SISTEM DETEKSI GESTUR JARI TANGAN MENGGUNAKAN MEDIAPIPE DAN FASTER-RCNN UNTUK MENGONTROL KECEPATAN KIPAS ANGIN

LAPORAN TUGAS AKHIR

Program Studi

S1 TEKNIK KOMPUTER

Oleh:

MUHAMMAD ALDI FAKHRUDDIN

19410200015

FAKULTAS TEKNOLOGI DAN INFORMATIKA

UNIVERSITAS DINAMIKA

2023

SISTEM DETEKSI GESTUR JARI TANGAN MENGGUNAKAN MEDIAPIPE DAN FASTER-RCNN UNTUK MENGONTROL KECEPATAN KIPAS ANGIN

TUGAS AKHIR

Diajukan sebagai salah satu syarat untuk menyelesaikan
Program Sarjana Teknik

Disusun Oleh:

Nama : **Muhammad Aldi Fakhruddin**

NIM : **19410200015**

Program : **S1 (Strata Satu)**

Jurusan : **Teknik Komputer**

FAKULTAS TEKNOLOGI DAN INFORMATIKA

UNIVERSITAS DINAMIKA

2023

TUGAS AKHIR

SISTEM DETEKSI GESTUR JARI TANGAN MENGGUNAKAN MEDIAPIPE DAN FASTER-RCNN UNTUK MENGONTROL KECEPATAN KIPAS ANGIN

Dipersiapkan dan disusun oleh:

Muhammad Aldi Fakhruddin

NIM: 19410200015

Telah diperiksa, dibahas, dan disetujui oleh Dewan Pembahas

Pada: 9 Januari 2023

Susunan Dewan Pembahas

Pembimbing:

I. **Heri Pratikno, M.T., MTCNA., MTCRE.**
NIDN 0716117302

II. **Musayyanah, S.ST., M.T.**
NIDN 0730069102

Digitally signed by Heri Pratikno
Date: 2023.01.09 16:17:01 +07'00'

Pembahas:

Weny Indah Kusumawati, S.Kom., M.MT
NIDN 0721047201

Universitas Dinamika
2023.01.09 16:35:25 +07'00'

Tugas Akhir ini telah diterima sebagai salah satu persyaratan

untuk memperoleh gelar sarjana

Digitally signed by Universitas Dinamika
Date: 2023.01.18 07:18:16 +07'00'

Tri Sagirani, S.Kom., M.MT.

NIDN: 0731017601

Dekan Fakultas Teknologi dan Informatika

UNIVERSITAS DINAMIKA

"Jika orang tidak berani mengambil resiko, ia termasuk orang-orang yang merugi"

-Muhammad Aldi Fakhruddin-

Atas berkat Tuhan yang Maha Esa,

Ku persembahkan kepada keluarga saya untuk dukungannya selama ini dan doa tiada hentinya. Bersama teman-teman Teknik Komputer Angkatan 2019. Terimakasih atas bantuan doa, dukungan yang membuat saya lebih semangat dan terus belajar menjadi pribadi yang lebih baik lagi.

PERNYATAAN

PERSETUJUAN PUBLIKASI DAN KEASLIAN KARYA ILMIAH

Sebagai mahasiswa **Universitas Dinamika**, Saya :

Nama	: Muhammad Aldi Fakhruddin
NIM	: 19410200015
Program Studi	: S1 Teknik Komputer
Fakultas	: Fakultas Teknologi dan Informatika
Jenis Karya	: Laporan Tugas Akhir
Judul Karya	: **SISTEM DETEKSI GESTUR JARI TANGAN MENGGUNAKAN MEDIAPIPE DAN FASTER-RCNN UNTUK MENGONTROL KECEPATAN KIPAS ANGIN**

Menyatakan dengan sesungguhnya bahwa :

1. Demi pengembangan Ilmu Pengetahuan, Teknologi dan Seni, Saya menyetujui memberikan kepada **Universitas Dinamika** Hak Bebas Royalti Non-Eksklusif (***Non-Exclusive Royalty Free Right***) atas seluruh isi/sebagian karya ilmiah Saya tersebut diatas untuk disimpan, dialihmediakan, dan dikelola dalam bentuk pangkalan data (***database***) untuk selanjutnya didistribusikan atau dipublikasikan demi kepentingan akademis dengan tetap mencantumkan nama Saya sebagai penulis atau pencipta dan sebagai pemilik Hak Cipta.
2. Karya tersebut diatas adalah hasil karya asli Saya, bukan plagiat baik sebagian maupun keseluruhan. Kutipan, karya, atau pendapat orang lain yang ada dalam karya ilmiah ini semata-mata hanya sebagai rujukan yang dicantumkan dalam Daftar Pustaka Saya.
3. Apabila dikemudian hari ditemukan dan terbukti terdapat tindakan plagiasi pada karya ilmiah ini, maka Saya bersedia untuk menerima pencabutan terhadap gelar kesarjanaan yang telah diberikan kepada Saya.

Demikan surat pernyataan ini Saya buat dengat sebenar-benarnya.

Surabaya, 12 Desember 2022

Muhammad Aldi Fakhruddin
NIM : 19410200015

ABSTRAK

Dalam Perkembangan teknologi saat ini mendorong manusia untuk berkreasi sesuatu yang membuat pekerjaan menjadi lebih mudah. Perkembangan teknologi mengalami kemajuan yang sangat pesat. Mikrokontroler sekarang berkembang pesat dan semakin diminati dalam aplikasi sistem kontrol. Salah satu modul mikrokontroler yang paling umum adalah Arduino. Arduino adalah papan mikrokontroler yang merupakan "sistem komputer" bekerja pada chip. *Computer Vision* adalah sebuah keilmuan yang dapat memungkinkan komputer untuk mendeteksi dan melihat objek atau benda disekitarnya. Pada penelitian Tugas Akhir ini, untuk mengimplementasikan *framework* Mediapipe dan Faster-RCNN sebagai alat pengontrol kecepatan putaran kipas angin menggunakan Arduino melalui proses deteksi dengan bentuk gestur jari tangan, adapun gestur jari tangan yang akan diimplementasikan adalah hanya tiga jari saja. Pada pengujian Mediapipe, dapat diketahui bahwa rata-rata akurasi gestur jari 0 pada jarak 10 cm sebesar 37%, jarak 50 cm sebesar 70%, jarak 100 cm akurasinya sebesar 54.7%, dan jarak 175 cm sebesar 63.3%. Untuk pengujian Faster-RCNN, dapat diketahui bahwa akurasi gestur jari 0 pada jarak 10 cm sebesar 34%, jarak 50 cm sebesar 24%, jarak 100 cm akurasinya sebesar 8.7%, dan jarak 175 cm sebesar 4.7%. Nilai rata-rata akurasi gestur jari 0, membuktikan proses komputasi yang dijalankan metode Mediapipe ringan daripada proses komputasi metode Faster-RCNN.

Kata kunci : Sistem Kontrol, Arduino, Deep Learning, Mediapipe, Faster-RCNN.

KATA PENGANTAR

Puji syukur saya panjatkan pada Tuhan Yang Maha Esa atas segala rahmat yang telah diberikan-Nya, sehingga penulis dapat menyelesaikan Laporan Tugas Akhir yang berjudul "Sistem Deteksi Gestur Jari Tangan menggunakan Mediapipe dan Faster-RCNN untuk Mengontrol Kecepatan Kipas Angin". Dalam perjalanan menyelesaikan pengerjaan Laporan Tugas Akhir ini penulis banyak mendapatkan bantuan dari berbagai pihak. Oleh karena itu penulis mengucapkan terima kasih kepada:

1. Allah SWT, karena dengan rahmat-Nya penulis dapat menyelesaikan Laporan Tugas Akhir ini.
2. Orang tua dan seluruh keluarga yang telah memberikan dorongan dan dukungan baik secara moril maupun materiil, sehingga penulis dapat menempuh dan menyelesaikan Laporan Tugas Akhir ini.
3. Ibu Tri Sagirani, S.Kom., M.MT., selaku Dekan Fakultas Teknologi dan Informatika (FTI) Universitas Dinamika.
4. Bapak Pauladie Susanto, S.Kom., M.T., selaku Ketua Program Studi S1 Teknik Komputer Universitas Dinamika.
5. Ibu Weny Indah Kusumawati, S.Kom., M.MT., selaku dosen pembahas. Penulis mengucapkan terimakasih atas bimbingan yang diberikan dan kesempatan serta tuntunan baik secara lisan maupun tertulis, sehingga penulis dapat menyelesaikan Tugas Akhir.
6. Bapak Heri Pratikno, M.T., MTCNA., MTCRE., selaku dosen pembimbing yang telah memberikan dukungan berupa motivasi, wawasan, dan saran bagi penulis selama pelaksanaan pengerjaan Tugas Akhir dan dalam pembuatan laporan Tugas Akhir.
7. Ibu Musayyanah, S.ST., M.T.., selaku dosen pembimbing yang banyak memberikan masukan dan solusi agar Tugas Akhir ini menjadi lebih baik dan penulis dapat menyelesaikan Tugas Akhir ini.

8. Seluruh rekan–rekan S1 Teknik Komputer angkatan 2019 yang telah memberikan dukungan dan semangatnya untuk membantu penulis menyelesaikan laporan Tugas Akhir ini.
9. Seluruh pihak yang tidak dapat disebutkan satu per satu yang telah memberikan dukungan serta bantuan dalam segala bentuk yang akhirnya terselesaikannya laporan Tugas Akhir ini.

Penulis berharap semoga laporan ini dapat berguna dan bermanfaat untuk menambah wawasan bagi pembacanya. Penulis juga menyadari dalam penulisan laporan ini banyak terdapat kekurangan. Oleh karena itu, penulis sangat mengharapkan saran dan kritik untuk memperbaiki kekurangan dan berusaha untuk lebih baik lagi.

Surabaya, 9 Januari 2023

Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

DAFTAR LAMPIRAN

Halaman

BAB I
PENDAHULUAN

1.1 Latar Belakang

Dalam perkembangan teknologi saat ini mendorong manusia untuk berkreasi sesuatu yang membuat pekerjaan menjadi lebih mudah. Perkembangan teknologi mengalami kemajuan yang sangat pesat. Pesatnya perkembangan teknologi di era globalisasi saat ini membawa banyak manfaat untuk kemajuan dalam berbagai aspek. Perkembangan. Ini berkat usaha orang-orang yang selalu penasaran dan ingin melakukannya, jadi semuanya lebih praktis. Saat ini, orang tidak ingin melakukan apapun dan melakukan segalanya dengan cepat tanpa menghabiskan banyak energi dan membuat pekerjaan menjadi lebih mudah. (Arisandi, 2014), maka dari itu muncul mikrokontroler sebagai bentuk teknologi yang memudahkan pekerjaan manusia.

Mikrokontroler sekarang berkembang pesat dan semakin diminati dalam aplikasi sistem kontrol. Salah satu modul mikrokontroler yang paling umum adalah Arduino. Arduino adalah sejenis papan berisi mikrokontroler. Arduino adalah papan mikrokontroler yang merupakan “sistem komputer” bekerja pada chip. Arduino memiliki prosesor, memori, input/output, dan mikrokontroler ini dapat dikatakan versi komputer mini dengan perangkat lunak pendukung untuk pemrograman pada Arduino IDE (*Integrated Development Environment*). Mikrokontroler keluarga ATMEL banyak digunakan oleh para pelajar atau mahasiswa di Indonesia. Dengan kemudahan pemrogramannya dan harganya yang cukup terjangkau menjadi alasan pemilihan mikrokontroler jenis ini.

Computer Vision adalah sebuah keilmuan yang dapat memungkinkan komputer untuk mendeteksi dan melihat objek atau benda disekitarnya. Dengan kemampuan tersebut, komputer mampu menganalisis sendiri benda yang ada di depannya, sehingga informasi tersebut menghasilkan perintah tertentu. Salah satu jenis *Computer Vision* adalah OpenCV. OpenCV (*Open Computer Vision Library*) adalah *library* perangkat lunak bersifat *open source* yang memiliki kegunaan masing–masing, seperti mendeteksi sebuah objek, mengenali wajah, identifikasi objek, mendeteksi gerakan tangan, dan lain-lain. (Perdananto, 2019). *Deep Learning* yang digunakan pada penelitian ini adalah *Faster-RCNN*.

Sekarang sudah banyak sistem kontrol yang menggunakan Arduino untuk kebutuhan manusia, tetapi untuk *computer vision* dan *deep learning* masih belum banyak diterapkan dalam banyak hal. Padahal saat ini *computer vision* sudah dapat terhubung dengan mikrokontoroller. Hal ini dapat dijadikan sebagai peluang untuk menemukan solusi dari sebuah permasalahan. Selain itu, penggunaan teknologi *framework* Mediapipe dapat digunakan secara efisien sebagai alat untuk mendeteksi gerakan tangan yang rumit secara tepat. Karena kemampuannya Fitur ekstraksi *landmark* dari Mediapipe cocok untuk mendeteksi gestur tangan, seperti pada penelitian Muhammad Rifki Pratama Nautica pada tahun 2022 yang berjudul *Hand Gestur Detection* sebagai alat bantu ajar berhitung menggunakan Medipipe dan *Convolutional Neural Networ*k secara *realtime* (Nautica, 2022).

Selain itu, pada jurnal Derry Alamsyah pada tahun 2022 yang berjudul *Deteksi Ujung Jari menggunakan Faster-RCNN dengan Arsitektur Inception v2 pada Citra Derau,* (Alamsyah, 2022), sistem deteksi ujung jari ini dirancang untuk mengukur akurasi pada metode Faster-RCNN. Penelitian ini juga mengambil referensi jurnal Pensi Asmaleni pada tahun 2020 yang berjudul *Pengembangan Sistem Kontrol Kipas Angin dan Lampu Otomatis Berbasis Saklar Suara Menggunakan Arduino Uno,* (Asmaleni, 2020), mengembangkan sistem kontrol kipas angin dan lampu secara otomatis yang dapat meringankan pekerjaan manusia serta menghemat listrik. Dari penelitian dan jurnal tersebut, melakukan deteksi gestur jari tangan menggunakan Mediapipe dan Faster-RCNN. Penelitian tersebut membahas mengenai kontrol kipas angin pada saklar.

Pada penelitian Tugas Akhir ini, peneliti mengutip dari penelitian (Nautica, 2022), (Alamsyah, 2022), (Asmaleni, 2020), untuk mengimplementasikan *framework* Mediapipe dan Faster-RCNN sebagai alat pengontrol kecepatan putaran kipas angin menggunakan Arduino melalui proses deteksi dengan bentuk gestur jari tangan, adapun gestur jari tangan yang diimplementasikan adalah hanya tiga jari saja mengikuti jumlah kecepatan putaran kipas angin yang ada dipasaran.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas, maka dapat dirumuskan masalah pada Tugas Akhir ini sebagai berikut:

1. Bagaimana menerapkan mediapipe pada sistem deteksi jari tangan?
2. Bagaimana menerapkan Faster-RCNN pada sistem akurasi klasifikasi deteksi jari tangan?
3. Bagaimana mengontrol kecepatan kipas angin secara *Computer Vision* melalui deteksi bentuk gestur jari tangan?
4. Bagaimana membangun komunikasi serial antara *Computer Vision* untuk Deep learning dan mikrokontroler Arduino?

1.3 Batasan Masalah

Dalam pembuatan Tugas Akhir ini, pembahasan masalah dibatasi pada beberapa hal berikut:

1. Pengujian dilakukan dengan *Computer Vision* yang dihubungkan ke Arduino dengan USB kabel menggunakan komunikasi serial.
2. Pencahayaan yang merata pada ruangan.
3. Sistem deteksi gestur jari tangan hanya menggunakan jari tangan kanan.
4. Gestur angka jari tangan yang dideteksi terbatas dari 0 hingga 3.

1.4 Tujuan

Berdasarkan latar belakang dan rumusan masalah diatas, mendapatkan tujuan pada Tugas Akhir ini sebagai berikut:

1. Mampu mengetahui besar performa pada sistem deteksi jari tangan menggunakan Mediapipe.
2. Mampu mengetahui besar akurasi klasifikasi pada sistem deteksi jari tangan menggunakan Faster-RCNN.
3. Mampu mengontrol kecepatan putaran kipas angin secara *Computer Vision* melalui deteksi bentuk gestur jari tangan.
4. Mampu berkomunikasi *secara serial* antara *Computer Vision untuk Deep learning* dengan mikrokontroler *Arduino.*

1.5 Manfaat

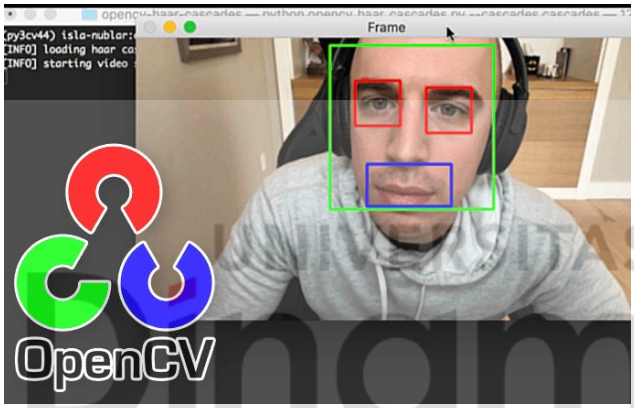
Adapun dari Tugas Akhir ini dapat diperoleh manfaat sebagai berikut:

1. Bagi penulis yaitu dapat menambah pengetahuan tentang *Computer Vision* dan deep learning dengan menghubungkan Arduino sebagai antarmuka.
2. Bagi mahasiswa yaitu dapat mengembangkan pengetahuan tentang *Computer Vision* dan deep learning dengan menghubungkan Arduino sebagai antarmuka.
3. Memudahkan manusia dalam mengontrol kecepatan putaran kipas angin hanya dengan gestur jari tangan.
4. Dapat menjadi referensi bagi mahasiswa yang linier atau sama.

BAB II
LANDASAN TEORI

2.1 OpenCV

OpenCV *(Open Computer Vision Library)* adalah yaitu pustaka perangkat lunak yang digunakan untuk citra digital yang bisa secara real-time yang dibuat oleh Intel. Bisa disebut juga sebagai pustaka perangkat lunak sumber terbuka dengan lisensi produk berlisensi BSD. OpenCV memiliki lebih dari 2500 jenis algoritme yang dioptimalkan dan tersedia untuk memenuhi kebutuhan visi komputer dan pembelajaran mesin. Berbagai algoritma OpenCV memiliki aplikasinya masing-masing, seperti deteksi wajah, deteksi gerakan tangan, deteksi objek dan lain-lain (Ramdhon, 2021).



Gambar 2.1 Contoh penerapan OpenCV
(Sumber: (Rosebrock, 2021))

2.2 Python

Python adalah bahasa pemrograman yang ditafsirkan yang dapat diterapkan pada berbagai kebutuhan pemrograman, dengan filosofi desain yang berfokus pada keterbacaan kode, dan juga merupakan salah satu bahasa yang umum digunakan dalam ilmu data, pembelajaran mesin, dan Internet (Zein, 2018). Pada penelitian ini, bahasa pemrograman Python digunakan sebagai bahasa utama untuk melakukan proses *training dataset* dan proses deteksi gestur jari tangan.

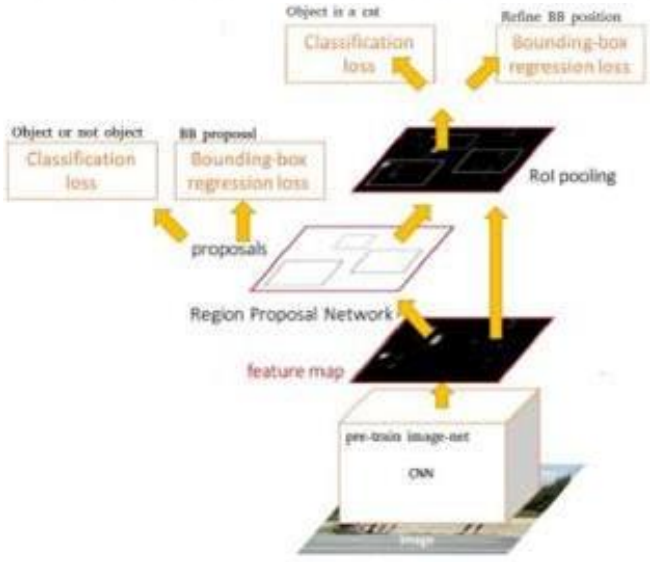
Gambar 2.2 Bahasa pemograman Python.
(Sumber: (Python.org, n.d.))

2.3 Deep Learning

Deep Learning merupakan sebuah varian *Machine Learning* yang mempunyai sebuah fungsi *training* pada sebuah komputer untuk menjalankan sebuah perintah secara spesifik. Adapun perintah-perintah yang dapat dilakukan berupa deteksi gambar, identifikasi gambar atau mengimplementasikan sebuah prediksi (Arsal, 2020). Dalam penelitian ini, *Deep Learning* digunakan sebagai proses utama pelatihan dan deteksi model pada program. Adapun jenis dari Deep Learning yang digunakan pada penelitian ini adalah Faster-RCNN.

2.4 Faster-RCNN

Faster-RCNN adalah bentuk pembelajaran mesin yang memiliki kemampuan untuk melatih komputer pribadi untuk melakukan perintah tertentu. Perintah yang dapat dieksekusi berupa pengenalan citra, identifikasi citra, atau implementasi prediksi (Arsal, 2020). Dalam penelitian ini, pembelajaran mendalam digunakan sebagai proses utama untuk membuat dan mendeteksi contoh dalam program. Jenis deep learning yang digunakan dalam penelitian ini adalah Faster-RCNN (Everitt, 2018).

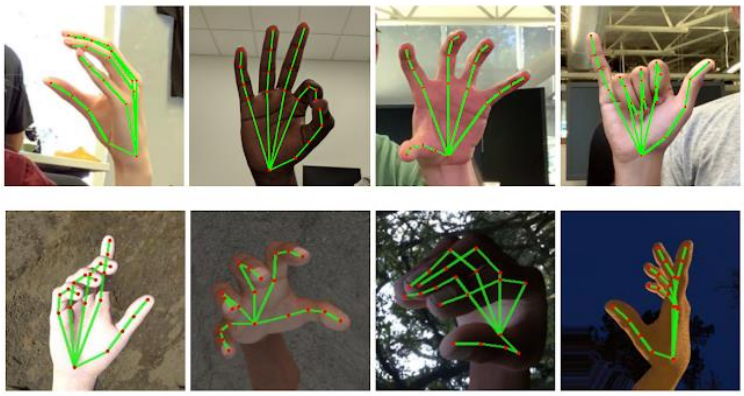


Gambar 2.3 Arsitektur Faster-RCNN
(Sumber: (Everitt, 2018))

Gambar 2.3 untuk mencari pada input gambar kemungkinan lokasi dari obyek. Posisi dari obyek yang ada pada gambar 2.3 mempunyai kemungkinan obyek yang dibatasi dari wilayah diketahui sebagai *region of interest* (ROI). RPN mengambil gambar diberbagai ukuran sebagai input dan ouput sekumpulan proposal obyek persegi panjang, masing-masing dengan skor obyektivitas. RPN yaitu proses deteksi yang lebih dalam pada daerah yang bukan dari gambar asli tetapi dari daerah peta fitur yang dihasilkan oleh CNN. Pada *Region Proposal Network* awalnya gambar dimasukkan ke dalam jaringan *Convolutional Neural Network*. Gambar input tadi diteruskan ke jaringan *convolutional layer* akhir yang menampilkan *feature map*. *Sliding window* ditempatkan disetiap bagian dari *feature map* (Tanner, 2019).

2.5 Mediapipe

Mediapipe adalah kerangka kerja yang dikembangkan oleh Google yang memungkinkan membuat garis pipa *(pipelines)* untuk memproses data observasi dari berbagai format audio dan video. Mediapipe dirancang untuk mereka yang ingin menerapkan kecerdasan buatan pada aplikasi mereka. Google telah menggunakan Mediapipe untuk penggunaan internal sejak 2012 dan merilisnya sebagai open source pada 2019. Kerangka kerja ini menawarkan berbagai solusi pembelajaran mesin seperti pengenalan wajah, iris, segmentasi rambut, holistik, dan lainnya, yang dapat dilihat di resmi Mediapipe situs web. Solusi yang ditawarkan kompatibel dengan sistem operasi Android dan iOS serta bahasa C++, Python, JS dan Coral (Mediapipe, 2019).



Gambar 2.4 Bentuk *Landmark* Mediapipe
(Sumber: (Mediapipe, 2019))

2.6 Arduino IDE

Arduino *IDE* merupakan kependekan daripada *Integrated Development Environment* berupa *software* penulisan program untuk melakukan fungsi-fungsi yang dibenamkan pada lingkungan pengembangan teritegrasi, *compile* dan *upload* program ke board arduino. Arduino menggunakan bahasa pemrograman C yang telah dimodifikasi yang biasa disebut dengan pemrograman C untuk Arduino.

Bahasa pemrograman *Arduino IDE* sudah dirubah sedemikian rupa untuk memudahkan pemula dalam melakukan penulisan sintaks pemrograman dari bahasa aslinya. *Arduini IDE* dibuat dari bahasa pemrograman java dilengkapi dengan *library* C/C++ yang biasa disebut *wiring* sehingga membuat operasi input dan output lebih mudah.

2.7 Arduino Uno

Arduino/Genuino Uno adalah papan mikrokontroler berbasis *ATmega328P* (datasheet). Ini memiliki 14 pin *input/output* digital (6 di antaranya dapat digunakan sebagai *output PWM*), 6 input analog, kristal kuarsa 16 MHz, koneksi USB, colokan listrik, *header ICSP*, dan tombol reset. Ini berisi semua yang diperlukan untuk mendukung mikrokontroler; cukup sambungkan ke komputer dengan kabel USB atau nyalakan dengan adaptor AC-ke-DC atau baterai untuk memulai. (IDE).

Gambar 2.5 Arduino Uno
(Sumber: (Khumaidi, 2019))

2.8 Modul Relay

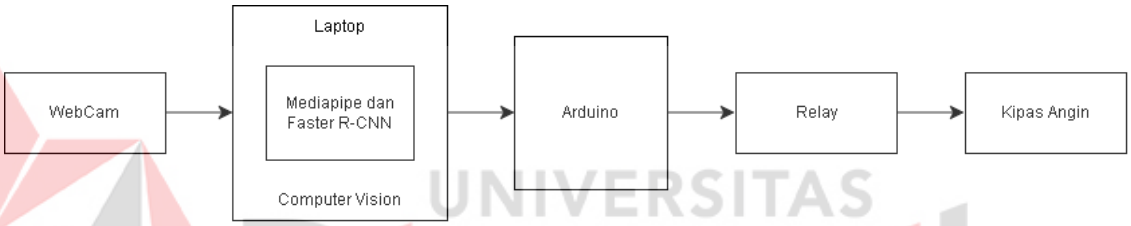
Relay adalah saklar (*switch*) yang dioperasikan secara listrik dan merupakan komponen elektromekanikal yang terdiri dari 2 bagian utama yakni elektromagnet (*coil*) dan mekanikal (seperangkat kontak saklar/switch). Relay menggunakan prinsip elektromagnetik untuk menggerakkan kontak saklar sehingga dengan arus listrik yang kecil (*low power*) dapat menghantarkan listrik yang bertegangan lebih tinggi. Sebagai contoh, dengan Relay yang menggunakan Elektromagnet 5 VDC dan 50 mA mampu menggerakan Armature Relay (yang berfungsi sebagai saklarnya) untuk menghantarkan listrik 220V 2A (Lubis, 2019).

Gambar 2.6 Modul Relay 4 Channel
(Sumber : (Component101, 2021)

BAB III
METODOLOGI PENELITIAN

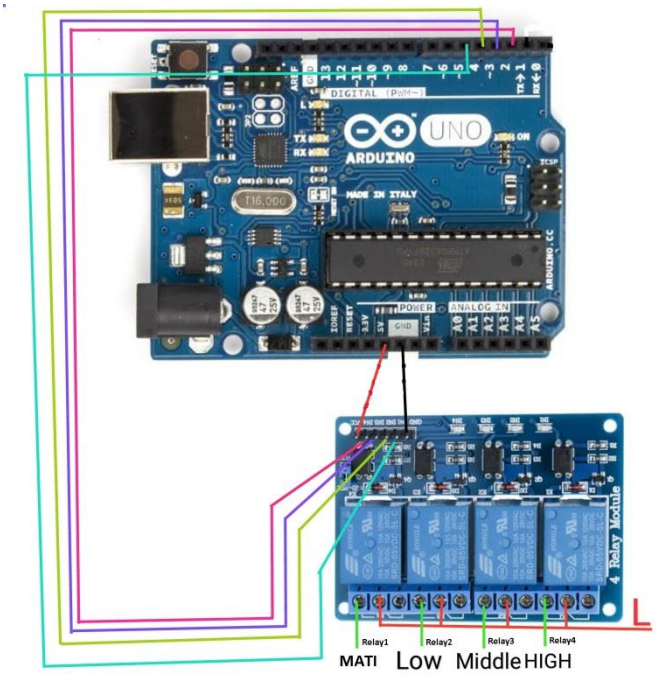
3.1 Perancangan Perangkat Keras

Gambar 3.1 merupakan perancangan *hardware* dengan inputan berasal dari kamera yang mendeteksi gestur jari tangan. Inputan tersebut selanjutnya diproses pada Laptop oleh sistem *Computer Vision* dengan menggunakan Mediapipe dan juga dapat menggunakan Faster-RCNN yang nilai tersebut tampil pada layar Laptop. Setelah nilai inputan sudah tampil, maka nilai tersebut dikirimkan ke Arduino sebagai antarmuka melalui komunikasi serial. Lalu nilai yang ada pada Arduino, melakukan aksi ke relay dengan mematikan atau mengontrol kecepatan kipas angin dari gestur jari tangan.



Gambar 3.1 Model perancangan *Hardware*

Selain itu, gambar 3.2 adalah gambar skematik dari mikrokontroler Arduino yang menghubung ke relay. Lalu relay disambungkan ke level kecepatan kipas angin dimulai dari level nol sebagai mati, level satu sebagai kecepatan satu atau rendah(*low*), level dua sebagai kecepatan dua atau sedang(*middle*), dan level tiga sebagai kecepatan tiga atau tinggi(*high*). NO yaitu *Normally Open*, sedangkan NC yaitu *Normaly Close*. maksudnya adalah saat kondisi diam/mati/tidak diberi tegangan, maka pin COM dan NO tidak terhubung(*open*), sedangkan pin COM dan NC terhubung(*close*), namun saat kondisi relay hidup/diberi tegangan maka NO akan berubah menjadi NC, dan NC akan berubah menjadi NO. Inilah yang sering digunakan sebagai pemutus/penghubung arus listrik (sakelar).



Gambar 3.2 Model perancangan *Hardware* 2

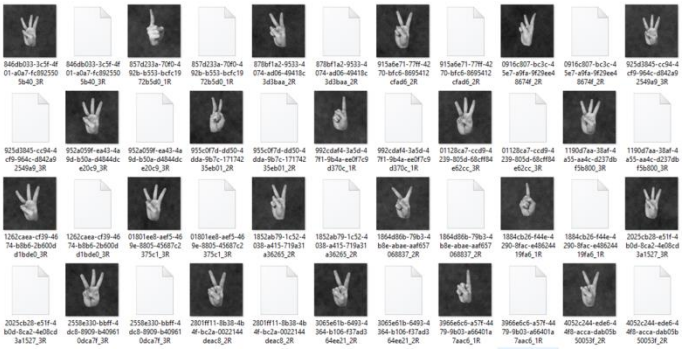
3.2 Instalasi *Environtment*

Dalam Tugas Akhir ini, sebelum memulai proses lebih lanjut, perlu dilakukan instalasi dan pengaturan *environment*. Proses ini dilakukan agar *library* yang dibutuhkan oleh program dapat terpasang dengan baik dan program mampu berjalan tanpa halangan. Instalasi ini dilakukan dengan memasang *Python 3.7.0*, *library* dan *plugins* yang dibutuhkan pada terminal pip dan Anaconda. Library yang digunakan yaitu protobuf, pillow, lxml, Cython, contextlib2, jupyter, matplotlib, pandas, opencv-python, mediapipe, cvzone, pyserial. Mikrokontroler Arduino yang didefinisikan terdapat *String* terminalRead;, digunakan untuk pembacaan data dari terminal serial arduino yang menggunakan bahasa *String*.

3.3 Dataset

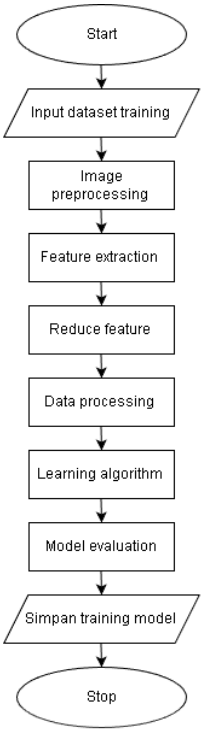
Dataset yang digunakan pada Tugas Akhir ini adalah dataset yang sebelumnya telah dikumpulkan pada penelitian sebelumnya (Nautica, 2022). Pada dataset penelitian sebelumnya terdapat 10 gestur jari, namun yang digunakan peneliti hanya tangan kanan 3 gestur jari tangan dan juga jari kosong. Dataset yang telah dikumpulkan terdapat 4800 citra. yang telah teraugmentasi yang mampu memenuhi kebutuhan penelitian. Proses augmentasi adalah sebuah proses

yang dimana gambar pada dataset diubah bentuk dan ditambah jumlahnya agar dataset menjadi lebih bervariasi dan akurasi model lebih akurat. Dataset ini digunakan oleh penulis agar mampu melakukan pendeteksian 3 jari tangan kanan dan juga jari kosong.



Gambar 3.3 Dataset jari tangan

Dataset gambar yang digunakan untuk *training* sebanyak 3840 gambar dan untuk *validation* sebanyak 960 gambar. Seluruh dataset nantinya dipisahkan dengan jumlah komposisi dataset tertinggi dipakai untuk *training*. Dataset dibagi menjadi 2 bagian dengan komposisi 80% untuk *training*, 20% untuk *validation*. Antar bagian dataset tidak boleh *overlap*, karena merusak proses training model. Komposisi *training* harus jauh lebih besar dari data *validation*. Karena jika komposisi training yang terlalu kecil, model tidak "belajar" (Azis, 2020). *Flowchart* untuk proses training Faster-RCNN dilakukan sebanyak 8000step yang setelah dilakukan training, membuat evaluasi model training. Kemudian disimpan dalam model training dengan format file nama.CKPT., seperti pada gambar 3.4



Gambar 3.4 *Flowchart* proses training *dataset*.

Dimulai dari start kemudian menginput dataset *training*. Lalu *image preprocessing* untuk mengubah intensitas *pixel* gambar agar mudah digunakan dalam proses selanjutnya. *Feature extraction* untuk mengenali suatu objek dengan melihat ciri-ciri khusus yang dimiliki objek tersebut. Setelah itu, *reduce feature* untuk memilih fitur yang berpengaruh dan mengesampingkan fitur yang tidak berpengaruh. Data *processing* untuk memproses dataset yang telah dipilih dari *reduce feature*. Kemudian *learning algorithm* untuk mempelajari algoritma dari proses *training* dataset yang telah selesai. Setelah algoritma dipelajari, model *evaluation* untuk mengevaluasi hasil dari *training* dataset yang telah disimpan dalam model. Kemudian simpan model *training* kedalam file dengan format .CKPT. lalu proses training telah selesai.

3.4 Metode Deteksi dengan Faster-RCNN

Pada metode deteksi dengan Faster-RCNN, program mencari gestur jari tangan yang dimuat untuk dijadikan acuan klasifikasi. Implementasi metode Faster-RCNN, pengujian deteksi menggunakan metode Faster-RCNN diawali dengan membuka dan menyalakan kamera. Pada proses deteksi, jika gestur jari

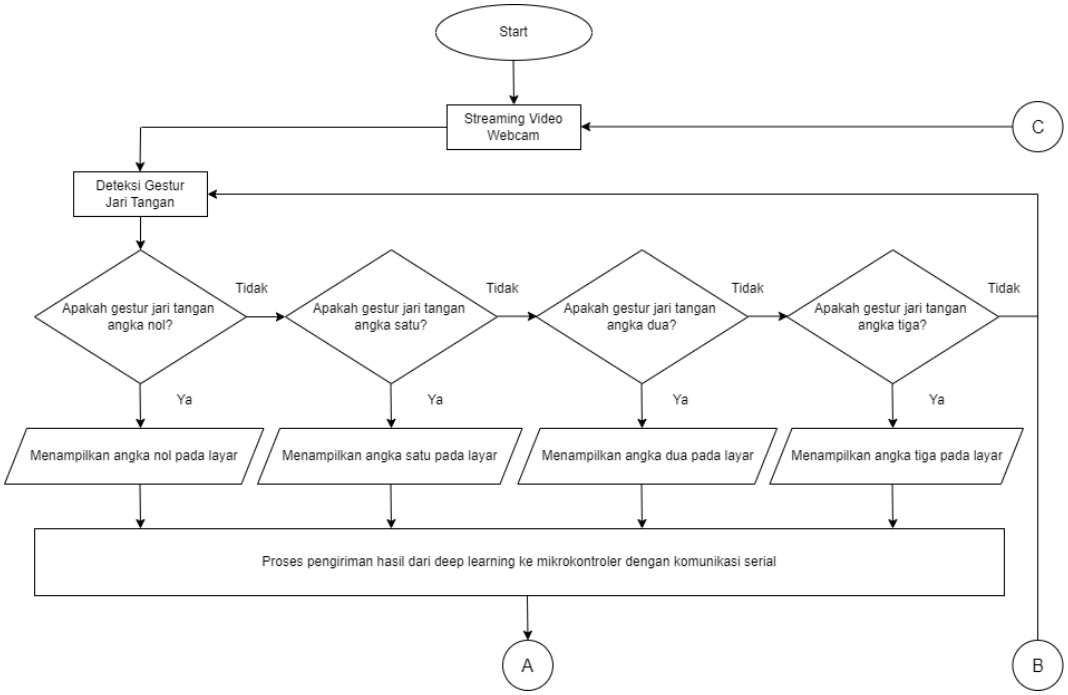
tangan dapat terdeteksi oleh training model Faster-RCNN yang telah dilatih, dan gestur jari tangan telah memasuki *region of interest* dalam program. Maka pada layer utama program muncul gestur angka berapa yang terdeteksi. *Region of interest* dalam program deteksi Faster-RCNN adalah sebuah area khusus untuk melakukan deteksi dari gestur jari tangan, dan kemudian melakukan proses *pre-processing* dengan merubah format gambar yang ditangkap kamera kedalam format gambar yang sesuai dengan dataset. Jika program tidak mendeteksi adanya gestur jari tangan, maka program tetap mencari deteksi gestur jari tangan. Jarak yang diuji pada penelitian ini yaitu dimulai di jarak yang paling dekat 10 cm, 50 cm, 100 cm dan 175 cm. Setelah itu, dicari rata-rata akurasi, error, dan FPS dari setiap gestur jari tangan.

3.5 Metode Deteksi dengan Mediapipe

Pada metode deteksi dengan Mediapipe, program mencari gestur jari tangan yang ada pada kamera. Jika ada gestur tangan yang terdeteksi, maka program memunculkan *landmark* yang digunakan untuk melakukan klasifikasi gestur jari tangan. Implementasi metode Mediapipe, lalu program mendeteksi gestur tangan dan melakukan ekstraksi *landmark* Mediapipe. Jika gestur terdeteksi, maka program menampilkan jumlah jari tangan yang terdeteksi. Jarak yang diuji pada penelitian ini yaitu dimulai di jarak yang paling dekat 10 cm, 50 cm, 100 cm dan 175 cm. Setelah itu, dicari rata-rata akurasi, error, dan FPS dari setiap gestur jari tangan.

3.6 Proses Testing Program Deep Learning

Berdasarkan gambar 3.5 dapat dijelaskan inputan berasal dari kamera yang mendeteksi gestur jari tangan, apakah mendeteksi gestur jari tangan satu, dua, tiga ataupun tidak mendeteksi jari disebut dengan nol. Setelah mendapatkan nilai yang dideteksi, lalu menampilkan hasil ke tampilan layar sebagai tanda bahwa telah mendeteksi gestur jari tangan. Setelah itu program mengirim hasil dari deteksi gestur jari tangan ke mikrokontorler Arduino dengan komunikasi serial.



Gambar 3.5 *Flowchart* proses *Testing* program *Deep Learning*.

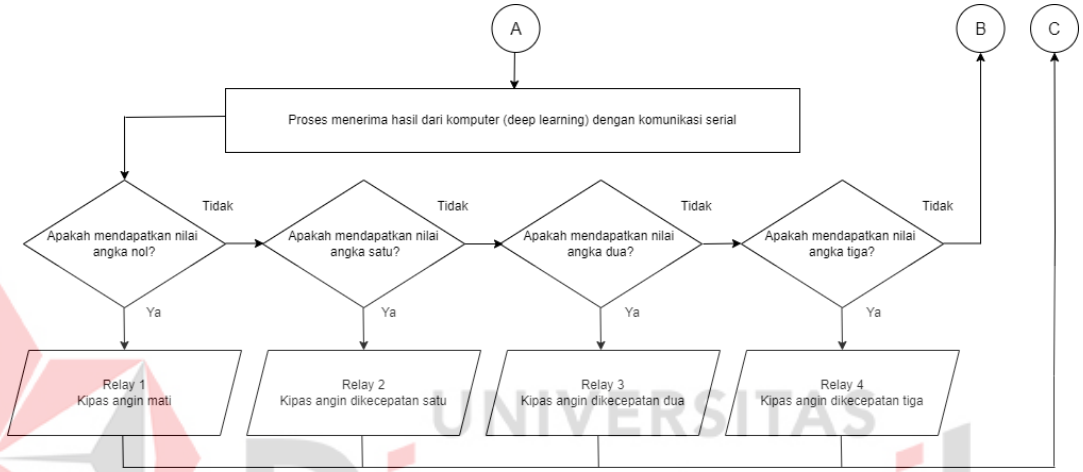
3.7 Format data

Data yang dikirimkan dari program computer vision untuk deep learning ke program mikrokontroler Arduino adalah data string. Format data string dari deep learning membutuhkan *library* pyserial sebagai pengiriman data ke mikrokontroler Arduino. Lalu nantinya diterima oleh mikrokontroler arduino, dengan cara mendefinisikan variable *String* terminalRead = Serial.read*String*Until, maka pembacaan data dari program deep learning yaitu berbahasa string dan kemudian diterima oleh Arduino dengan menggunakan variabel yang berbahasa *String* juga. Kemudian pada arduino data string ini dibuat percabangan. Data *String* ini nantinya yang dapat mendeteksi apakah nilai yang didapat yaitu “0”, “1”, “2”, “3”. Setelah itu, data tersebut digunakan untuk melanjutkan proses percabangan untuk relay.

3.8 *Flowchart* Program Mikrokontroler

Berdasarkan gambar 3.6, menerima hasil dari komputer angka nilai gestur jari tangan pada mikrokontroler Arduino, setelah itu mengecek kondisi apakah nilai yang didapatkan angka satu, dua, tiga, ataupun nol. Nilai ini didapat dari komunikasi serial antara program deep learning dengan mikrokontroler

menggunakan usb serial. Seperti halnya dari pembahasan format data diatas. Jika nilai yang didapat satu, selanjutnya relay2 menjalankan aksi kecepatan putaran kipas angin ditingkat ke satu. Jika nilai yang didapat dua, selanjutnya relay3 menjalankan aksi kecepatan putaran kipas angin ditingkat ke dua. Jika nilai yang didapat tiga, selanjutnya relay4 menjalankan aksi kecepatan putaran kipas angin ditingkat ke tiga. Jika nilai yang didapat nol, selanjutnya relay1 menjalankan aksi kipas angin mati. Setelah itu kembali mengecek kamera untuk melakukan deteksi gestur jari tangan.



Gambar 3.6 *Flowchart* proses testing program pada Arduino.

3.9 Membangun Komunikasi serial antara Computer Vision dan Mikrokontroler Arduino

3.9.1 Tujuan Membangun Komunikasi serial antara *Computer Vision* dan Mikrokontroler Arduino

Tujuan untuk membangun komunikasi serial antara computer vision dan mikrokontroler Arduino.adalah mengetahui cara membangun sebuah komunikasi *Computer Vision* untuk Deep Learning dengan mikrokontroler Arduino secara Serial.

3.9.2 Prosedur Membangun Komunikasi serial antara Computer Vision dan Mikrokontroler Arduino

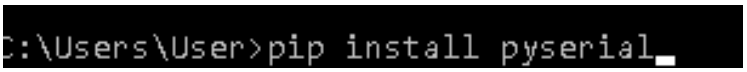
Adapun langkah–langkah dalam melakukannya adalah sebagai berikut:

1. Menginputkan *library* pyserial pada program python.
2. Mendeklarasikan sebuah variabel untuk komunikasi serial.

3. Python mengirim nilai ke arduino menggunakan komunikasi serial.
4. Arduino menerima nilai dari python dengan komunikasi serial.

3.9.3 Hasil dari Membangun Komunikasi serial antara Computer Vision dan Mikrokontroler Arduino

Pada program python, dalam menggunakan fungsi serial, terlebih dahulu untuk menginstall *library* pyserial melalui cmd (*command promp*) dengan mengetik:



Setelah itu, mengimpor *library* pyserial pada program python dengan mengetik:

```
import serial
```

"import serial" *library* yang berfungsi sebagai komunikasi program python dengan program mikrokontroler Arduino. Kemudian mendeklarasikan variabel yang bernama "serialComm" ini yang berfungsi mengetahui letak port usb serial dan *baud rate* yang ada pada arduino.

```
serialComm = serial.Serial('COM3', 9600)
serialComm.write(str(hand_mode).encode())
```

Kemudian kode ini berfungsi untuk mengirim nilai dari variabel hand_mode ke serial arduino, nilai yang dikirimkan ke arduino bertipe data *string*. Variabel hand_mode ini adalah variabel untuk kode tangan kanan yang diberi nilai string "1".

```
serialComm.write(str(counter).encode())
serialComm.write(e.encode())
```

Lalu kode ini berfungsi untuk mengirim nilai dari variabel counter ke serial arduino, nilai yang dikirimkan ke arduino bertipe data *string*. Variabel counter ini adalah variabel jumlah jari tangan satu sampai tiga. Kode dibawahnya berfungsi untuk memberi baris baru agar tidak tercampur pada data serial.

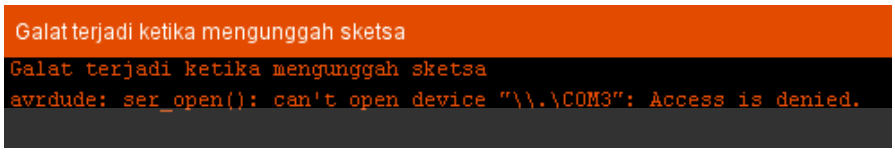
```
String terminalRead;
```

Pada program Arduino, pertama mendeklarasikan variabel yang bernama terminalRead dengan tipe data *string.* Setelah itu, variabel terminalRead diberi perintah untuk menerima nilai atau karakter dari komunikasi serial yang ada,

dengan menjadikannya tipe data *string*. Jadi variabel terminalRead ini untuk menyimpan nilai atau karakter yang didapatkan dari adanya komunikasi serial yang telah terjadi.

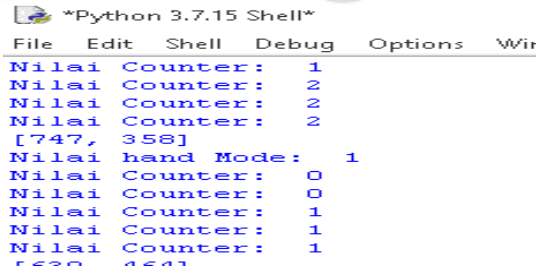
```
terminalRead = Serial.readStringUntil('\n');

if(terminalRead == "13"){
```

Lalu proses terakhir yaitu dengan percabangan dari variabel terminalRead apakah nilai atau karakter yang ada pada variabel terminalRead ini sama dengan “13”, “1” menunjukkan mode tangan satu atau tangan kanan, “3” menunjukkan jari yang dideteksi adalah bentuk gestur jari tiga. Jika iya, maka masuk kedalam percabangan.

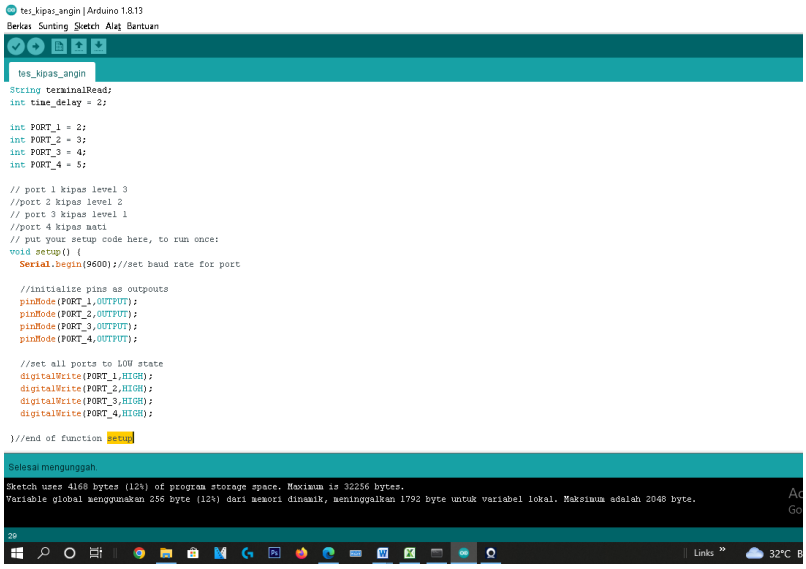


Gambar 4.7 Gagal dalam berkomunikasi serial.

Pada gambar 4.7 kesalahan tersebut terjadi karena belum bisa menyambung komunikasi secara serial antara mikrokontroler Arduino dengan *Computer Vision* untuk Deep Learning (program python).



Gambar 4.8 Python dapat mengeluarkan *output* nilai



Gambar 4.9 Arduino dapat mengunggah program.

Pada gambar 4.8 dan 4.9, jika Arduino IDE dapat mengunggah program, maka yang terjadi pada Python shell mengeluarkan nilai print out yang ada pada program Python. Hal tersebut dikatakan sudah saling terhubung dengan komunikasi secara serial melalui USB serial Arduino.

BAB IV
HASIL DAN PEMBAHASAN

Pada Bab 4 ini membahas pengujian dan analisis, dimulai dari *software* dan *hardware*. Pada tahap pengujian *software* ini memastikan bahwa proses training dataset berjalan dengan baik. Lalu pada tahap pengujian *hardware* memastikan kemampuan komponen alat yang digunakan berjalan sesuai dengan yang telah dibuat.

4.1 Pengujian Komparasi Metode Mediapipe dan Faster-RCNN

Sebelum membuat sistem deteksi gestur jari tangan dengan metode Mediapipe dan Faster-RCNN, diperlukan dataset untuk detraining, kemudian dijadikan model Faster-RCNN dan juga *library* Mediapipe untuk mendeteksi gestur jari tangan dengan *landmark* Mediapipe.

4.1.1 Tujuan Komparasi Metode Mediapipe dan Faster-RCNN

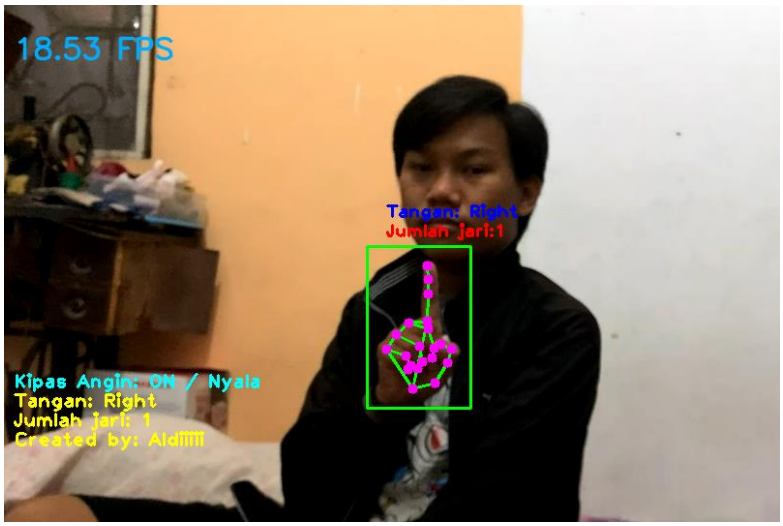
Pengujian ini dilakukan untuk mengetahui besar performa pada sistem deteksi jari tangan menggunakan Mediapipe dan mengetahui besar akurasi klasifikasi pada sistem deteksi jari tangan menggunakan Faster-RCNN dengan jarak dimulai dari 10, 50, 100 dan 175 cm.

4.1.2 Prosedur Pengujian Komparasi Metode Mediapipe dan Faster-RCNN

Adapun langkah–langkah dalam melakukan pengujian komparasi ini adalah sebagai berikut:

1. Memberikan input deteksi gestur jari tangan yang sama disetiap metode yang digunakan.
2. Menjalankan proses deteksi gestur jari tangan pada setiap metode.
3. Mengambil *frame per second* dan akurasi data pada setiap metode.

4.1.3 Hasil Pengujian Komparasi Metode Mediapipe dan Faster-RCNN



Gambar 4.1 Hasil deteksi metode Mediapipe

Pada gambar 4.1, hasil deteksi metode Mediapipe mampu mendeteksi gestur jari 1 pada tangan kanan. Deteksi gestur jari tangan ini menggunakan kamera handphone dengan jarak 100cm yang disambungkan ke komputer dan pencahayaan yang merata. Hal ini disebabkan dari ekstraksi dari *landmark* milik mediapipe yang terlihat pada tangan subjek uji. Dalam program yang diproses pada penelitian ini, pengaturan parameter dari *landmark* Mediapipe berjalan ketika salah satu *keypoints* pada jari membuka, *keypoints* tersebut memberikan nilai 1 kepada program. Pada gambar 4.1, menunjukkan *keypoints* yang membuka, sehingga memberi nilai 1 pada program, karena 4 jari yang lainnya menutup setelah dijumlahkan dengan *keypoints* jari lainnya.

A. Hasil Pengujian Metode Mediapipe dengan Jarak 50cm

Dalam tabel 4.1 pengujian nilai akurasi dan *error* yang diambil dengan mencoba program deteksi gestur tangan sebanyak 10 kali kepada setiap subjek. Setiap subjek yang diuji melakukan 4 gestur jari tangan yang berbeda dimulai dari gestur angka 0,1,2 dan 3. Jadi program mendeteksi 1 kali gestur tangan jika benar, maka bernilai 10% akurasi dan 90% *error*, dengan melakukan 10 kali percobaan deteksi gestur kepada subjek. Jika benar seluruhnya, maka nilainya adalah 100% akurasi dan 0% *error*.

Pada pengujian tabel 4.1, dapat diketahui bahwa akurasi, *error* dan FPS dari seluruh deteksi gestur tangan. Pengujian metode mediapipe dengan jarak 50 cm, dengan tangan yang diuji yaitu bagian tangan kanan. Subjek pada penelitian ini yaitu sebanyak 5 subjek dengan setiap subjek 3 kali percobaan, diambil untuk setiap gestur jari tangan.

Tabel 4.1 Hasil pengujian metode Mediapipe jarak 50cm

No.	Subjek	Jarak (cm)	Gestur	Hasil Deteksi		Akurasi (%)	*Error* (%)	FPS
				Benar	Salah			
1	Subjek 1	50	0	8	2	80	20	24
		50	1	10	0	100	00	29
		50	2	9	1	90	10	23
		50	3	9	1	90	10	31
		50	0	1	9	10	90	30
		50	1	10	0	100	00	18
		50	2	9	1	90	10	22
		50	3	9	1	90	10	29
		50	0	8	2	80	20	18
		50	1	10	0	100	00	29
		50	2	10	0	100	00	17
		50	3	9	1	90	10	23
2	Subjek 2	50	0	9	1	90	10	24
		50	1	9	1	90	10	22
		50	2	10	0	100	00	21
		50	3	10	0	100	00	26
		50	0	9	1	90	10	24
		50	1	9	1	90	10	18
		50	2	9	1	90	10	29
		50	3	9	1	90	10	19
		50	0	1	9	10	90	23
		50	1	9	1	90	10	18
		50	2	10	0	100	00	23
		50	3	9	1	90	10	21
3	Subjek 3	50	0	8	2	80	20	25
		50	1	9	1	90	10	25
		50	2	8	2	80	20	25
		50	3	9	1	90	10	22
		50	0	9	1	90	10	24
		50	1	10	0	100	00	28
		50	2	9	1	90	10	31
		50	3	9	1	90	10	22
		50	0	9	1	90	10	23
		50	1	8	2	80	20	25
		50	2	9	1	90	10	19
		50	3	9	1	90	10	14
4	Subjek 4	50	0	8	2	80	20	27
		50	1	10	0	100	00	28
		50	2	9	1	90	10	32
		50	3	8	2	80	20	23
		50	0	9	1	90	10	33
		50	1	9	1	90	10	23
		50	2	9	1	90	10	29
		50	3	9	1	90	10	28
		50	0	9	1	90	10	27

No.	Subjek	Jarak (cm)	Gestur	Hasil Deteksi		Akurasi (%)	*Error* (%)	FPS
				Benar	Salah			
		50	1	9	1	90	10	32
		50	2	8	2	80	20	32
		50	3	8	2	80	20	29
5	Subjek 5	50	0	8	2	80	20	23
		50	1	9	1	90	10	24
		50	2	9	1	90	10	22
		50	3	8	2	80	20	21
		50	0	1	9	10	90	29
		50	1	9	1	90	10	30
		50	2	8	2	80	20	24
		50	3	9	1	90	10	26
		50	0	8	2	80	20	31
		50	1	9	1	90	10	24
		50	2	9	1	90	10	19
		50	3	9	1	90	10	23
Rata – Rata Gestur Jari 0						**70**	**30**	**25**
Rata – Rata Gestur Jari 1						**92.7**	**7.3**	**25**
Rata – Rata Gestur Jari 2						**90**	**10**	**24**
Rata – Rata Gestur Jari 3						**88.7**	**11.3**	**24**

Pada tabel 4.1, dilakukan pengujian menentukan nilai rata-rata akurasi, *error* dan FPS dari setiap gestur tangan dengan metode Mediapipe. Hasil nilai rata-rata dari setiap gestur jari tangan memiliki selisih nilai yang sedikit. Tetapi pada gestur 0 akurasinya sebesar 70% dan *error*-nya 30%. Hal ini diasumsikan karena mediapipe mendeteksi tangan dengan ekstraksi dari *landmark* Mediapipe yang mempunyai *keypoints* untuk pembacaan jari tangan.

Keypoints ini pada saat membaca gestur jari 0, salah pembacaan menjadi gestur 1, 2, 3 yang menyebabkan nilai rata-rata akurasi dan error dari gestur jari 0 memiliki selisih nilai yang timpang. Untuk hasil nilai rata-rata dari FPS, diambil dari keluaran FPS pada saat program sedang berjalan, kemudian diambil nilai FPS setiap percobaan pada subjek. Jadi FPS ini dihasilkan dari spesifikasi *hardware* komputer yang digunakan, semakin baik spesifikasi *hardware* yang digunakan, maka semakin baik FPS yang dihasilkan.

B. Hasil Pengujian Metode Mediapipe

Pada pengujian tabel 4.2, pengujian ini hasil dari sistem deteksi gestur tangan menggunakan metode Mediapipe pada setiap jarak yang terdiri dari jarak 10 cm, 50 cm, 100 cm, 175 cm. Pengujian ini diambil nilai rata-rata dari akurasi, error dan FPS pada setiap gestur jari tangan yaitu gestur jari 0, 1, 2 dan 3.

Pengujian jarak 10 cm terdapat pada lampiran 1, lalu pengujian jarak 100 cm terdapat pada lampiran 2 dan pengujian jarak 175 cm terdapat pada lampiran 3.

Tabel 4.2 Hasil pengujian metode Mediapipe setiap jarak

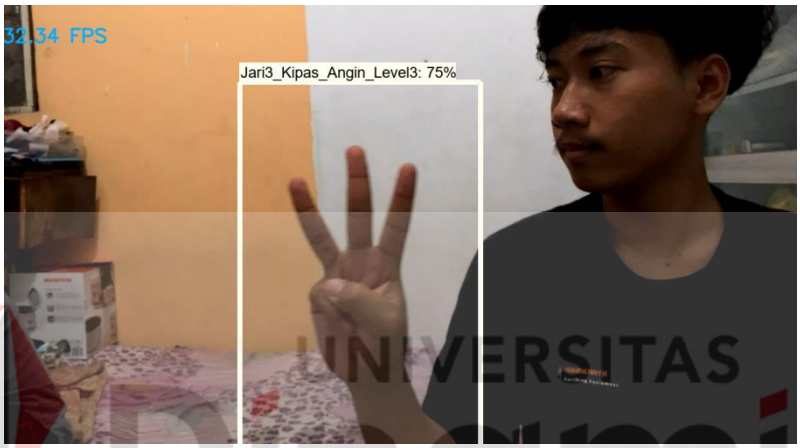
Jarak (cm)	Gestur	Akurasi (%)	*Error* (%)	FPS
10	0	37	63	24
	1	46.7	53.3	23
	2	42.7	57.3	23
	3	40	60	22
50	0	70	30	25
	1	92.7	7.3	25
	2	90	10	24
	3	88.7	11.3	24
100	0	54.7	45.3	24
	1	80	20	22
	2	78	22	24
	3	74	26	23
175	0	63.3	36.7	25
	1	80.7	19.3	25
	2	67.3	32.7	24
	3	66	34	23

Pada tabel 4.2, pengujian metode mediapipe pada tiap jarak, menentukan nilai rata-rata akurasi, *error* dan FPS dari setiap gestur tangan dan dari jarak yang berbeda dengan metode Mediapipe. Jika terdapat nilai *Error* yang tinggi, maka deteksi gestur jari 0 salah mendeteksi. Pada hasil nilai rata-rata di jarak 10 cm, hasil nilai rata-rata akurasi yang didapatkan kecil dibandingkan dengan uji jarak yang lain. Hal ini terjadi dari deteksi gestur tangan di jarak yang paling dekat dengan kamera, karena tidak dapat mendeteksi gestur jari dengan maksimal.

Sedangkan pada jarak 100 cm dan 175 cm hasil nilai rata-rata akurasinya terdapat nilai yang seharusnya pada jarak 100 cm nilai yang didapatkan lebih baik dibandingkan pada jarak 175 cm. Hal ini diasumsikan saat mendeteksi gestur jari pada jarak 100 cm, FPS yang menurun serta sistem deteksi mengalami drop dari *hardware* yang digunakan menyebabkan kesalahan deteksi gestur jari tangan. Mediapipe proses komputasinya yang ringan, tetapi spesifikasi *hardware* juga berpengaruh untuk mendapatkan kestabilan pada sistem. Pada jarak 175 cm saat mendeteksi gestur jari sistem deteksi dan FPS berjalan dengan stabil yang membuat deteksi gestur jari dapat terdeteksi.

C. Hasil Pengujian Metode Faster-RCNN

Pada gambar 4.2, hasil deteksi metode Faster-RCNN mampu mendeteksi gestur jari tangan 3 pada tangan kanan dan menampilkan nilai 75% yang merupakan hasil dari proses komputasi sistem Faster-RCNN. Deteksi gestur jari tangan ini menggunakan kamera handphone dengan jarak 50 cm yang disambungkan ke komputer dan pencahayaan yang merata. Faster-RCNN, mencari input gambar kemungkinan lokasi dari obyek. Posisi dari obyek yang ada pada gambar memiliki kemungkinan obyek dibatasi dari wilayah yang diketahui sebagai *region of interest* (ROI).



Gambar 4.2 Hasil deteksi metode Faster-RCNN

Pada pengujian tabel 4.3, pengujian ini hasil dari sistem deteksi gestur tangan menggunakan metode Faster-RCNN pada setiap jarak yang terdiri dari jarak 10 cm, 50 cm, 100 cm, 175 cm. Pengujian ini diambil nilai rata-rata dari akurasi, error dan FPS pada setiap gestur jari tangan yaitu gestur jari 0, 1, 2 dan 3. Pengujian jarak 10 cm terdapat pada lampiran 4, pada jarak 50 cm terdapat pada lampiran 5, lalu pengujian jarak 100 cm terdapat pada lampiran 6 dan pengujian jarak 175 cm terdapat pada lampiran 7.

Tabel 4.3 Hasil pengujian metode Faster-RCNN setiap jarak

Jarak (cm)	Gestur	Akurasi (%)	*Error* (%)	FPS
10	0	34	66	18
	1	60	40	24
	2	14	86	17
	3	16	84	17
50	0	24	76	21
	1	73	27	20
	2	14	86	20

Jarak (cm)	Gestur	Akurasi (%)	*Error* (%)	FPS
	3	12	88	20
100	0	8.7	91.3	18
	1	58	42	21
	2	2.7	97.3	18
	3	5.3	94.7	17
175	0	4.7	95.3	17
	1	51.3	48.7	22
	2	2	98	17
	3	1.3	98.7	17

Pada tabel 4.3, pengujian metode Faster-RCNN disetiap jarak, menentukan nilai rata-rata akurasi, *error* dan FPS dari setiap gestur tangan dan jarak yang berbeda. Jika terdapat nilai *Error* yang tinggi, maka deteksi gestur jari 0 salah mendeteksi. Pada hasil nilai rata-rata di jarak 10 cm dan 50 cm, hasil nilai rata-rata akurasi yang didapatkan selisihnya sedikit dibandingkan dengan uji jarak 100 cm dan 175 cm. Hal ini terjadi dari deteksi gestur tangan di jarak yang paling dekat dengan kamera yaitu 10 cm masih dapat membaca gestur jari dan juga 50 cm jarak lebih dari 10 cm.

Tetapi untuk jarak 100 cm dan 175 cm, sistem deteksi tidak mampu mendeteksi gestur jari pada jarak tersebut. Hal ini diasumsikan adanya keterbatasan dari sistem deteksi Faster-RCNN sehingga dalam mendeteksi pada jarak 100 cm dan 175 cm hanya beberapa jari saja yang dapat terdeteksi. Selain itu, dapat terjadi dengan FPS yang menurun dikarenakan spesifikasi *hardware* yang digunakan. Karena Faster-RCNN membutuhkan proses komputasi yang lebih berat, sehingga menyebabkan pada jarak yang jauh, gestur jari tidak dapat terdeteksi.

D. Komparasi Metode Mediapipe dan Faster-RCNN

Pada pengujian ini, dengan mengkomparasi metode Mediapipe dan Faster-RCNN dari nilai rata-rata akurasi, *error* dan FPS. Pengujian dilakukan dengan menggunakan gestur jari 0 sebagai perbandingan. Hasil nilai rata-rata akurasi gestur jari 0 dilakukan perbandingan antara metode Mediapipe dengan metode Faster-RCNN.

Tabel 4.4 Komparasi Mediapipe dengan Faster-RCNN pada gestur jari 0

Jarak (cm)	Mediapipe			Faster-RCNN		
	Akurasi (%)	Error (%)	FPS	Akurasi (%)	Error (%)	FPS

Jarak (cm)	Mediapipe			Faster-RCNN		
	Akurasi (%)	Error (%)	FPS	Akurasi (%)	Error (%)	FPS
10	37	63	24	34	66	18
50	70	30	25	24	76	21
100	54.7	45.3	24	8.7	91.3	18
175	63.3	36.7	25	4.7	95.3	17

Dari perbandingan nilai rata-rata akurasi, *error* dan FPS antara metode Mediapipe dan Faster-RCNN disetiap jarak pada gestur jari tangan 0. Nilai rata-rata akurasi yang dihasilkan metode Mediapipe, hasilnya tinggi jika dilihat pada jarak 10 cm, 50 cm, 100 cm dan 175 cm dibandingkan dengan metode Faster-RCNN. Pada hasil nilai rata-rata akurasi ini membuktikan proses komputasi yang dijalankan metode Mediapipe lebih ringan daripada proses komputasi metode Faster-RCNN. Jika terdapat nilai *Error* yang tinggi, maka deteksi gestur jari 0 salah mendeteksi. FPS yang dihasilkan dari Mediapipe lebih tinggi dari Faster-RCNN. Jadi Mediapipe masih lebih mumpuni karena proses komputasi yang ringan.

4.2 Mengontrol Kecepatan Kipas Angin secara Computer Vision melalui Deteksi Gestur Jari Tangan

4.2.1 Tujuan Mengontrol Kecepatan Kipas Angin secara *Computer Vision* melalui Deteksi Gestur Jari Tangan

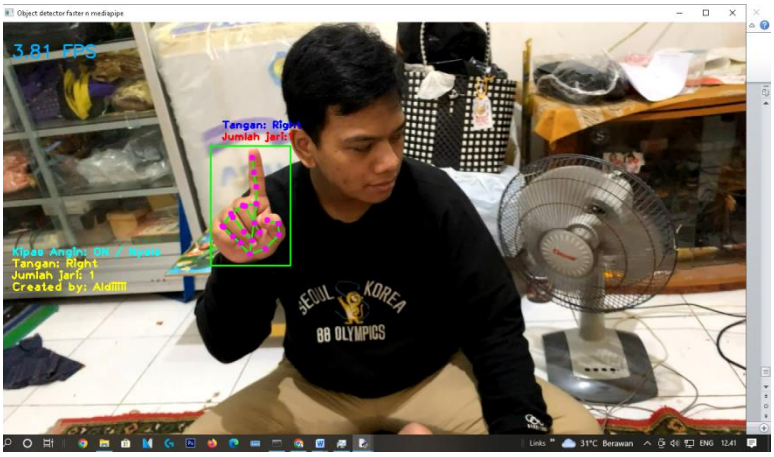
Tujuannya adalah dapat mengontrol kecepatan kipas angin secara computer vision melalui deteksi gestur jari tangan.

4.2.2 Prosedur Mengontrol Kecepatan Kipas Angin secara Computer Vision melalui Deteksi Gestur Jari Tangan

Adapun langkah–langkah dalam melakukannya adalah sebagai berikut:

1. Memberikan input deteksi gestur jari tangan pada computer vision.
2. Menjalankan proses deteksi gestur jari tangan dan menjalankan proses mengontrol kecepatan kipas angin.
3. Mengambil hasil keluaran dari deteksi gestur jari tangan dan kontrol kecepatan kipas angin.

4.2.3 Hasil dari Membangun Komunikasi serial antara Computer Vision dan Mikrokontroler Arduino



Gambar 4.10 Hasil deteksi gestur jari untuk mengontrol kipas angin.

Pada gambar 4.10 yaitu hasil deteksi gestur jari tangan untuk mengontrol kipas angin program sudah berjalan sesuai harapan peneliti. Subjek uji membentuk gestur jari tangan satu, kipas angin tersebut menyala karena relay yang menjalankan aksi kecepatan putaran kipas angin ditingkat ke satu dari data yang diterima Arduino.

Tabel 4.5 Uji kontrol kecepatan putaran kipas angin dengan Mediapipe

No.	Subjek	Jarak (cm)	Gestur	Output	Kecepatan Nyala Kipas				Hasil		Keterangan
					0	1	2	3	Tepat	Salah	
1	Subjek 1	100	0	0	✓				✓		
		100	1	1		✓			✓		
		100	2	2					✓		
							✓				
		100	3	3				✓	✓		
		100	0	0	✓				✓		
		100	1	1		✓			✓		
		100	2	2					✓		
							✓				
		100	3	3				✓	✓		
		100	0	1		✓				✓	Output salah; sistem deteksi berpindah ke jari 1.
		100	1	1		✓			✓		
		100	2	2					✓		
							✓				
		100	3	2						✓	Output salah; sistem deteksi berpindah ke jari 2.
							✓				
2	Subjek 2	100	0	0	✓				✓		
		100	1	1		✓			✓		
		100	2	2					✓		
							✓				

No.	Subjek	Jarak (cm)	Gestur	Output	Kecepatan Nyala Kipas				Hasil		Keterangan
					0	1	2	3	Tepat	Salah	
		100	3	1		✓				✓	Output salah; sistem deteksi berpindah ke jari 1.
		100	0	0	✓				✓		
		100	1	1		✓			✓		
		100	2	2					✓		
							✓				
		100	3	3				✓	✓		
		100	0	0	✓				✓		
		100	1	1		✓			✓		
		100	2	2					✓		
							✓				
		100	3	3				✓	✓		
3	Subjek 3	100	0	0	✓				✓		
		100	1	1		✓			✓		
		100	2	2					✓		
							✓				
		100	3	3				✓	✓		
		100	0	0	✓				✓		
		100	1	1		✓			✓		
		100	2	2					✓		
							✓				
		100	3	3				✓	✓		
		100	0	1		✓				✓	Output salah; sistem deteksi berpindah ke jari 1.
		100	1	1		✓			✓		
		100	2	2					✓		
							✓				
		100	3	3				✓	✓		
4	Subjek 4	100	0	0	✓				✓		
		100	1	1		✓			✓		
		100	2	2					✓		
							✓				
		100	3	3				✓	✓		
		100	0	1		✓				✓	Output salah; sistem deteksi berpindah ke jari 1.
		100	1	1		✓			✓		
		100	2	2					✓		
							✓				
		100	3	3				✓	✓		
		100	0	0	✓				✓		
		100	1	1		✓			✓		
		100	2	2					✓		
							✓				
		100	3	3				✓	✓		
5	Subjek 5	100	0	0	✓				✓		
		100	1	1		✓			✓		
		100	2	2					✓		
							✓				
		100	3	3				✓	✓		
		100	0	0	✓				✓		
		100	1	1		✓			✓		
		100	2	2					✓		
							✓				
		100	3	3				✓	✓		
		100	0	0	✓				✓		
		100	1	1		✓			✓		
		100	2	1		✓				✓	Output salah; sistem deteksi berpindah ke jari 1.

No.	Subjek	Jarak (cm)	Gestur	Output	Kecepatan Nyala Kipas				Hasil		Keterangan
					0	1	2	3	Tepat	Salah	
		100	3	2						✓	Output salah; sistem deteksi berpindah ke jari 2.
							✓				
Jumlah Hasil Tepat dan Salah pada Uji Kontrol Kecepatan Kipas Angin									53	7	

Pada tabel 4.3 setelah dilakukan pengujian, hasil deteksi gestur jari tangan untuk mengontrol kecepatan putaran kipas angin dengan metode Mediapipe diambil dengan cara program deteksi 4 gestur jari tangan yang berbeda dimulai dari angka 0,1,2, dan 3 dengan jarak 100cm. Hasil *output* pada tabel harus sesuai dengan gestur jari tangan yang sudah ditentukan. Jika salah, maka berdampak salah juga ke sistem kontrol kecepatan putaran kipas angin. Karena sistem kontrol kecepatan putaran kipas angin saling berkaitan dengan sistem deteksi gestur jari tangan dari adanya komunikasi secara serial antara mikrokontroler Arduino dengan *Computer Vision* untuk Deep Learning.

Pengujian ini diuji dengan jarak 100cm, dengan kamera handphone dan pencahayaan yang merata. Setelah itu total dari tabel 4.3 menunjukkan hasil yang sudah tepat mendapatkan nilai sebesar 53, Sedangkan hasil yang salah mendapatkan nilai sebesar 7. Hasil yang didapat pada kategori salah ini disebabkan karena pada saat sistem mendeteksi gestur jari tangan 0, output yang keluar menjadi 1 yang seharusnya output yang benar adalah 0. Hal ini disebabkan karena sistem mendeteksinya salah menangkap jumlah tangan yang sedang dideteksinya.

Sistem deteksi ini masih dapat berjalan melalui metode Mediapipe, sebagai pengontrol kecepatan kipas angin. Dalam metode Faster-RCNN belum dapat melakukan kontrol kecepatan kipas angin. Sistem deteksi gestur juga masih belum mencapai tingkat sempurna, karena spesifikasi *hardware* yang digunakan masih rendah yang menyebabkan FPS turun maupun program *not responding*, sehingga kontrol kecepatan kipas angin menjadi salah juga karena deteksi gestur jari tangannya salah. Jadi program deteksi gestur jari tangan masih dapat dikembangkan untuk mencari nilai hasil yang tinggi dan memiliki tingkat keakuratan yang baik dengan metode lain.

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan pengujian pada sistem deteksi gestur jari tangan untuk mengontrol kecepatan putaran kipas angin yang dirancang pada Tugas Akhir ini, maka ada beberapa kesimpulan:

1. Dari hasil pengujian yang telah dilakukan, hasil nilai rata-rata akurasi setiap gestur jari tangan dengan metode Mediapipe pada jarak 10 cm gestur 0 akurasinya sebesar 37%, gestur 1 akurasinya sebesar 46.7%, gestur 2 akurasinya sebesar 42.7%, gestur 3 akurasinya sebesar 40%. Pada jarak 50 cm gestur 0 akurasinya sebesar 70%, gestur 1 akurasinya sebesar 92.7%, gestur 2 akurasinya sebesar 90%, gestur 3 akurasinya sebesar 88.7%.
2. Hasil nilai rata-rata akurasi dan *error* seluruh deteksi gestur jari tangan dengan metode Faster-RCNN pada jarak 10 cm gestur 0 akurasinya sebesar 34%, gestur 1 akurasinya sebesar 60%, gestur 2 akurasinya sebesar 14%, gestur 3 akurasinya sebesar 16%. Pada jarak 50 cm gestur 0 akurasinya sebesar 24%, gestur 1 akurasinya sebesar 73%, gestur 2 akurasinya sebesar 14%, gestur 3 akurasinya sebesar 12%.
3. Pada pengujian ini, dengan mengkomparasi metode Mediapipe dan Faster-RCNN. Hasil nilai rata-rata akurasi ini membuktikan proses komputasi yang dijalankan metode Mediapipe ringan daripada proses komputasi metode Faster-RCNN. FPS yang dihasilkan dari Mediapipe juga tinggi dari Faster-RCNN.
4. Dalam pengujian sistem deteksi gestur jari tangan untuk mengontrol kecepatan putaran kipas angin, dilakukan yang dapat mengontrol kipas angin adalah metode Mediapipe. Pengujian dengan jarak 100 cm. Hasil yang “tepat” mendapatkan nilai sebesar 88.33% (53 ketepatan), dapat dikatakan bahwa sistem deteksi gestur jari tangan sudah berjalan dengan baik, tetapi hasil yang “salah” mendapatkan nilai sebesar 11.67% (7 kesalahan), hal ini disebabkan kesalahan pada saat pembacaan deteksi gestur jari tangan yang masih belum mencapai tingkat sempurna.

5. Dalam membangun sebuah komunikasi secara serial antara *Computer Vision* dan mikrokontroler Arduino. Pengujian ini telah berhasil terhubung antara Computer Vision dengan mikrokontroler Arduino, sehingga pengujian ini telah dapat mengontrol kecepatan putaran kipas angin melalui sistem deteksi gestur jari tangan.

5.2 Saran

Adapun saran untuk pengembangan pada penelitian ini agar lebih baik terdapat beberapa saran berikut:

1. Mengimplementasikan sistem deteksi gestur jari tangan yang dapat dikembangkan dengan objek yang lain.
2. Pemilihan arsitektur metode deteksi yang lebih mumpuni, disarankan untuk menggunakan arsitektur *Single-Shot Detection* atau *YOLO.*
3. Untuk penggunaan metode Faster-RCNN, disarankan untuk penggunaan yang dapat mendeteksi objek yang banyak sekaligus, seperti objek mendeteksi kendaraan di lampu merah, Faster-RCNN dapat mendeteksi objek dengan jumlah banyak sekaligus.
4. Spesifikasi dalam penggunaan *hardware* yang lebih baik agar dalam proses komputasi mampu berjalan lebih lancar, disarankan untuk spesifikasi RAM minimal sebesar 16 GB dengan kartu grafis yang memiliki VRAM minimal sebesar 6GB. Karena dalam proses komputasi yang bekerja adalah RAM dan juga VRAM pada kartu grafis.

DAFTAR PUSTAKA

Alamsyah, D. (2022). Deteksi Ujung Jari menggunakan Faster-RCNN dengan

Arsitektur Inception v2 pada Citra Derau. *JuSiTik*, 1-5.

Arisandi, E. D. (2014). Kemudahan Pemograman Mikrokontroller Arduino Pada Aplikasi Wahana Terang. *SETRUM*, 46-49.

Arsal, M. (2020). Face Recognition Untuk Akses Pegawai Bank Menggunakan Deep Learning Dengan Metode CNN. *TEKNOSI*, 55-63.

Asmaleni, P. (2020). Pengembangan Sistem Kontrol Kipas Angin dan Lampu Otomatis Berbasis Saklar Suara Menggunakan Arduino Uno. *Jurnal Kumparan Fisika*, 59-66.

Azis, H. (2020). Performa Klasifikasi K-NN dan Cross-validation pada Data. *ILKOM Jurnal Ilmiah*, 81-86.

Component101. (2021). *5v Four-Channel Relay Module*. Retrieved from component101: https://components101.com/switches/5v-four-channel-relay-module-pinout-features-applications-working-datasheet

Everitt. (2018, August 10). *Yolo Vs Faster-RCNN*. Retrieved from Everitt's Blog: https://everitt257.github.io/post/2018/08/10/object_detection.html

IDE, A. (n.d.). *Arduino Uno/Genuino*. Retrieved from https://www.arduino.cc/en/software

Kho, D. (n.d.). *Pengertian Relay dan Fungsinya*. Retrieved September 6, 2022, from https://teknikelektronika.com/pengertian-relay-fungsi-relay/

Khumaidi, A. (2019). *Mikrokontroler Arduino*. Retrieved from lecturer ppns: https://lecturer.ppns.ac.id/aguskhumaidi/2019/09/05/mikrokontroler-arduino/

Lubis, Z. (2019). Kontrol Mesin Air Otomatis Berbasis Arduino Dengan Smartphone. *Buletin Utama Teknik*, 155-159.

Mediapipe. (2019). *Live Machine Learning Anywhere*. Retrieved from Mediapipe Developer: https://mediapipe.dev/

Nautica, M. R. (2022). Hand Gesture Detection sebagai Alat Bantu Ajar Berhitung menggunakan Mediapipe dan Convolutional Neural Network secara Realtime. *Repository Universitas Dinamika*, 1-44.

Perdananto, A. (2019). Penerapan Deep Learning Pada Aplikasi Prediksi Penyakit Pneumonia Berbasis Convolutional Neural Network. *JICT*, 1-10.

Prastyo, E. A. (2020). *Sensor Suhu DS18B20*. Retrieved September 6, 2022, from https://www.edukasielektronika.com/2020/09/sensor-suhu-ds18b20.html

Python.org. (n.d.). *Python Software Foundation*. Retrieved from Python Logo: https://www.python.org/community/logos/

Ramdhon, A. N. (2021). Penerapan Face Recognition Pada Sistem Presensi. *JACOST*, 12-17.

Rosebrock, A. (2021, April 12). *pyimagesearch*. Retrieved from OpenCV Haar Cascades: https://pyimagesearch.com/2021/04/12/opencv-haar-cascades/

Shaoqing Ren, K. H. (2016). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arxiv*, 1-14.

Tanner, G. (2019). *Detectron2 - Object Detection with PyTorch*. Retrieved from Gilbert Tanner: https://gilberttanner.com/blog/detectron-2-object-detection-with-pytorch/

Zein, A. (2018). Pendeteksian Kantuk Secara Real Time Menggunakan Pustaka OPENCV dan DLIB PYTHON. *Sainstech*, 22-26.