv5s, dan YOLOv5m. Nilai mAP terbaik

didapatkan oleh model YOLOv5m sebesar 84%. Sedangkan, model dengan kecepatan inferensi tercepat didapat oleh model YOLOv5n dengan kecepatan 5ms per *frame*. Adapun beberapa kesimpulan yang diperoleh dari hasil pengujian dan analisis sebagai berikut:

1. Pada tugas akhir ini telah dilakukan implementasi *face recognition* dan *object tracking* dengan menggunakan algoritma YOLOv5 pada *quadcopter*.
2. Ukuran *layer* dan kedalaman kanal mempengaruhi kecepatan waktu komputasi dan kemampuan model untuk melakukan prediksi, Dikarenakan mempengaruhi jumlah parameter yang akan diproses.
3. Model yang paling baik digunakan untuk implementasi *face recognition* terhadap *quadcopter* adalah model YOLOv5s, dikarenakan memiliki selisih nilai mAP yang kecil jika dibandingkan dengan nilai mAP tertinggi dan juga memiliki kecepatan inferensi yang serupa dengan model YOLOv5n.
4. Ketiga model memiliki jarak optimal deteksi sejauh 8 meter, tetapi mengalami penurunan *confidence score* untuk setiap meternya.
5. Ketiga model memiliki performa yang baik pada kondisi intensitas cahaya yang tinggi dibandingkan dengan intensitas cahaya rendah.
6. Untuk kecepatan inferensi yang kurang baik dapat diatasi dengan menggunakan komponen GPU dengan kualitas yang lebih baik.

B. Saran

Saran yang bisa diberikan agar penelitian ini dapat dikembangkan adalah sebagai berikut:

1. Melakukan eksplorasi fitur *hyperparamter* untuk meningkatkan nilai mAP.
2. Menambahkan jumlah *dataset* yang memiliki kualitas dan variasi yang lebih banyak, sehingga kemampuan model untuk melakukan prediksi dapat dilakukan dilingkungan manapun.
3. Menggunakan komponen GPU dengan kualitas yang lebih baik dari yang digunakan pada tugas akhir ini.

Referensi

[1] H.-J. Hsu and K.-T. Chen, "Face recognition on drones," 05 2015, pp. 39–44.

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[3] R. Ranjan, A. Bansal, J. Zheng, H. Xu, J. Gleason, B. Lu, A. Nanduri, J.-C. Chen, C. D. Castillo, and R. Chellappa, "A fast and accurate system for face detection, identifcation, and verifcation," *IEEE Transactions on Biometrics, Behavior, and Identity Science*, vol. 1, no. 2, pp. 82–96, 2019.

[4] H.-J. Hsu and K.-T. Chen, "Face recognition on drones," 05 2015, pp. 39–44.

[5] Y. Fan, Y. Luo, and X. Chen, "Research on face recognition technology based on improved YOLO deep convolution neural network," *Journal of Physics: Conference Series*, vol. 1982, no. 1, p. 012010, jul 2021. [Online]. Available: https://doi.org/10.1088/1742-6596/1982/1/012010

[6] Droneomega. What is a quadcopter explained thoroughly. [Online]. Available: https://droneomega.com/what-is-a-quadcopter/

[7] Raydiant. Everything about face tracking. [Online]. Available: https: //sightcorp.com/knowledge-base/face-tracking/#:∼:text=Face%20Tracking%20Technology%20detects%20and,be%20used%20online%20or%20offline.

[8] P. Andono and T. Sutojo, *Pengolahan Citra Digital*. Penerbit Andi. [Online]. Available: "https://books.google.co.id/books?id=zUJRDwAAQBAJ"

[9] T. Contributor. convolutional neural network. [Online]. Available: https: //searchenterpriseai.techtarget.com/defnition/convolutional-neural-network

[10] T. H. Afridi. A multimodal memes classifcation: A survey and open research issues. [Online]. Available: https://www.researchgate.net/figure/ A-generic-CNN-Architecture fg1 344294512

[11] J. Du, "Understanding of object detection based on cnn family and yolo," *Journal of Physics: Conference Series*, vol. 1004, p. 012029, 04 2018.

[12] F. E. Ramadhan, "Penerapan image classifcation dengan pre-trained model mobilenet dalam client-side machine learning," B.S. thesis, Fakultas Sains dan Teknologi Universitas Islam Negeri Syarif Hidayatullah . . . , 2020

[13] M. Maithani. Guide to yolov5 for real-time object detection. [Online]. Available: https://analyticsindiamag.com/yolov5/

[14] X. Zhu, S. Lyu, X. Wang, and Q. Zhao, "Tph-yolov5: Improved yolov5 based on transformer prediction head for object detection on drone-captured scenarios," *CoRR*, vol. abs/2108.11539, 2021. [Online]. Available: https://arxiv.org/abs/2108.11539

[15] G. Jocher. Yolov5. [Online]. Available: https://github.com/ultralytics/yolov5

[16] B. Yan, P. Fan, X. Lei, Z. Liu, and F. Yang, "A real-time apple targets detection method for picking robot based on improved yolov5," *Remote Sensing*,

vol. 13, no. 9, 2021. [Online]. Available: https: //www.mdpi.com/2072-4292/13/9/1619

[17] python. What is python? executive summary. [Online]. Available: https://www.python.org/doc/essays/blurb/

[18] M. G. Meysam Vakili and M. Rezae, "Performance analysis and comparison of machine and deep learning algorithms for iot data classifcation," *Materials Today: Proceedings*, 01 2020.

[19] V. Meel. How to analyze the performance of machine learning models. [Online]. https://viso.ai/deep-learning/analyzing-machine-learning-model-performance-strategies/

[20] A. chaurasia. How are your yolov5 models doing? [Online]. Available: https://wandb.ai/ayush/yoloV5/reports/ How-are-your-YOLOv5-models-doing---VmlldzoyNjM3MTY