TUGAS AKHIR – KI141502

RANCANG BANGUN COMPUTATION OFFLOADING DENGAN METODE FUZZY MULTI CRITERIA DECISION MAKING UNTUK PENINGKATAN KINERJA PROSES KOMPUTASI PADA MIKROKONTROLER ARDUINO (STUDI KASUS PENGHITUNGAN JUMLAH ORANG PADA GAMBAR)

WIDHI MAHAPUTRA PANDE PUTU
NRP 05111440000058

Dosen Pembimbing
Waskitho Wibisono, S.Kom., M.Eng., Ph.D.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

TUGAS AKHIR – KI141502

RANCANG BANGUN COMPUTATION OFFLOADING DENGAN METODE FUZZY MULTI CRITERIA DECISION MAKING UNTUK PENINGKATAN KINERJA PROSES KOMPUTASI PADA MIKROKONTROLER ARDUINO (STUDI KASUS PENGHITUNGAN JUMLAH ORANG PADA GAMBAR)

WIDHI MAHAPUTRA PANDE PUTU
NRP 05111440000058

Dosen Pembimbing
Waskitho Wibisono, S.Kom., M.Eng., Ph.D.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

(Halaman ini sengaja dikosongkan)

UNDERGRADUATE THESES – KI1502

DESIGN AND DEVELOPMENT OF COMPUTATION OFFLOADING FOR IMPROVING COMPUTATION PERFORMANCE IN ARDUINO MICROCONTROLLER (CASE STUDY : COUNTING PEOPLE IN IMAGE CASE)

WIDHI MAHAPUTRA PANDE PUTU
NRP 05111440000058

Advisor
Waskitho Wibisono, S.Kom., M.Eng., Ph.D.

DEPARTMENT OF INFORMATICS
Faculty of Information and Communication Technology
Sepuluh Nopember Institute of Technology
Surabaya 2018

(Halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN

RANCANG BANGUN *COMPUTATION OFFLOADING* DENGAN METODE *FUZZY MULTI CRITERIA DECISION MAKING* UNTUK PENINGKATAN KINERJA PROSES KOMPUTASI PADA MIKROKONTROLER ARDUINO (STUDI KASUS PENGHITUNGAN JUMLAH ORANG PADA GAMBAR)

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Komputasi Berbasis Jaringan
Program Studi S-1 Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:
WIDHI MAHAPUTRA PANDE PUTU
NRP: 05111440000058

Disetujui oleh Pembimbing Tugas Akhir:

1. Waskitho Wibisono, S.Kom., M.Eng., Ph.D. .....
   (NIP. 197410222000031001) (Pembimbing 1)

SURABAYA
JUNI, 2018

(Halaman ini sengaja dikosongkan)

RANCANG BANGUN *COMPUTATION OFFLOADING* DENGAN METODE *FUZZY MULTI CRITERIA DECISION MAKING* UNTUK PENINGKATAN KINERJA PROSES KOMPUTASI PADA MIKROKONTROLER ARDUINO (STUDI KASUS PENGHITUNGAN JUMLAH ORANG PADA GAMBAR)

Nama Mahasiswa	**: WIDHI MAHAPUTRA PANDE PUTU**
NRP	**: 05111440000058**
Jurusan	**: Departemen Informatika FTIK-ITS**
Dosen Pembimbing 1	**: Waskitho Wibisono, S.Kom., M.Eng., Ph.D.**
Dosen Pembimbing 2	**: -**

Abstrak

Offloading adalah suatu metode pengeksekusian sebuah beban kerja pada sebuah perangkat dengan mengirimkan modul berisi beban kerja tersebut kepada perangkat lain yang memiliki sumber daya dan kemampuan komputasi yang lebih baik. Hasil eksekusi dari beban kerja akan diterima kembali oleh perangkat yang telah mengirim modul beban kerja sebelumnya. Teknik ini dianggap sebagai salah satu cara mengatasi keterbatasan perangkat bergerak yang memiliki sumber daya dan kemampuan komputasi yang terbatas. Oleh karena itu, diperlukan adanya penerapan metode offloading dalam mengeksekusi beban kerja pada perangkat bergerak dengan tujuan dapat melakukan peningkatan kinerja pada perangkat.

Penentuan beban kerja diproses secara lokal atau offloading ditentukan oleh sebuah decision maker. Pada tugas akhir ini decision maker menggunakan metode Fuzzy Multi Criteria Decision Making.

Pada tugas akhir ini, pemanfaatan computation offloading diharapkan dapat meningkatkan kinerja proses komputasi

sehingga meminimalisir waktu eksekusi beban kerja. Dari uji coba, dilakukan, metode computation offloading dengan Fuzzy Multi Criteria Decision Making mendapatkan waktu eksekusi beban kerja yang lebih cepat dibandingkan dengan metode lokal dan offloading saja.

Kata kunci: Offloading, image processing, Perangkat IoT, beban kerja.

DESIGN AND DEVELOPMENT OF COMPUTATION OFFLOADING FOR IMPROVING COMPUTATION PERFORMANCE IN ARDUINO MICROCONTROLLER (CASE STUDY : COUNTING PEOPLE IN IMAGE CASE)

Student's Name	**: WIDHI MAHAPUTRA PANDE PUTU**
Student's ID	**: 05111440000058**
Department	**: Informatics FTIK-ITS**
First Advisor	**: Waskitho Wibisono, S.Kom., M.Eng., Ph.D.**
Second Advisor	**: -**

Abstract

Offloading is a method of executing a workload on a device by sending a module containing workload to other device that have better resources and computing capabilities. This technique is considered as one of the ways to overcoming the limitations of devices that have limited resources and computing capabilities. Therefore, this offloading method to executing the workload on a device with purpose improving performance on devices.

The decision of task to be done in local or offloading mode is calculate by decision maker. In this undergraduate thesis, the decision maker using Fuzzy Multi Criteria Decision Making method.

In this undergraduate thesis, the hope to improve computation process to reduce execution time. From the experiment, the computation offloading method with Fuzzy Multi Criteria Decision Making got execution time of task less than the only local method and offloading method.

Keywords : Offloading, image processing, workload.

(***Halaman ini sengaja dikosongkan***)

KATA PENGANTAR

Astungkara, segala puji dan syukur bagi Ida Sang Hyang Widhi Wasa, yang telah melimpahkan nikmat, dan rejeki-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul :

"**RANCANG BANGUN *COMPUTATION OFFLOADING* DENGAN METODE *FUZZY MULTI CRITERIA DECISION MAKING* UNTUK PENINGKATAN KINERJA PROSES KOMPUTASI PADA MIKROKONTROLER ARDUINO (STUDI KASUS PENGHITUNGAN JUMLAH ORANG PADA GAMBAR)**"

yang merupakan salah satu syarat dalam menempuh ujian sidang guna memperoleh gelar Sarjana Komputer. Selesainya Tugas Akhir ini tidak terlepas dari bantuan dan dukungan beberapa pihak, sehingga pada kesempatan ini penulis mengucapkan terima kasih kepada:

1. Ida Sang Hyang Widhi Wasa, karena atas izin-Nya lah penulis dapat menyelesaikan Tugas Akhir dengan baik.
2. Kedua orang tua, Pande Made Raka Sumandi serta Ni Made Putri Widyawati, dan ketiga saudara Pande Made Indra Putra, Pande Komang Gita Nandini, serta Pande Ketut Gunawan Saputra, terima kasih atas doa dan bantuan moral dan material selama penulis belajar di departemen Informatika ITS.
3. Bapak Waskitho Wibisono, S.Kom., M.Eng., Ph.D. selaku pembimbing I yang telah membimbing dan memberikan motivasi, nasehat dan bimbingan dalam menyelesaikan Tugas Akhir ini sekaligus dosen wali penulis yang telah memberikan arahan, masukan dan motivasi kepada penulis.
4. Bapak Darlis Herumurti, S.Kom., M.Kom. selaku kepala departemen Informatika ITS.

5. Bapak Dr. Eng. Radityo Anggoro, S.Kom., M.Sc. selaku koordinator Tugas Akhir di departemen Informatika ITS.
6. Seluruh dosen dan karyawan Teknik Informatika ITS yang telah memberikan ilmu dan pengalaman kepada penulis selama menjalani masa studi di ITS.
7. Ibu Eva Mursidah dan Ibu Sri Budiati yang selalu mempermudah penulis dalam peminjaman buku di RBTC.
8. Seluruh rekan – rekan TC 2014 yang sudah mendukung penulis selama perkuliahan.
9. Teman-teman seperjuangan RMK NCC/KBJ, yang telah menemani dan menyemangati penulis.
10. Rekan kerja, Ida Ayu Gede Danika Esa Pradnyani yang sudah memberi dukungan dan motivasi kepada penulis.
11. Rekan – rekan kontrakan E-103 Nobby, Dharmawan, Fintanto, Fahmi, dan Hanendyo yang sudah memberi dukungan dan motivasi kepada penulis.

Penulis menyadari bahwa Tugas Akhir ini masih memiliki banyak kekurangan sehingga dengan kerendahan hati penulis mengharapkan kritik dan saran dari pembaca untuk perbaikan ke depan.

Surabaya, Mei 2018

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

(***Halaman ini sengaja dikosongkan***)

DAFTAR KODE SUMBER

(Halaman ini sengaja dikosongkan)

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi komunikasi saat ini telah berkembang dengan pesat guna memenuhi kebutuhan hidup manusia. Perkembangan teknologi ini memunculkan berbagai konsep baru seperti *Internet of Things* (IoT). IoT merupakan konsep dimana perangkat yang terhubung dengan internet dapat bertukar data dan mempengaruhi perangkat atau objek fisik sekitarnya [1]. Contoh perangkat IoT adalah Arduino, dan Raspberry Pi. Arduino adalah perangkat pengendali mikro single-board yang bersifat open-source [2]. Arduino tidak hanya sekedar sebuah alat pengembangan, Arduino adalah kombinasi dari hardware, bahasa pemrograman dan *Integrated Development Environment* (IDE) yang canggih [3]. Arduino memiliki prosesor Atmel AVR dan softwarenya memiliki bahasa pemrograman sendiri [2]. Sedangkan Raspberry Pi merupakan *Single Board Computer* (SBC) yang dapat menjalankan sistem operasi Raspbian dan ukurannya yang kecil dengan harga yang murah [4].

Tujuan awal dibuat Arduino dan Raspberry Pi adalah untuk membuat perangkat mudah dan murah, dari perangkat yang ada saat itu. Dan perangkat tersebut ditujukan untuk para siswa yang akan membuat perangkat desain dan interaksi. Berbagai macam project telah dibuat dengan menggunakan Arduino dan Raspberry Pi dengan tujuan mempermudah suatu kegiatan atau menyelesaikan permasalahan masyarakat dunia saat ini. Namun demikian, Arduino dan Raspberry Pi memiliki masalah tersendiri yang tidak bisa dihindari seperti sumber energi baterai yang memiliki daya tahan yang terbatas, memori yang terbatas, dan kemampuan komputasi yang terbatas. Kebanyakan *project* yang dioperasikan pada Arduino dan Raspberry Pi membutuhkan sumber daya yang besar seperti pengolahan citra gambar secara

high-definition, *face recognition*, dan pengolahan data dari sensor yang dipasang pada Arduino dan Raspberry Pi.

Dalam perkembangannya, teknologi komputasi *cloud* secara *offloading* telah direncanakan sebagai teknologi yang menjanjikan untuk mengatasi keterbatasan pada perangkat. Mekanisme komputasi *cloud* secara *offloading* dianggap sebagai salah satu teknik yang paling baik dan sangat diperlukan yang berpotensi dalam menghemat energi untuk pengguna sebuah perangkat. Namun, upaya penelitian saat ini untuk mekanisme *offloading* masih terbatas dan memiliki banyak kekurangan. Mekanisme *offloading* tidak dapat dilakukan jika perangkat tidak memiliki koneksi internet dan juga tidak dapat diterapkan pada beberapa proses komputasi yang harus dilakukan secara lokal. Keputusan dalam menentukan sebuah proses komputasi aplikasi dilakukan secara metode *offloading* atau secara lokal dipengaruhi oleh beberapa faktor – faktor yang menjadi tantangan yang akan diteliti pada tugas akhir ini.

Hasil yang diharapkan adalah aplikasi mikrokontroler Arduino (client) dapat menentukan sebuah komputasi dilakukan secara *offloading* pada server atau lokal pada kondisi tertentu secara optimal agar mendapatkan execution time paling minimal dan penghematan sumber daya baterai yang maksimal pada perangkat di sisi client.

1.2 Rumusan Masalah

Tugas akhir ini mengangkat beberapa rumusan masalah sebagai berikut:

1. Bagaimana menerapkan *computation offloading* dengan metode *Fuzzy Multi Criteria Decision Making* pada perangkat IoT yang dapat meningkatkan performa proses komputasi?
2. Bagaimana menentukan penerapan *computation offloading* dengan metode *Fuzzy Multi Criteria Decision Making* pada perangkat IoT dengan menggunakan faktor – faktor pendukung sebagai acuan?

3. Bagaimana hasil waktu eksekusi beban kerja dalam penerapan *computation offloading* dengan metode *Fuzzy Multi Criteria Decision Making* pada perangkat IoT?

1.3 Batasan Permasalahan

Permasalahan yang dibahas pada tugas akhir ini memiliki batasan sebagai berikut:

1. Perangkat IoT yang digunakan adalah mikrokontroler Arduino Yun dan Raspberry Pi 2.
2. Data gambar yang digunakan untuk *image processing* adalah dataset dari Penn-Fudan Database [5].
3. Metode yang digunakan untuk meningkatkan kinerja proses komputasi adalah *computation offloading.*
4. Metode penentuan beban kerja diproses secara lokal atau secara *offloading* adalah metode *Fuzzy Multi Criteria Decision Making.*
5. Library yang digunakan untuk pengolahan citra digital adalah OpenCV.

1.4 Tujuan

Tujuan dari tugas akhir ini adalah sebagai berikut:

1. Membangun sebuah *computation offloading* dengan metode *Fuzzy Multi Criteria Decision Making* pada perangkat IoT yang dapat menentukan secara dinamis dan optimal suatu beban kerja diproses secara *offloading* ke *server* atau secara lokal berdasarkan faktor – faktor pendukung *computation offloading*.
2. Melakukan implementasi komputasi *offloading* pada perangkat IoT untuk peningkatan kinerja proses komputasi suatu beban kerja.

1.5 Manfaat

Dengan dibuatnya tugas akhir ini diharapkan dapat memberikan manfaat untuk menghasilkan sebuah *computation offloading* yang dapat menentukan secara dinamis pemrosesan suatu beban kerja yang optimal dengan tujuan untuk memaksimalkan performa komputasi agar memiliki waktu eksekusi seminimal mungkin.

Sedangkan bagi penulis, tugas akhir ini bermanfaat sebagai sarana untuk mengimplementasikan ilmu dan algoritma *image processing* pada studi kasus penghitungan jumlah orang pada gambar agar berguna bagi masyarakat.

1.6 Metodologi

Pembuatan tugas akhir ini dilakukan dengan menggunakan metodologi sebagai berikut :

1.6.1 Penyusunan Proposal

Tahap awal tugas akhir ini adalah menyusun proposal tugas akhir. Pada proposal, diajukan gagasan untuk menerapkan komputasi *offloading* sebagai solusi dari keterbatasan mikrokontroler Arduino seperti keterbatasan *processor* yang memproses komputasi pada setiap beban kerja perangkat lunak.

1.6.2 Studi Literatur

Pada tahap ini dilakukan untuk mencari informasi dan studi literatur apa saja yang dapat dijadikan referensi untuk membantu pengerjaan Tugas Akhir ini. Informasi didapatkan dari buku dan literatur yang berhubungan dengan *computation offloading* yang akan dibangun dan metode untuk *image processing*. Informasi yang dicari adalah implementasi *computation offloading* pada perangkat, faktor – faktor yang digunakan untuk menentukan

dilakukannya *computation offloading*, dan metode-metode *image processing.*

1.6.3 Analisis dan Desain Perangkat Lunak

Pada tahap ini akan dilakukan analisa, perancangan, dan pendefinisian kebutuhan sistem untuk mengetahui permasalahan yang akan dihadapi pada tahap implementasi. Kemudian akan dijabarkan kebutuhan – kebutuhan tersebut ke dalam perancangan fitur sistem. Berikut langkah yang akan dilakukan dalam perancangan proses perangkat lunak :

1. Instalasi *library* pada mikrokontroler Arduino dan Raspberry Pi.
2. Instalasi *library* pada *cloud server*.
3. Perancangan komunikasi antar perangkat IoT (mikrokontroler Arduino dan Raspberry Pi).
4. Perancangan komunikasi antara perangkat IoT dan *cloud server*.
5. Perancangan metode *Fuzzy Multi Criteria Decision Making* pada mikrokontroler Arduino.
6. Perancangan metode penghitungan jumlah orang pada Raspberry Pi dan *cloud server*.

1.6.4 Implementasi Perangkat Lunak

Implementasi merupakan tahap untuk membangun metode-metode yang sudah diajukan pada proposal Tugas Akhir. Implementasi dilakukan dengan menggunakan suatu perangkat lunak yaitu Arduino IDE dan Visual Studio Code sebagai *Text Editor.*

1.6.5 Pengujian dan Evaluasi

Pada tahap ini perangkat lunak yang telah dibangun diuji coba dengan menggunakan data uji coba yang ada. Data uji coba tersebut diuji coba pada perangkat lunak mikrokontroler Arduino

sebagai *client* dan *cloud server* dengan tujuan mengetahui kemampuan perangkat lunak dalam menentukan penggunaan metode *offloading* atau tidaknya (lokal) pada suatu beban kerja berdasarkan faktor – faktor pendukung yang terjadi saat itu pada perangkat *client* dan mengevaluasi hasil tugas akhir dengan jurnal pendukung yang ada. Hasil evaluasi mencakup waktu eksekusi dari kemampuan perangkat lunak tersebut dalam menentukan *computation offloading* suatu beban kerja pada mikrokontroler Arduino.

1.6.6 Penyusunan Buku

Pada tahap ini disusun buku sebagai dokumentasi dari pelaksanaan tugas akhir yang mencangkup seluruh konsep, teori, implementasi, serta hasil yang telah dikerjakan.

1.7 Sistematika Penulisan Laporan

Sistematika penulisan laporan tugas akhir adalah sebagai berikut:

1. Bab I. Pendahuluan
   Bab ini berisikan penjelasan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, dan sistematika penulisan dari pembuatan tugas akhir.
2. Bab II. Tinjauan Pustaka
   Bab ini berisi kajian teori dari metode dan algoritma yang digunakan dalam penyusunan Tugas Akhir ini. Secara garis besar, bab ini berisi tentang sistem operasi OpenWrt, sistem operasi Raspbian, metode *offloading*, faktor – faktor penentu dilakukannya metode *offloading*, socket sebagai metode pengiriman data, serta MobileNets dan OpenCV digunakan dalam penerapan *image processing* pada citra.
3. Bab III. Perancangan Sistem
   Bab ini berisi pembahasan mengenai perancangan dari *computation offloading* yang diterapkan pada mikrokontroler

Arduino untuk menentukan dilakukannya metode *offloading* pada suatu beban kerja saat melakukan *image processing* berdasarkan faktor – faktor tertentu.

4. Bab IV. Implementasi
   Bab ini menjelaskan implementasi yang berbentuk *Pseudocode* yang berupa *Pseudocode* dari *offloading computation* dari sistem perangkat lunak *image recognition* beserta penerapan komunikasi antar server dan client.
5. Bab V. Hasil Uji Coba dan Evaluasi
   Bab ini berisikan hasil uji coba dari *computation offloading* yang digunakan untuk menentukan dilakukannya metode *offloading* pada suatu beban kerja dari sistem perangkat lunak *image processing* yang sudah diimplementasikan pada *Pseudocode*. Uji coba dilakukan dengan menggunakan datatest dan dataset citra yang memiliki kualitas rendah. Hasil evaluasi mencakup kedinamisan dari kemampuan *computation offloading* dalam menentukan dilakukannya metode *offloading* pada beban kerja berdasarkan faktor – faktor penentu yang di alami perangkat client dan server saat itu.
6. Bab VI. Kesimpulan dan Saran
   Bab ini merupakan bab yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan, masalah-masalah yang dialami pada proses pengerjaan Tugas Akhir, dan saran untuk pengembangan solusi ke depannya.
7. Daftar Pustaka
   Bab ini berisi daftar pustaka yang dijadikan literatur dalam tugas akhir.
8. Lampiran
   Dalam lampiran terdapat tabel-tabel data hasil uji coba dan *Pseudocode* program secara keseluruhan.

(Halaman ini sengaja dikosongkan)

BAB II
TINJAUAN PUSTAKA

Bab ini berisi pembahasan mengenai teori-teori dasar yang digunakan dalam tugas akhir. Teori-teori tersebut diantaranya adalah Arduino, sistem operasi OpenWrt, metode *offloading*, faktor – faktor penentu dilakukannya metode *offloading*, socket dan beberapa teori lain yang mendukung pembuatan tugas akhir.

2.1 Arduino

Arduino adalah perangkat pengendali mikro *single-board* yang bersifat *open-source* [2]. Arduino memiliki berbagai macam jenis diantaranya Arduino Uno, Arduino Due, Arduino Mega, Arduino Leonardo, Arduino Mini, Arduino Micro, Arduino Yun dan lainnya [2]. Arduino Yun adalah board mikrokontroler dengan *microchip* ATmega32u4 dan Atheros AR9331 [6]. Arduino Yun mendukung sistem operasi Linux yang bernama OpenWrt [6]. Arduino Yun memiliki beberapa fitur diantaranya Ethernet dan WiFi, USB-A port, micro-SD card slot, 20 digital *input/output pins*, 16 MHz *crystal oscillator*, *micro* USB *connection*, ICSP *header*, dan 3 tombol reset [6].

Gambar 2.1 Arduino Yun [6]

2.2 Sistem Operasi OpenWrt

OpenWrt adalah project sistem operasi *open-source* yang digunakan untuk perangkat *embedded* [7]. Komponen utama OpenWrt adalah Linux, util-linux, uClibc, dan BusyBox [7]. Semua komponen sudah dioptimasi dengan ukuran yang kecil agar bisa dioperasikan pada *storage* terbatas dan memori yang terbatas. OpenWrt dapat diatur dengan menggunakan *command-line interface* (ssh) [7].

2.3 Raspberry Pi

Raspberry Pi adalah komputer *single-board circuit* yang dapat digunakan untuk menjalankan program perkantoran, permainan, dan pemutar media beresolusi tinggi [8]. Raspberry Pi memiliki dua model yaitu model A dan model B [8]. Perbedaan model A dan model B terletak pada penyimpanan yang digunakan, model A sebesar 256 MB dan model B sebesar 512 MB [8].

Gambar 2.2 Raspberry Pi 2 Model B [9]

2.4 Sistem Operasi Raspbian

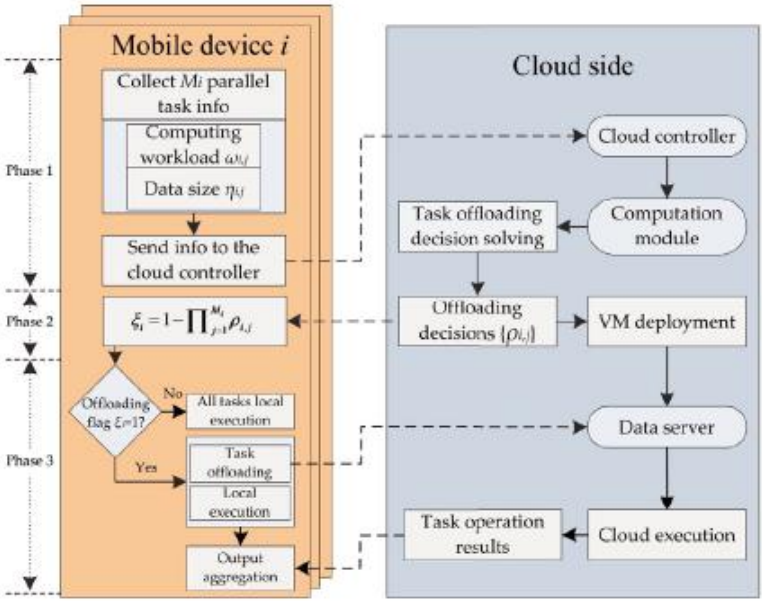
Sistem operasi Raspbian adalah sistem operasi berbasis Debian yang dikhususkan untuk perangkat Raspberry Pi [10]. Raspbiian memiliki beberapa seri diantaranya Raspbian Stretch dan Raspbian Jessie [10]

2.5 *Internet of Things (IoT)*

IoT merupakan konsep dimana perangkat yang terhubung dengan internet dapat bertukar data dan mempengaruhi perangkat atau objek fisik sekitarnya [1]. IoT adalah internet dari 3 hal diantaranya manusia dengan manusia, manusia dengan mesin, dan mesin dengan mesin [1]. Contoh perangkat IoT adalah mikrokontroler Arduino dan Raspberry Pi.

2.6 *Offloading*

Metode *Offloading* telah menjadi teknik yang menjanjikan untuk memecahkan masalah yang dihadapi perangkat *smartphone*, yaitu dengan memungkinkan *smartphone* melakukan metode *offload* atau mengirimkan suatu beban kerja komputasi yang intensif ke server [11]. Berbagai upaya telah dilakukan untuk menerapkan metode *offload* ke *server* untuk memperoleh keuntungan dari kapabilitas *crossplatform bytecode* [11]. Contoh alur eksekusi beban kerja pada penerapan metode *offloading* yang diimplementasikan pada perangkat bergerak dapat dilihat pada Gambar 2.3.



Gambar 2.3 Alur eksekusi beban kerja pada perangkat bergerak ke perangkat komputasi awan [11]

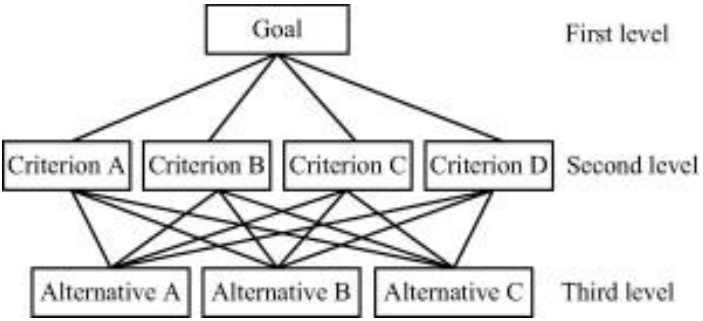
2.7 Faktor Penentu Metode *Offloading*

Keputusan dalam menentukan dilakukannya metode *Offloading* adalah tantangan. Hal ini bergantung pada *volume* data dan transmisi *bandwidth*, konteks yang di eksekusi dari beban kerja komputasi, perbandingan dalam kemampuan eksekusi antara SoC (*system on chip*) *smartphone* dan *processor server* serta seberapa efek keuntungan jika melakukan metode *Offloading* [11]. Membuat data rinci sebuah beban kerja komputasi dan pemantauan jaringan *quality-of-service* (QoS) sangat penting dalam *framework*. Selain itu, karena komputasi dan karakteristik komunikasi adalah dinamis, sebuah *daemon* yang secara periodik mengukur

kapabilitas komputasi, performa komunikasi, konsumsi daya dan secara dinamis membuat keputusan dilakukannya metode *offloading* untuk setiap *task* yang memenuhi syarat untuk dilakukannya *offloading*, hal ini dibutuhkan dalam *framework* [11].

2.8 *Fuzzy Multi Criteria Decision Making*

Multi Criteria Decision Making (MCDM) adalah suatu metode pengambilan keputusan untuk menetapkan alternatif terbaik dari sejumlah alternatif berdasarkan beberapa kriteria tertentu. Salah satu jenis MCDM yaitu *Fuzzy Multi Criteria Decision Making*. Metode ini dikembangkan untuk membantu pengambil keputusan dalam melakukan pengambilan keputusan terhadap beberapa alternatif keputusan untuk mendapatkan suatu keputusan yang akurat dan optimal [12]. *Fuzzy Multi Criteria Decision Making* adalah salah satu metode yang bisa membantu pengambil keputusan dalam melakukan pengambilan keputusan terhadap beberapa alternatif keputusan yang harus diambil dengan beberapa kriteria yang akan menjadi bahan pertimbangan [12]. Secara umum MCDM digambarkan pada Gambar 2.4



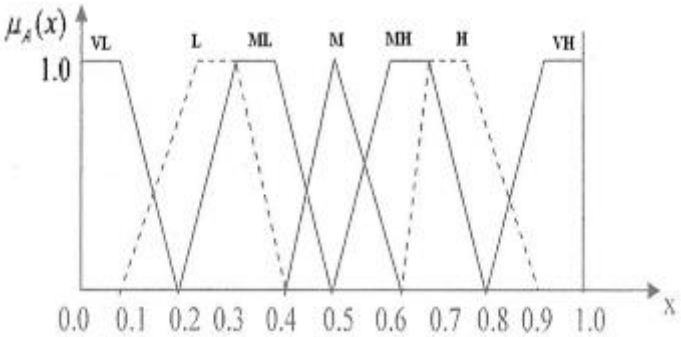
Gambar 2.4 *Multi Criteria Decision Making* [12]

Fuzzy Multi Criteria Decision Making diawali dengan inisialisasi *fuzzy number* yang ditunjukkan pada Gambar 2.5.

Linguistic variable	Fuzzy number
Very high (VH)	(0.8,0.9,1,1)
High	(0.6,0.7,0.8,0.9)
Moderate high	(0.5,0.6,0.7,0.8)
Moderate	(0.4,0.5,0.5,0.6)
Moderate low	(0.2,0.3,0.4,0.5)
Low	(0.1,0.2,0.3,0.4)
Very low	(0.0,0.0,0.1,0.2)

Gambar 2.5 *Fuzzy Number* dan variabel linguistik [12]

Fuzzy Number direpresentasikan pada grafik yang ditunjukkan pada Gambar 2.6.



Gambar 2.6 Grafik representasi *fuzzy number* [12]

Langkah 2: Setiap kriteria akan diberikan variabel linguistik dan *fuzzy number* dimasukkan ke masing-masing kriteria. Lalu dihitung rata-ratanya dengan persamaan pada Gambar 2.7.

$$A_{ij} = \frac{1}{p} \oplus \left(a^j{}_{i1} \oplus a^j{}_{i2} + \ldots + a^j{}_{ik}\right); \quad K = 1, 2, \ldots\ldots\ldots\ldots, P. \text{ eq}\ldots\ldots 1$$

Gambar 2.7 Persamaan rata-rata *fuzzy number* [12]

Langkah 3: Menghitung nilai defuzzyfikasi dari setiap kriteria dengan rumus yang ditunjukkan pada Gambar 2.8.

$$e = \frac{(a+b+c+d)}{4}$$

Gambar 2.8 Persamaan nilai defuzzyfikasi [12]

Langkah 4: Menghitung normalisasi *weight* dari masing-masing kriteria dengan membagi nilai defuzzyfikasi kriteria dengan total jumlah kriteria yang ada.

Langkah 5: Mengkonversi nilai-nilai faktor penentu menjadi variabel linguistik untuk masing-masing alternatif.

Langkah 6: Menghitung skor akhir dari masing-masing alternatif dengan mengalikan nilai defuzzyfikasi dengan nilai *weight* dari masing-masing kriteria yang ditunjukkan pada Gambar 2.9.

$$TS = [Xij]\ [Wj]$$

Gambar 2.9 Persamaan nilai skor akhir *Fuzzy Multi Criteria Decision Making* [12]

Langkah 7: Membandingkan nilai skor akhir dari masing-masing alternatif. Alternatif yang memiliki nilai skor akhir yang paling besar adalah alternatif yang dipilih.

2.9 OpenCV

*Open*CV (*Open Source Computer Vision*) adalah *library* yang utamanya digunakan untuk pemrosesan visi komputer. *Open*CV adalah *library* gratis yang dapat digunakan di berbagai *platform*, seperti GNU/Linux maupun Windows. *Open*CV mulanya ditulis dalam bahasa pemrograman C++, namun saat ini *Open*CV dapat digunakan pada berbagai bahasa seperti Python, Java, atau MATLAB [13].

2.10 MobileNets

MobileNets adalah *class* model yang efisien digunakan untuk perangkat *mobile* dan perangkat *embedded*. MobileNets digunakan untuk *object detection*. MobileNets menggunakan *width multiplier* dan *resolution multiplier* untuk mengurangi *size* dan latency [14].

BAB III
PERANCANGAN SISTEM

Bab ini membahas mengenai perancangan dan pembuatan sistem. Perangkat lunak pada perangkat IoT yang dibuat pada tugas akhir ini adalah menentukan keputusan dilakukannya *computation offloading* dengan metode *Fuzzy Multi Criteria Decision Making* berdasarkan faktor-faktor penentu yang mendukung metode *offloading*.

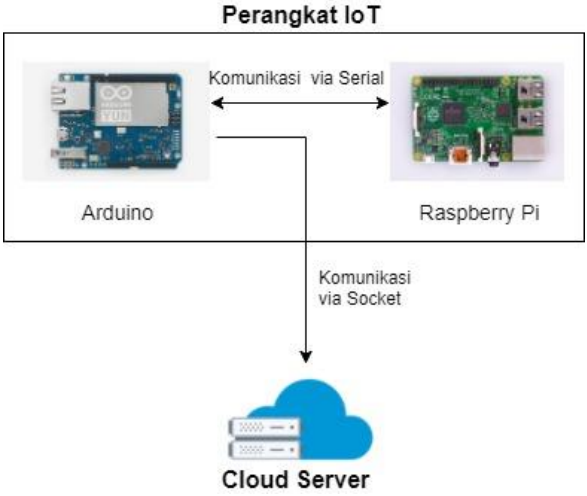
3.1 Deskripsi Umum Sistem

Pada sub bab ini dijelaskan mengenai deskripsi umum sistem yang dibangun. Sistem yang dibangun adalah sistem yang menerapkan *computation offloading* dengan metode *Fuzzy Multi Criteria Decision Making* pada perangkat IoT yang terdiri dari mikrokontroler Arduino dan Raspberry Pi. Studi kasus yang digunakan adalah penghitungan jumlah orang pada gambar dengan menggunakan data citra dari Penn-Fudan Database [5]. Mikrokontroler Arduino menentukan apakah beban kerja diproses secara lokal atau *offloading* dengan metode *Fuzzy Multi Criteria Decision Making*. *Fuzzy Multi Criteria Decision Making* menggunakan nilai RAM Raspberry Pi yang tersedia, *download speed*, *upload speed*, dan jumlah data citra yang diproses untuk menentukan apakah beban kerja diproses secara lokal atau *offloading*. Jika diproses secara lokal maka data citra akan dikirimkan ke Raspberry Pi lalu diproses, jika diproses secara *offloading* maka data citra akan dikirimkan ke *cloud server* lalu diproses.

3.2 Arsitektur Umum Sistem

Pada sub bab ini dijelaskan mengenai arsitektur umum sistem yang dibangun, sistem terdiri dari perangkat IoT (Arduino dan Raspberry Pi) dan *cloud server*. Komunikasi antar perangkat

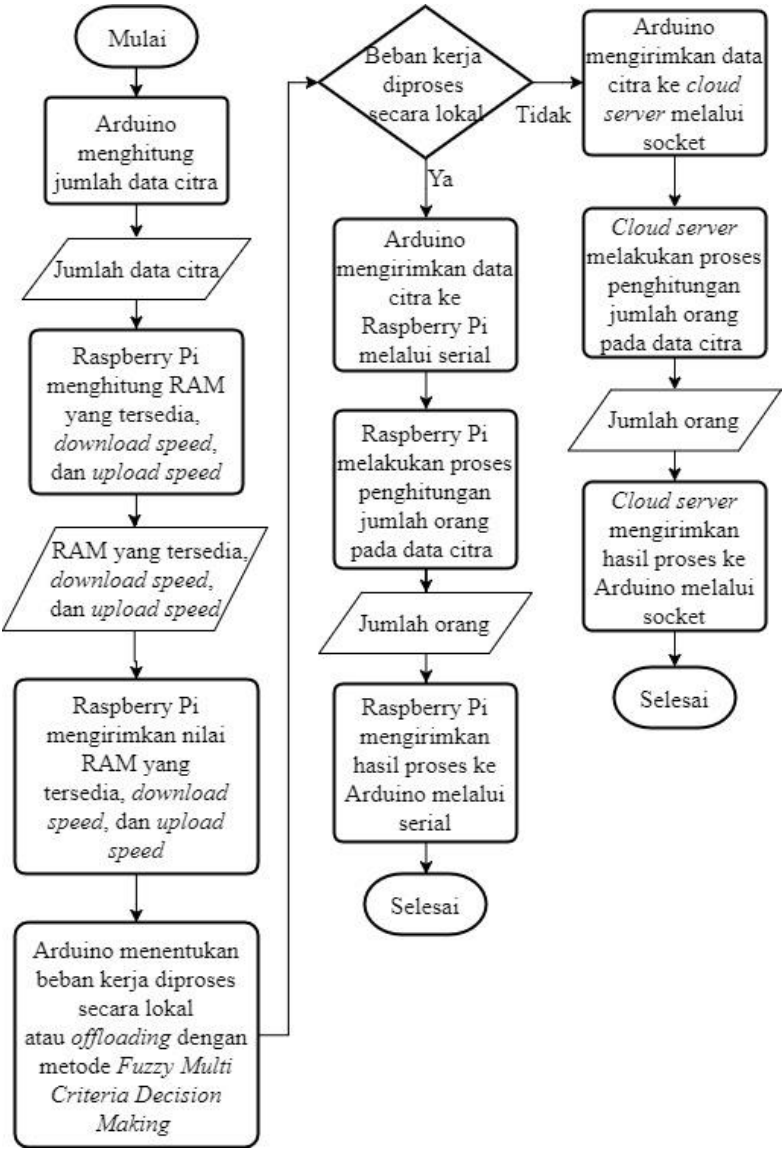
IoT menggunakan serial, sedangkan komunikasi antara perangkat IoT dan *cloud server* menggunakan socket. Arsitektur umum sistem ditunjukkan pada Gambar 3.1.



Gambar 3.1 Arsitektur umum sistem

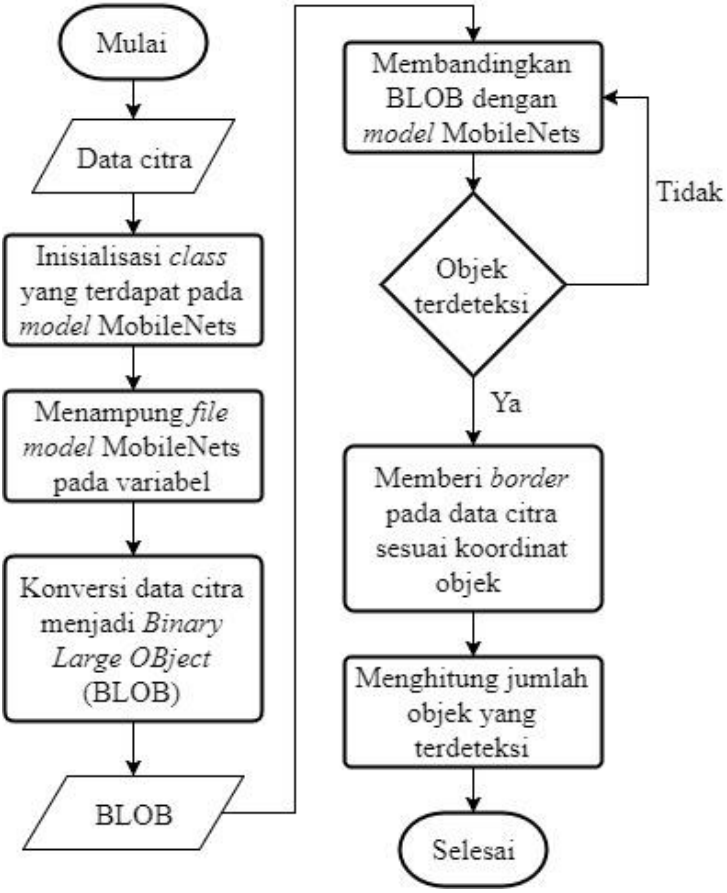
Sistem dimulai dengan mikrokontroler Arduino menghitung jumlah data citra yang akan diproses, lalu Raspberry Pi mengirim nilai RAM yang tersedia, *download speed*, dan *upload speed* ke mikrokontroler Arduino. Setelah itu, mikrokontroler Arduino menentukan apakah beban kerja akan diproses secara lokal atau secara *offloading* dengan metode *Fuzzy Multi Criteria Decision Making* yang menggunakan faktor-faktor diantaranya RAM Raspberry Pi yang tersedia, *download speed*, *upload speed*, dan jumlah data citra yang akan diproses. Jika beban kerja diproses secara lokal maka mikrokontroler Arduino akan mengirimkan data citra ke Raspberry Pi lalu Raspbery Pi melakukan proses penghitungan jumlah orang pada gambar dan mengirimkan hasilnya ke mikrokontroler Arduino melalui serial. Sedangkan jika beban kerja diproses secara *offloading* maka mikrokontroler Arduino akan mengirimkan data citra ke *cloud server* lalu *cloud*

server akan melakukan proses penghitungan jumlah orang pada gambar dan mengirimkan hasilnya ke mikrokontroler Arduino melalui socket. Alur kerja sistem ditunjukkan pada Gambar 3.2.



Gambar 3.2 Diagram alir sistem

Proses penghitungan jumlah orang pada gambar dimulai dengan mengakses *file model* MobileNets, lalu data citra dikonversi menjadi *Binary Large OBject* (BLOB). Hasil konversi dibandingkan dengan *model* MobileNets, jika objek sudah terdeteksi maka data citra akan diberi penanda sesuai dengan lokasi objek yang terdeteksi. Proses penghitungan jumlah orang ditunjukkan pada Gambar 3.3.



Gambar 3.3 Diagram alir penghitungan jumlah orang pada gambar

3.3 Data

Pada sub bab ini dijelaskan mengenai data yang digunakan sebagai masukan pada perangkat IoT untuk selanjutnya diolah secara lokal atau secara *offloading* sehingga menghasilkan data keluaran yang diharapkan dengan waktu pemrosesan seminimal mungkin.

3.3.1 Data Masukan

Data masukan adalah data yang digunakan sebagai masukan awal dari sistem. Data masukan berupa citra yang akan digunakan sebagai datatest untuk uji coba perhitungan jumlah orang.

Contoh citra sebagai data masukan datatest ditunjukkan pada Gambar 3.4.

Gambar 3.4 Contoh data masukan citra yang digunakan sebagai datatest [5]

3.3.2 Data Keluaran

Data masukan akan dilakukan *image processing* dengan menggunakan model MobileNets dan memulai proses deteksi objek dengan mengubah data masukan menjadi *Binary Large OBject* (BLOB) lalu membandingkannya dengan model yang

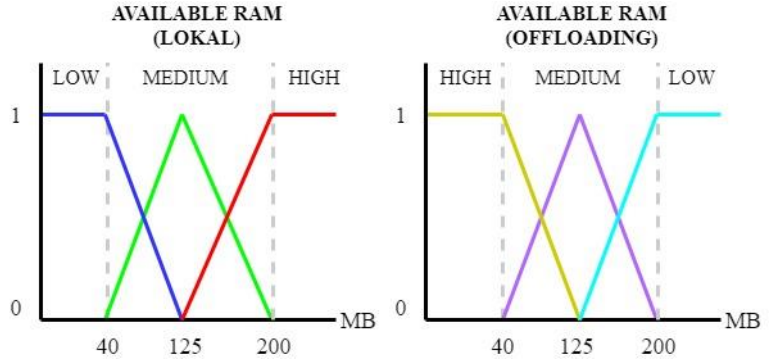
sudah dimuat sebelumnya. Hasil dari proses *image processing* adalah jumlah orang dan citra yang sudah diperbaharui dengan posisi orang yang terdeteksi. Contoh data keluaran ditunjukkan pada Gambar 3.5.

Gambar 3.5 Contoh data citra keluaran proses penghitungan jumlah orang

3.4 Faktor Penentu Metode *Offloading*

3.4.1 Kapasitas Raspberry Pi RAM yang tersedia

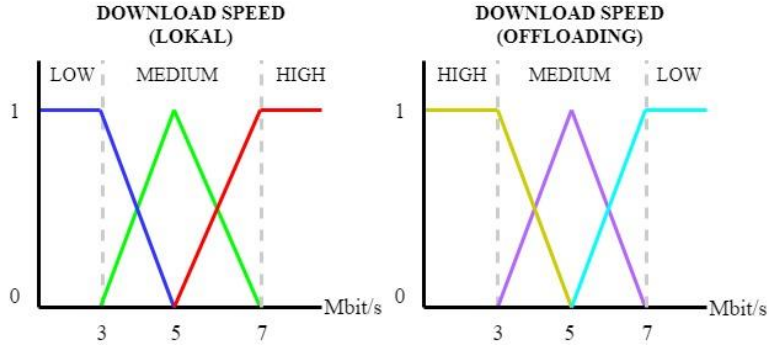
Data yang pertama adalah kapasitas Raspberry Pi RAM yang tersedia yang terbagi menjadi 3 kelas urut dari yang buruk ke baik yaitu *Low*, *Medium*, dan *High*. Data ini diperoleh dengan mengambil RAM Raspberry Pi yang tersedia melalui *bash command*. Penentuan kelas terbagi menjadi 2 macam. Penentuan kelas untuk metode lokal dan *offloading* menggunakan fungsi keanggotaan yang ditunjukkan pada Gambar 3.6.



Gambar 3.6 Fungsi keanggotaan metode lokal dan *offloading* kapasitas Raspberry Pi RAM yang tersedia

3.4.2 *Download Speed*

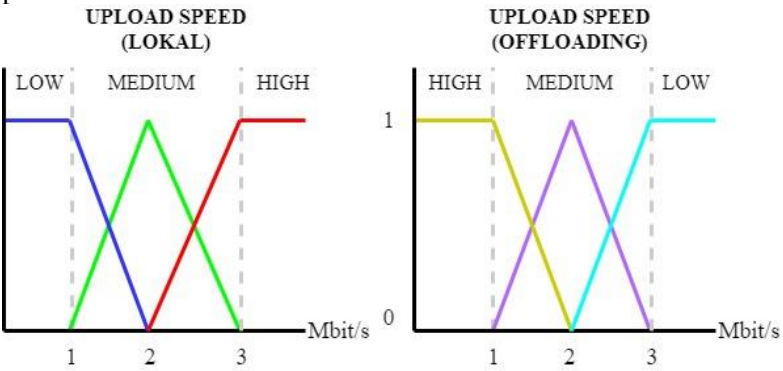
Data yang kedua adalah *download speed* dari Perangkat IoT yang terbagi menjadi 3 kelas urut dari yang buruk ke baik yaitu *Low*, *Medium*, dan *High*. Data ini diperoleh dengan *library speedtest* agar kecepatan *download* diperoleh. Penentuan kelas terbagi menjadi 2 macam. Penentuan kelas untuk metode lokal dan *offloading* menggunakan fungsi keanggotaan yang ditunjukkan pada Gambar 3.7.



Gambar 3.7 Fungsi keanggotaan metode lokal dan *offloading download speed*

3.4.3 *Upload Speed*

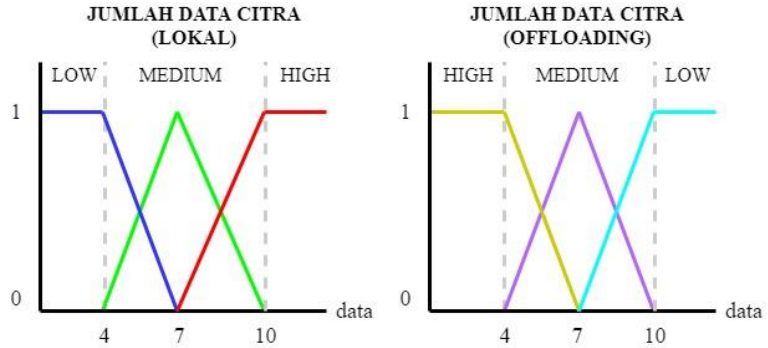
Data yang ketiga adalah *upload speed* dari Perangkat IoT yang terbagi menjadi 3 kelas urut dari yang buruk ke baik yaitu *Low*, *Medium*, dan *High*. Data ini diperoleh dengan *library speedtest* agar kecepatan *upload* diperoleh. Penentuan kelas terbagi menjadi 2 macam. Penentuan kelas untuk metode lokal dan *offloading* menggunakan fungsi keanggotaan yang ditunjukkan pada Gambar 3.8.



Gambar 3.8 Fungsi keanggotaan metode lokal dan *offloading* *upload speed*

3.4.4 Jumlah Data Citra yang diproses

Data yang keempat adalah jumlah data citra yang diproses yang terbagi menjadi 3 kelas urut dari yang buruk ke baik yaitu *Low*, *Medium*, dan *High*. Penentuan kelas terbagi menjadi 2 macam. Penentuan kelas untuk metode lokal menggunakan fungsi keanggotaan yang ditunjukkan pada Gambar 3.9.



Gambar 3.9 Fungsi keanggotaan metode lokal dan *offloading* jumlah data citra yang diproses

3.5 *Offloading* dengan Metode *Fuzzy Multi Criteria Decision Making*

Offloading merupakan salah satu metode untuk peningkatan kinerja proses komputasi pada perangkat dengan cara melakukan *offload* beban kerja dari perangkat ke *cloud server*. Metode ini sangat mendukung untuk diimplementasikan sebab konektivitas internet yang bertindak sebagai *virtual bus* dapat diperoleh atau didapatkan dimana saja dan kapan saja saat ini. Studi kasus pada tugas akhir ini, *cloud server* tidak dibatasi oleh sumber daya baterai dan memiliki *processor* dengan performa lebih cepat dibandingkan dengan perangkat IoT sebagai *client* sehingga apabila melakukan *offloading* beban kerja ke *cloud server*, diharapkan eksekusi operasi terhadap beban kerja lebih cepat dan meminimalisir waktu eksekusi.

Pada tugas akhir ini, beban kerja yang akan di *offload* ke *cloud server* sudah ditentukan yaitu *image processing* dengan studi kasus penghitungan jumlah orang pada gambar. Dalam memulai eksekusi *offload* beban kerja pada *cloud server*, ada beberapa hal yang harus dilakukan sebelumnya agar hasil dari eksekusi *offload* beban kerja ke *cloud server* maksimal dan sesuai yang diharapkan. Identifikasi beban kerja, kondisi Perangkat IoT *client* dan

pemantauan kualitas jaringan internet sangat penting dalam metode ini. Kondisi Perangkat IoT seperti kapasitas RAM yang tersedia, *download speed, upload speed*, dan jumlah data citra yang akan diproses. Selanjutnya, hal – hal yang dapat mempengaruhi pengambilan keputusan metode eksekusi kita sebut sebagai faktor – faktor penentu metode *offloading* dan fungsi yang digunakan untuk menentukan keputusan metode eksekusi disebut dengan *decision maker*.

Decision maker adalah fungsi yang dipanggil setiap beban kerja yang telah ditentukan pada Perangkat IoT akan dieksekusi. Data faktor – faktor penentu metode *offloading* akan digunakan sebagai catatan histori performa oleh *decision maker* untuk memperkirakan dan membandingkan biaya dari kedua metode eksekusi tersebut sebelum memutuskan metode eksekusi yang akan dipilih. Pada perancangan tugas akhir ini, *decision maker* melakukan pengecekan data terhadap kapasitas RAM yang tersedia pada Perangkat IoT, *download speed, upload speed,* dan jumlah data citra yang akan diproses.

Decision maker pada tugas akhir ini menggunakan metode *Fuzzy Multi Criteria Decision Making*. Langkah-langkah *Fuzzy Multi Criteria Decision Making* pada tugas akhir ini adalah sebagai berikut :

1. Inisialisasi *fuzzy number* dari masing-masing variabel linguistik (*Low*, *Medium*, High)
2. Penentuan variabel linguistik (*Low, Medium, High*) dari masing-masing faktor penentu metode *offloading* (kapasitas RAM yang tersedia, *download speed*, *upload speed*, dan jumlah data citra yang diproses). Pada tugas akhir ini, menggunakan lima *expert* untuk tiap faktor penentu.
3. Menghitung rata-rata *fuzzy number* tiap faktor penentu metode *offloading* sesuai dengan jumlah *expert*.
4. Menghitung nilai defuzzyfikasi dari tiap faktor penentu metode *offloading*. Nilai defuzzyfikasi didapatkan dengan membagi total dari rata-rata *fuzzy number* tiap faktor penentu dengan empat.

5. Menghitung nilai *weight* dari tiap faktor dengan membagi nilai defuzzyfikasi dengan total *fuzzy number* yang ada.
6. Mengkonversi nilai-nilai faktor penentu menjadi variabel linguistik (*Low, Medium, High*) untuk metode lokal dan metode *offloading* dengan menggunakan fungsi keanggotaan dari masing-masing faktor.
7. Menghitung nilai defuzzyfikasi dari tiap faktor penentu untuk metode lokal dan metode *offloading*
8. Menghitung skor akhir dari metode lokal dan metode *offloading* yang diperoleh dari menjumlahkan hasil kali antara nilai defuzzyfikasi tiap faktor penentu untuk metode lokal dan metode *offloading* dengan nilai *weight* dari tiap faktor penentu.
9. Membandingkan skor akhir dari metode lokal dan metode *offloading*, jika skor akhir metode lokal lebih besar maka metode tersebut yang digunakan dan begitu juga sebaliknya

BAB IV
IMPLEMENTASI

Bab ini berisi penjelasan mengenai implementasi dari perancangan yang sudah dilakukan pada bab sebelumnya. Implementasi berupa *Pseudocode* untuk membangun program.

4.1 Lingkungan Implementasi

Implementasi penerapan *computation offloading* dengan metode *Fuzzy Multi Criteria Decision Making* pada sistem perangkat lunak *image processing* menggunakan spesifikasi perangkat keras dan perangkat lunak seperti yang ditunjukkan pada Tabel 4.1.

Tabel 4.1 Lingkungan Implementasi Perangkat Lunak

Privilege	Perangkat	Jenis Perangkat	Spesifikasi
Perangkat IoT (Arduino Yun)	Perangkat Keras	Prosesor	Atheros AR9331
		Memori	64 MB DDR2
	Perangkat Lunak	Sistem Operasi	OpenWrt
Perangkat IoT (Raspberry Pi 2)	Perangkat Keras	Prosesor	ARM Cortex-A7 CPU
		Memori	1 GB
		Koneksi	USB wifi dongle

	Perangkat Lunak	Sistem Operasi	Raspbian OS
Cloud Server	Penyedia Layanan	Digital Ocean	
	IP Address	128.199.246.173	
	Perangkat Keras	Prosesor	Intel(R) Xeon(R) CPU E5-2630 @ 2.30GHz
		Memori	512 MB
	Perangkat Lunak	Sistem Operasi	Ubuntu 16.04
	Perangkat Lunak	Perangkat Pengembang	Arduino IDE dan Visual Studio Code sebagai *text editor*, dan Net Balancer sebagai pengatur kecepatan internet

4.2 Implementasi

Pada sub bab implementasi ini menjelaskan mengenai pembangunan perangkat lunak secara detail dan menampilkan *Pseudocode* yang digunakan untuk pengambilan nilai-nilai faktor penentu metode *offloading*, penghitungan jumlah orang pada gambar, metode *Fuzzy Multi Criteria Decision Making* untuk menentukan apakah beban kerja diproses secara lokal atau secara *offloading*, dan komunikasi antara Perangkat IoT dan *cloud server*.

4.2.1 Komunikasi antar Perangkat IoT *via* Serial

Sistem terdiri dari Perangkat IoT dan *cloud server*, dimana Perangkat IoT terdiri dari mikrokontroler Arduino dan Raspberry Pi. Mikrokontroler Arduino dan Raspberry Pi tersambung menggunakan kabel USB. Kedua perangkat saling berkomunikasi via serial. Pada mikrokontroler Arduino, serial diinisialisasi pada fungsi `setup` dengan *baudrate* 9600. Fungsi `setup` ditunjukkan pada *Pseudocode* 4.1.

```
1  function setup()
2      Begin Serial <- 9600
3      Begin Bridge
4      Begin Console
5      while not Console
6      print "[INFO] Waiting program from Raspberry"
```

***Pseudocode* 4.1. Kode sumber inisialisasi Serial pada Mikrokontroler Arduino**

Sedangkan pada perangkat Raspberry Pi, serial diinisalisasi pada fungsi `main` dengan *baudrate* 9600. Fungsi `main` ditunjukkan pada *Pseudocode* 4.2.

```
1  function main()
2       print "[INFO] Initialize serial with bitrate
                9600"
3       ser = serial.Serial(port='/dev/ttyACM0',
                baudrate=9600, parity=PARITY_NONE,
                stopbits=STOPBITS_ONE,
                bytesize=EIGHTBITS, timeout=1)
```

***Pseudocode* 4.2 Kode sumber inisialisasi Serial pada Raspberry Pi**

4.2.2 Faktor Penentu Metode *Offloading*

Pada awal sistem berjalan, setelah inisialisasi serial antar Perangkat IoT maka sistem akan mengambil nilai dari faktor penentu metode *offloading*. Faktor penentu metode *offloading*

terdiri dari 4 macam diantaranya kapasitas RAM Raspberry Pi yang tersedia, *download speed*, *upload speed*, dan jumlah data citra yang diproses. Tiap nilai faktor metode *offloading* diperoleh pada perangkat mikrokontroler Arduino dan Raspberry Pi. Mikrokontroler Arduino mengambil nilai jumlah data citra yang diproses menggunakan fungsi `getTotalData`. Fungsi `getTotalData` pada *Pseudocode* 4.3.

```
1  function getTotalData()
2       process.runShellCommand("python
              /mnt/sda1/arduino/www/tugasakhir/coun
3             tfile.py")
        while process is available:
           c = read process
6          result += c
7       return result
```

***Pseudocode* 4.3 Kode sumber untuk mendapatkan jumlah data citra yang diproses**

Sedangkan Raspberry Pi untuk mendapatkan kapasitas RAM yang tersedia menggunakan *bash command* "/proc/meminfo". *Bash command* dijalankan melalui bahasa Python pada fungsi `getFreeRAM`. Satuan keluaran dari fungsi `getFreeRAM` adalah *Kilo Bytes*. Fungsi `getFreeRAM` ditunjukkan pada *Pseudocode* 4.4.

```
1  function getFreeRAM()
2       command = "grep 'MemFree' /proc/meminfo | awk
                '{print $2}'"
3       result   =   subprocess.check_output(command,
                shell=True).strip()
4       return result
```

***Pseudocode* 4.4 Kode sumber untuk mendapatkan kapasitas RAM yang tersedia**

Faktor penentu metode *offloading* yang lainnya diantaranya *download speed* dan *upload speed*. *Download speed* dan *upload speed* diperoleh dengan menggunakan *library*

speedtest. *Library speedtest* diterapkan pada fungsi `getInternetSpeed` yang ditunjukkan pada *Pseudocode* 4.5 dengan parameter *category* yang digunakan sebagai pembeda *download speed* dan *upload speed*.

```
1   function getInternetSpeed(category)
2       command = "speedtest --simple --no-pre-
                allocate | grep 'Download\|Upload' |
3               awk '{print $2}'"
        result  = subprocess.check_output(command,
4               shell=True).strip().splitlines()
        return result[category]
```

Pseudocode* 4.5 Kode sumber untuk mendapatkan *download speed* dan *upload speed

4.2.3 Perhitungan *Fuzzy Multi Criteria Decision Making*

Perhitungan *Fuzzy Multi Criteria Decision* dilakukan dengan langkah-langkah sebagai berikut :

1. Inisialisasi *fuzzy number* dari masing-masing variabel linguistik (*Low, Medium, High*). Inisialisasi ditunjukkan pada Tabel 4.2.

Tabel 4.2 Inisialisasi *Fuzzy Number*

Variabel Linguistik	***Fuzzy Number***
Low (L)	(0.0, 0.3, 0.3, 0.5)
Medium (M)	(0.3, 0.5, 0.5, 0.7)
High (H)	(0.5, 0.7, 0.7, 1.0)

2. Penentuan variabel lingustik dari *expert* masing-masing faktor penentu metode offloading. Pada tugas akhir ini, terdapat lima *expert* penentu untuk masing-masing faktor penentu yang ditunjukkan pada Tabel 4.3.

Tabel 4.3 Penentuan variabel linguistik oleh *expert* pada faktor penentu metode *offloading*

Faktor Penentu	*Expert*				
	E1	E2	E3	E4	E5
Available RAM	M	M	L	M	L
Download Speed	H	M	H	H	M
Upload Speed	M	H	H	M	H
Jumlah Data Citra	H	M	H	M	M

Penentuan variabel linguistik oleh *expert* pada faktor penentu metode *offloading* ditentukan oleh tingkat pentingnya faktor penentu tersebut terhadap metode *offloading*. Keuntungan dari teori *fuzzy* ini ditentukan dari penentuan variabel linguistik oleh *expert* pada faktor penentu metode *offloading*.

3. Mengkonversi nilai variabel linguistik pada faktor penentu menjadi *fuzzy number*. Hasil konversi berupa *fuzzy decision matrix* yang ditunjukkan pada Gambar 4.1.

$$\begin{bmatrix} (0.3,0.5,0.5,0.7) & (0.3,0.5,0.5,0.7) & (0.0,0.3,0.3,0.5) & (0.3,0.5,0.5,0.7) & (0.0,0.3,0.3,0.5) \\ (0.5,0.7,0.7,1.0) & (0.3,0.5,0.5,0.7) & (0.5,0.7,0.7,1.0) & (0.5,0.7,0.7,1.0) & (0.3,0.5,0.5,0.7) \\ (0.3,0.5,0.5,0.7) & (0.5,0.7,0.7,1.0) & (0.5,0.7,0.7,1.0) & (0.3,0.5,0.5,0.7) & (0.5,0.7,0.7,1.0) \\ (0.5,0.7,0.7,1.0) & (0.3,0.5,0.5,0.7) & (0.5,0.7,0.7,1.0) & (0.3,0.5,0.5,0.7) & (0.3,0.5,0.5,0.7) \end{bmatrix}$$

Gambar 4.1 *Fuzzy Decision Matrix*

4. Menghitung nilai rata-rata tiap *fuzzy number* dari *fuzzy decision matrix*. Sebagai contoh, rata-rata *fuzzy number* 1 pada faktor *Available* RAM diperoleh dengan cara $\frac{0.3+0.3+0.0+0.3+0.0}{5} = 0.18$ dan berlaku juga untuk rata-rata *fuzzy number* yang lainnya. Selanjutnya menghitung nilai defuzzyfikasi dari tiap faktor penentu dengan rumus $\frac{avg1+avg2+avg3+avg4}{4}$, sebagai contoh nilai defuzzyfikasi

pada faktor *Available* RAM diperoleh dengan cara $\frac{0.18+0.42+0.42+0.62}{4}$ = 0.41. Lalu menghitung *weight* dengan rumus $\frac{\text{nilai defuzzyfikasi}}{\text{total criteria}*\text{total fuzzy number}}$, sebagai contoh nilai *weight* pada faktor *Available* RAM diperoleh dengan cara $\frac{0.41}{4*4}$ = 0.0256. Hasil lebih lengkapnya ditunjukkan pada Tabel 4.4.

Tabel 4.4 Hasil Normalisasi *Weight*

Faktor	Rata-rata *Fuzzy Number*				Defuzzyfikasi	*Weight*
Available RAM	0.18	0.42	0.42	0.62	0.41	0.0256
Download Speed	0.42	0.62	0.62	0.88	0.635	0.0397
Upload Speed	0.42	0.62	0.62	0.88	0.635	0.0397
Jumlah Data Citra	0.38	0.58	0.58	0.82	0.59	0.0369

5. Konversi nilai faktor penentu metode *offloading* menjadi variabel linguistik dengan fungsi keanggotaan pada Sub Bab 3.4. Sebagai contoh, nilai *Available* RAM yaitu 150 MB, *download speed* yaitu 4.2 Mbit/s, *upload speed* yaitu 3.4 Mbit/s, dan jumlah data citra yaitu 1. Untuk metode lokal, nilai *Available* RAM dikonversi dengan rumus *high* = $\frac{\text{nilai}-125}{200-125}$ sedangkan untuk *medium* = $1-(\frac{\text{nilai}-125}{200-125})$. Maka diperoleh nilai *high* = $\frac{150-125}{200-125}$ = 0.3, sedangkan *medium* = $1-(\frac{150-125}{200-125})$ = 0.7. Jika dibandingkan maka nilai *medium* lebih besar dari *high* sehingga variabel linguistik metode lokal menjadi *medium*. Sedangkan metode *offloading*, nilai *Available* RAM dikonversi dengan rumus *low* = $\frac{\text{nilai}-125}{200-125}$ sedangkan untuk *medium* = $1-(\frac{\text{nilai}-125}{200-125})$. Maka diperoleh nilai *high* = $\frac{150-125}{200-125}$ = 0.3,

sedangkan *medium* = $1 - (\frac{150-125}{200-125})$ = 0.7. Jika dibandingkan maka nilai *medium* lebih besar dari *low* sehingga variabel linguistik metode *offloading* menjadi *medium*.

Tabel 4.5 Hasil Konversi Nilai Faktor ke Variabel Linguistik

Faktor	Metode	Variabel Linguistik
Available RAM	Lokal	*Medium*
	Offloading	*Medium*
Download Speed	Lokal	*High*
	Offloading	*Low*
Upload Speed	Lokal	*Low*
	Offloading	*High*
Jumlah Data Citra	Lokal	*Low*
	Offloading	*High*

6. Pengubahan variabel linguistik menjadi *fuzzy number* lalu menghitung nilai defuzzyfikasi dari tiap faktor penentu untuk metode lokal dan metode *offloading* yang ditunjukkan pada Tabel 4.6.

Tabel 4.6 Hasil Defuzzyfikasi Nilai Faktor Penentu Metode *Offloading*

Faktor	Metode	Variabel Linguistik	*Fuzzy Number*	Defuzzyfikasi
Available RAM	Lokal	*Medium*	(0.3, 0.5, 0.5, 0.7)	0.5
	Offloading	*Medium*	(0.3, 0.5, 0.5, 0.7)	0.5
Download Speed	Lokal	*High*	(0.5, 0.7, 0.7, 1.0)	0.725

	Offloading	*Low*	(0.0, 0.3, 0.3, 0.5)	0.275
Upload Speed	Lokal	*Low*	(0.0, 0.3, 0.3, 0.5)	0.275
	Offloading	*High*	(0.5, 0.7, 0.7, 1.0)	0.725
Jumlah Data Citra	Lokal	*Low*	(0.0, 0.3, 0.3, 0.5)	0.275
	Offloading	*High*	(0.5, 0.7, 0.7, 1.0)	0.725

7. Menghitung skor akhir dengan rumus TS = $[X_{ij}]$ $[W_j]$. Skor akhir metode lokal diperoleh dengan perhitungan (0.5 x 0.0256) + (0.725 x 0.0397) + (0.275 x 0.0397) + (0.275 x 0.0369) = 0.063 sedangkan skor akhir metode *offloading* diperoleh dengan perhitungan (0.5 x 0.256) + (0.275 x 0.397) + (0.725 x 0.397) + (0.725 x 0.369) = 0.079. Bandingkan skor akhir metode lokal dan metode *offloading*, sesuai dengan hasil perhitungan diatas maka metode *offloading* memiliki nilai lebih besar dari metode lokal sehingga akan dilakukan metode *offloading*.

4.2.4 *Fuzzy Multi Criteria Decision Making Code*

Fuzzy Multi Criteria Decision Making pada tugas akhir ini digunakan sebagai *decision maker* dalam menentukan apakah beban kerja akan diproses secara lokal atau secara *offloading*. *Fuzzy Multi Criteria Decision Making* menggunakan nilai faktor penentu metode *offloading* yang sudah didapatkan sebelumnya untuk menentukan beban kerja diproses secara lokal atau secara

offloading. Langkah pertama yaitu inisialisasi *fuzzy number* dari masing-masing variabel linguistik (*Low*, *Medium*, dan *High*) pada fungsi `getFuzzyNumber` yang ditunjukkan pada *Pseudocode* 4.6.

```
1   function getFuzzyNumber(category, position)
2       fnLow[] = {0.0, 0.3, 0.3, 0.5}
        fnMedium[] = {0.3, 0.5, 0.5, 0.7}
3       fnHigh[] = {0.5, 0.7, 0.7, 1.0}

4       if category = "L":
5          return fnLow[position]
6       else if category = "M":
7          return fnMedium[position]
8       else if category = "H":
9          return fnHigh[position]
```

Pseudocode 4.6 Kode sumber untuk inisialisasi *fuzzy number*

Langkah kedua yaitu penentuan bobot dari masing-masing faktor penentu metode *offloading*, pada tugas akhir ini pemberi bobot terdiri dari lima data. Langkah ketiga yaitu menghitung rata-rata *fuzzy number* tiap faktor penentu metode *offloading* sesuai dengan jumlah pemberi bobot. Langkah ini diterapkan pada fungsi `countAvgFuzzyScore` yang ditunjukkan pada *Pseudocode* 4.7.

```
1   function countAvgFuzzyScore(*criteria, fuzzyPos,
              totalExpert)
2       total = 0
3       for i=0; i<totalExpert; i++:
4          total += getFuzzyNumber(criteria[i],
                     fuzzyPos)
5       return total/totalExpert
```

Pseudocode 4.7 Kode sumber untuk menghitung rata–rata *fuzzy number*

Langkah keempat yaitu menghitung nilai defuzzyfikasi dari tiap faktor penentu metode *offloading*. Nilai defuzzyfikasi didapatkan dengan membagi total dari rata-rata *fuzzy number* tiap

faktor penentu dengan empat. Langkah ini diterapkan pada *Pseudocode* 4.8.

```
1   defuzzyC1 = (c1AvgFuzzy1+c1AvgFuzzy2+
            c1AvgFuzzy3+c1AvgFuzzy4)/totalFuzzyNumber

2   defuzzyC2 = (c2AvgFuzzy1+c2AvgFuzzy2+
            c2AvgFuzzy3+c2AvgFuzzy4)/totalFuzzyNumber

3   defuzzyC3 = (c3AvgFuzzy1+c3AvgFuzzy2+
            c3AvgFuzzy3+c3AvgFuzzy4)/totalFuzzyNumber

4   defuzzyC4 = (c4AvgFuzzy1+c4AvgFuzzy2+
            c4AvgFuzzy3+c4AvgFuzzy4)/totalFuzzyNumber
```

Pseudocode* 4.8 Kode sumber untuk menghitung nilai defuzzyfikasi dari faktor penentu metode *offloading

Langkah kelima yaitu menghitung nilai weight dari tiap faktor penentu *offloading* dengan total *fuzzy number* yang ada. Langkah keenam yaitu mengkonversi nilai-nilai faktor penentu menjadi variabel linguistik (*Low, Medium, High*) untuk metode lokal dan metode *offloading* dengan menggunakan fungsi keanggotaan dari masing-masing faktor. Langkah keenam ini diterapkan pada fungsi `convertToLinguistic` yang ditunjukkan pada *Pseudocode* 4.9.

```
1   function convertToLinguistic (criteria, value,
          type)
2       switch(criteria):
3           // Case for Raspberry RAM
4           case 1:
5               batas1 = 40.0
6               batas2 = 125.0
7               batas3 = 200.0
8           break
9           // Case for Download Speed
10          case 2:
11              batas1 = 3.0
12              batas2 = 5.0
13              batas3 = 7.0
14          break
15          // Case for Upload Speed
16          case 3:
17              batas1 = 1.0
18              batas2 = 2.0
19              batas3 = 3.0
20          break
21          // Case for Total File
22          case 4:
23              batas1 = 4
24              batas2 = 7
25              batas3 = 10
26          break

27      if value <= batas1:
28        if criteria == 2 || criteria == 3:
29          if type == "lokal":
30            return "H"
31          else:
32            return "L"
33        else:
34          if type == "lokal":
35            return "L"
36          else:
37            return "H"
38      else if value >= batas3:
39        if criteria == 2 || criteria == 3:
40          if type == "lokal":
41            return "L"
42          else:
```

```
43              return “H”
44            else:
45              if type == “lokal”:
46                return "H"
47              else:
48                return “M”
49          else if value == batas2:
50            return “M”
51          else if value > batas1 && value < batas2:
52            low = 1-((value-batas1)/(batas2-batas1))
53            medium = (value-batas1)/(batas2-batas1)
54            if low > medium:
55              if criteria == 2 || criteria == 3:
56                if type == “lokal”:
57                  return “H”
58                else:
59                  return “L”
60              else:
61                if type == “lokal”:
62                  return “L”
63                else:
64                  return “H”
65            else:
66              return “M”
67          else if value > batas2 && value < batas3:
68            high = (value-batas2)/(batas3-batas2)
69            medium = 1-((value-batas2)/(batas3-
70                     batas2))
71            if high > medium:
72              if criteria == 2 || criteria == 3:
73                if type == “lokal”:
74                  return “L”
75                else:
76                  return “H”
77              else:
78                if type == “lokal”:
79                  return “H”
80                else:
81                  return “L”
82            else:
83              return “M”
```

***Pseudocode* 4.9 Kode sumber untuk mengkonversi nilai faktor penentu *offloading* menjadi variabel linguistik**

Langkah ketujuh yaitu menghitung nilai defuzzyfikasi dari tiap faktor penentu untuk metode lokal dan metode *offloading*. Penerapan langkah ini ditunjukkan pada *Pseudocode* 4.10.

```
1   defuzzyC1Lokal = (getFuzzyNumber(c1Lokal,
    0)+getFuzzyNumber(c1Lokal,1)+getFuzzyNumber(c1L
    okal,2)+getFuzzyNumber(c1Lokal,3))/4

2   defuzzyC2Lokal = (getFuzzyNumber(c2Lokal,
    0)+getFuzzyNumber(c2Lokal,1)+getFuzzyNumber(c2L
    okal,2)+getFuzzyNumber(c2Lokal,3))/4

3   defuzzyC3Lokal = (getFuzzyNumber(c3Lokal,
    0)+getFuzzyNumber(c3Lokal,1)+getFuzzyNumber(c3L
    okal,2)+getFuzzyNumber(c3Lokal,3))/4

4   defuzzyC4Lokal = (getFuzzyNumber(c4Lokal,
    0)+getFuzzyNumber(c4Lokal,1)+getFuzzyNumber(c4L
    okal,2)+getFuzzyNumber(c4Lokal,3))/4

5   defuzzyC1Offload = (getFuzzyNumber(c1Offload,
    0)+getFuzzyNumber(c1Offload,1)+getFuzzyNumber(c
    1Offload,2)+getFuzzyNumber(c1Offload,3))/4

6   defuzzyC2Offload = (getFuzzyNumber(c2Offload,
    0)+getFuzzyNumber(c2Offload,1)+getFuzzyNumber(c
    2Offload,2)+getFuzzyNumber(c2Offload,3))/4

7   defuzzyC3Offload = (getFuzzyNumber(c3Offload,
    0)+getFuzzyNumber(c3Offload,1)+getFuzzyNumber(c
    3Offload,2)+getFuzzyNumber(c3Offload,3))/4

8   defuzzyC4Offload = (getFuzzyNumber(c4Offload,
    0)+getFuzzyNumber(c4Offload,1)+getFuzzyNumber(c
    4Offload,2)+getFuzzyNumber(c4Offload,3))/4
```

Pseudocode* 4.10 Kode sumber untuk menghitung nilai defuzzyfikasi untuk metode lokal dan *offloading

Langkah terakhir yaitu menghitung skor akhir dari metode lokal dan *offloading* yang diperoleh dari menjumlahkan hasil kali

antara nilai defuzzyfikasi tiap faktor penentu untuk metode lokal dan *offloading* dengan nilai *weight* dari tiap faktor penentu. Setelah itu membandingkan skor akhir dari metode lokal dan *offloading*. Langkah ini diterapkan pada *Pseudocode* 4.11.

```
1   finalScoreLokal = (defuzzyC1Lokal*weightC1)
    +(defuzzyC2Lokal*weightC2)+(defuzzyC3Lokal*weig
    htC3)+(defuzzyC4Lokal*weightC4)

2   finalScoreOffload = (defuzzyC1Offload*weightC1)
    +(defuzzyC2Offload*weightC2)+(defuzzyC3Offload*
    weightC3)+(defuzzyC4OOffload*weightC4)

3   If finalScoreLokal > finalScoreOffload:
4         return "local"
5   else:
6         return "offloading"
```

Pseudocode* 4.11 Kode sumber untuk menghitung skor akhir metode lokal dan *offloading

4.2.5 Komunikasi Perangkat IoT dan *cloud server*

Komunikasi perangkat IoT dan *cloud server* dengan menggunakan *socket*. Penerapan komunikasi via *socket* pada Perangkat IoT ditunjukkan pada *Pseudocode* 4.12.

```
1   s = socket.socket()
2   port = 6666
3   s.connect(('128.199.246.173', port))

4   function send_image():
5      print '[INFO] Sending file to server'
6      image = open file image
7      image_encode = encode image to base64
8      image_size = length of image_encode
9      # Send Filename
10     s.send(args['image'])
11     s.recv(1024)
12     # Send image length
13     s.send(str(image_size))
14     print image_size
```

```
15    s.recv(1024)
16    # Send image
17    s.send(image_encode)
18    s.recv(1024)

19  function receive_image():
20    image_size = int(s.recv(1024))
21    s.send('ACK!')
22    while image_size > 0:
23       if image_size > 1024:
24           image_data = s.recv(1024)
25       else:
26           image_data = s.recv(image_size)
27       if not image_data:
28           break
29       image_size -= len(image_data)
30       image_string += image_data
31    s.send('ACK!')
32    image_result = open image
33    image_decode = decode base64 to image
34    image_result.write(image_decode)
35    print '[INFO] Image receive succesfully'

36  function main():
37    send_image()
38    jumlah_orang = s.recv(1024)
39    s.send('ACK!')
40    receive_image()

41  main()
```

***Pseudocode* 4.12 Kode sumber implementasi komunikasi socket pada Perangkat IoT**

Sedangkan penerapan komunikasi via *socket* pada *cloud server* ditunjukkan pada *Pseudocode* 4.13.

```
s = socket.socket()
print "[INFO] Socket successfully created"
port = 6666
s.setsockopt(socket.SOL_SOCKET,
socket.SO_REUSEADDR, 1)
s.bind(('', port))
print "[INFO] Socket binded to {}".format(port)
s.listen(99)
print '[INFO] Socket is listening'

while True:
    c, addr = s.accept()
    print '[INFO] Got connection'
    # Receive Image from Client
    image_name = c.recv(1024)
    c.send('ACK!')
    image_size = int(c.recv(1024))
    print image_size
    c.send('ACK!')
    image_string = ''
    while image_size > 0:
        if image_size > 1024:
            image_data = c.recv(1024)
        else:
            image_data = c.recv(image_size)
        image_size -= len(image_data)
        image_string += image_data
    c.send('ACK!')
    image_result = open image
    image_decode = decode base64 to image
    image_result.write(image_decode)
    # END OF Receive Image from Client
```

Pseudocode* 4.13 Kode sumber implementasi komunikasi socket pada *cloud server

4.2.6 Penghitungan Jumlah Orang pada Gambar

Penghitungan jumlah orang pada gambar dimulai dengan mengakses *file model* MobileNets, lalu data citra dikonversi menjadi *Binary Large OBject* (BLOB). Hasil konversi

dibandingkan dengan *model* MobileNets, jika objek orang sudah terdeteksi maka data citra akan diberi penanda sesuai dengan lokasi objek yang terdeteksi. Objek orang yang terdeteksi akan dihitung dan dijumlahkan. Penerapan ini ditunjukkan pada *Pseudocode* 4.14.

```
1   CLASSES = ["background", "aeroplane", "bicycle",
    "bird",  "boat","bottle",  "bus",  "car",  "cat",
    "chair",  "cow",  "diningtable","dog",  "horse",
    "motorbike",       "person",       "pottedplant",
    "sheep","sofa", "train", "tvmonitor"]

2   COLORS = np.random.uniform(0,255,
              size=(len(CLASSES),3))

3   start_time = time.time()
4   net = cv2.dnn.readNetFromCaffe(args["prototxt"],
    args["model"])

5   image = cv2.imread(args["image"])
6   (h, w) = image.shape[:2]
7   blob = cv2.dnn.blobFromImage(cv2.resize(image,
    (300, 300)), 0.007843, (300, 300), 127.5)

8   net.setInput(blob)
9   detections = net.forward()

10  for i in np.arange(0, detections.shape[2]):
11      confidence = detections[0, 0, i, 2]
12      if confidence > args["confidence"]:
13           idx = int(detections[0, 0, i, 1])
14           box = detections[0, 0, i, 3:7] *
             np.array([w, h, w, h])
16           (startX, startY, endX, endY) =
             box.astype("int")

17           if CLASSES[idx] == "person":
18              cv2.rectangle(image, (startX,
                startY),(endX, endY), COLORS[idx],2)
19              y = startY - 15
20              if startY - 15 > 15 else startY + 15
```

```
21              cv2.putText(image, label, (startX,
                y), cv2.FONT_HERSHEY_SIMPLEX, 0.5,
                COLORS[idx], 2)

22  cv2.imwrite("hasil.jpg", image)
```

***Pseudocode* 4.14 Kode sumber untuk penghitungan jumlah orang pada gambar**

(Halaman ini sengaja dikosongkan)

BAB V
HASIL UJI COBA DAN EVALUASI

Bab ini berisi penjelasan mengenai skenario uji coba dan evaluasi penentuan keputusan eksekusi beban kerja pada *computation offloading*. Hasil uji coba didapatkan dari implementasi pada Bab 4 dengan skenario yang berbeda. Bab ini berisikan pembahasan mengenai lingkungan pengujian, data pengujian, dan uji kinerja.

5.1 Lingkungan Pengujian

Lingkungan pengujian pada uji coba permasalahan penentuan keputusan eksekusi beban kerja oleh sistem melalui faktor – faktor penentu metode *offloading* menggunakan spesifikasi keras dan perangkat lunak seperti yang ditunjukkan pada Tabel 5.1.

Tabel 5.1 Spesifikasi Lingkungan Pengujian

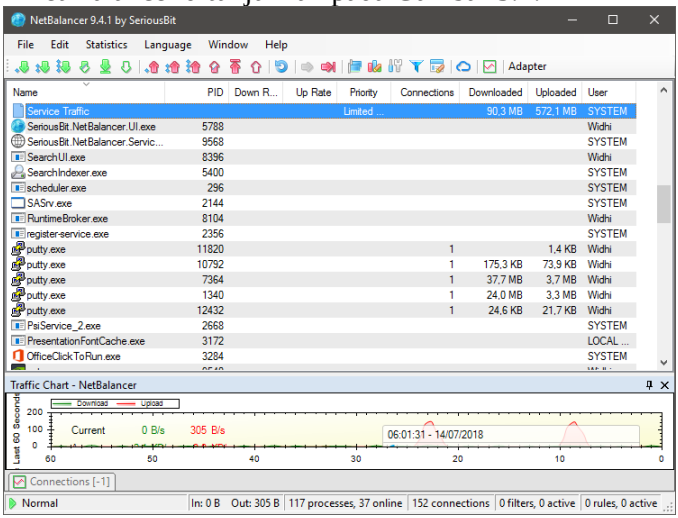
Privilege	Perangkat	Jenis Perangkat	Spesifikasi
Perangkat IoT (Arduino Yun)	Perangkat Keras	Prosesor	Atheros AR9331
		Memori	64 MB DDR2
	Perangkat Lunak	Sistem Operasi	OpenWrt
Perangkat IoT (Raspberry Pi 2)	Perangkat Keras	Prosesor	ARM Cortex-A7 CPU
		Memori	1 GB

		Koneksi	USB wifi dongle
	Perangkat Lunak	Sistem Operasi	Raspbian OS
Cloud Server	Penyedia Layanan	Digital Ocean	
	IP Address	128.199.246.173	
	Perangkat Keras	Prosesor	Intel(R) Xeon(R) CPU E5-2630 @ 2.30GHz
		Memori	512 MB
	Perangkat Lunak	Sistem Operasi	Ubuntu 16.04
	Perangkat Lunak	Perangkat Pengembang	Arduino IDE dan Visual Studio Code sebagai *text editor,* dan Net Balancer sebagai pengatur kecepatan internet

Perangkat pengujian terdiri dari Mikrokontroler Arduino, Raspberry Pi, dan *Cloud Server*. Mikrokontroler Arduino dan Raspberry Pi merupakan satu Perangkat IoT dan saling terhubung dengan kabel USB yang ditunjukkan pada Gambar 5.1.

Gambar 5.1 Perangkat Perangkat IoT

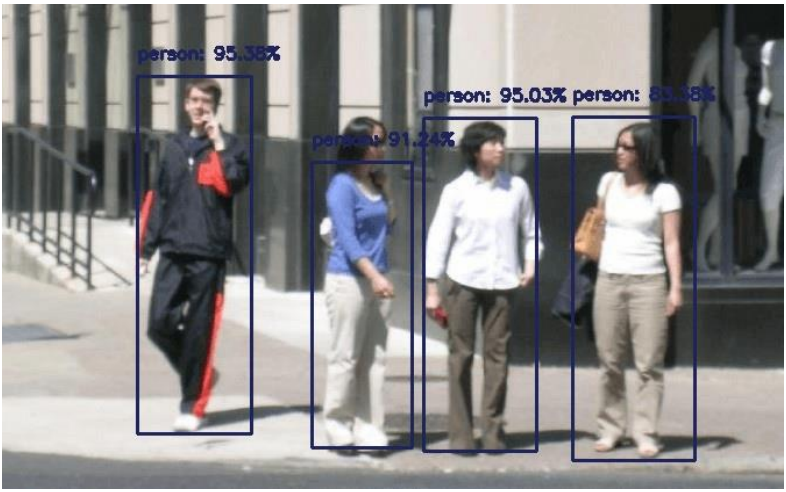
Perangkat lunak Net Balancer digunakan untuk mengatur kecepatan internet dari *access point* yang digunakan untuk uji coba. Net Balancer bekerja dengan cara membatasi kecepatan *download* atau *upload* pada *service* "Service Traffic". Tampilan perangkat lunak Net Balancer ditunjukkan pada Gambar 5.2.



Gambar 5.2 Net Balancer

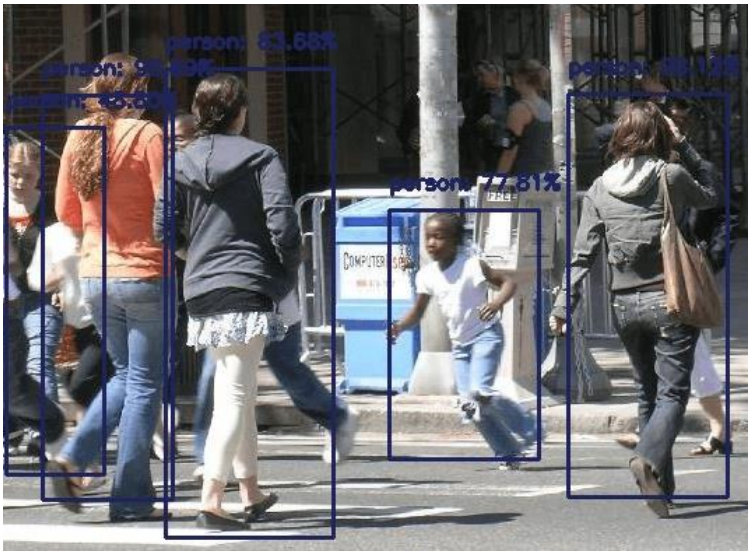
Data citra yang digunakan pada uji coba adalah dataset dari Penn-Fudan Database [5]. Berikut adalah beberapa data citra yang digunakan pada uji coba dan hasil pemrosesannya.

Gambar 5.3 Contoh data citra 1 [5]

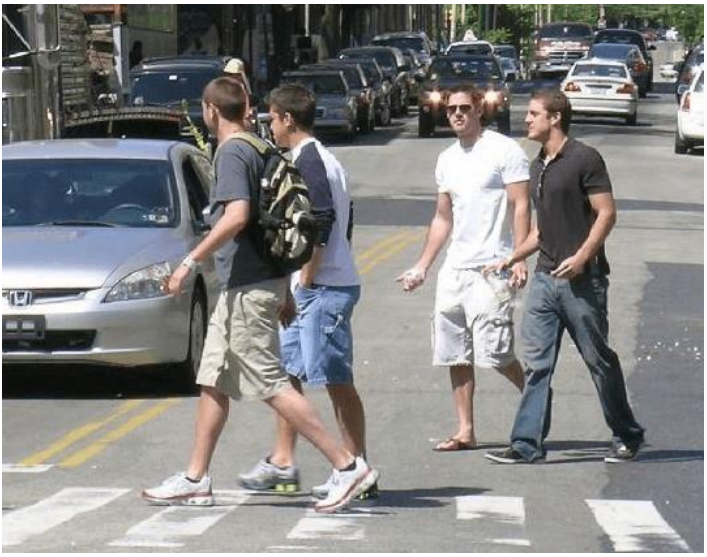


Gambar 5.4 Contoh hasil pemrosesan data citra 1

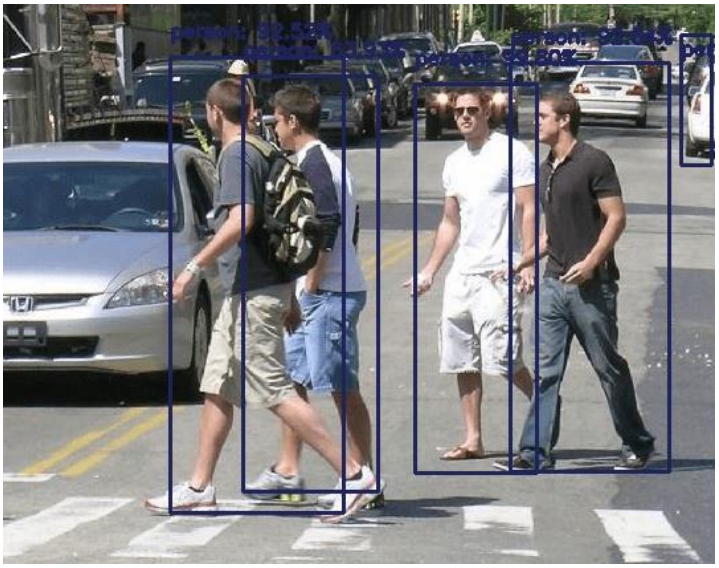
Gambar 5.5 Contoh data citra 2 [5]



Gambar 5.6 Contoh hasil pemrosesan data citra 2



Gambar 5.7 Contoh data citra 3 [5]



Gambar 5.8 Contoh hasil pemrosesan data citra 3

Gambar 5.9 Contoh data citra 4

Gambar 5.10 Contoh hasil pemrosesan data citra 5

Gambar 5.11 Contoh data citra 5

Gambar 5.12 Contoh hasil pemrosesan data citra 5

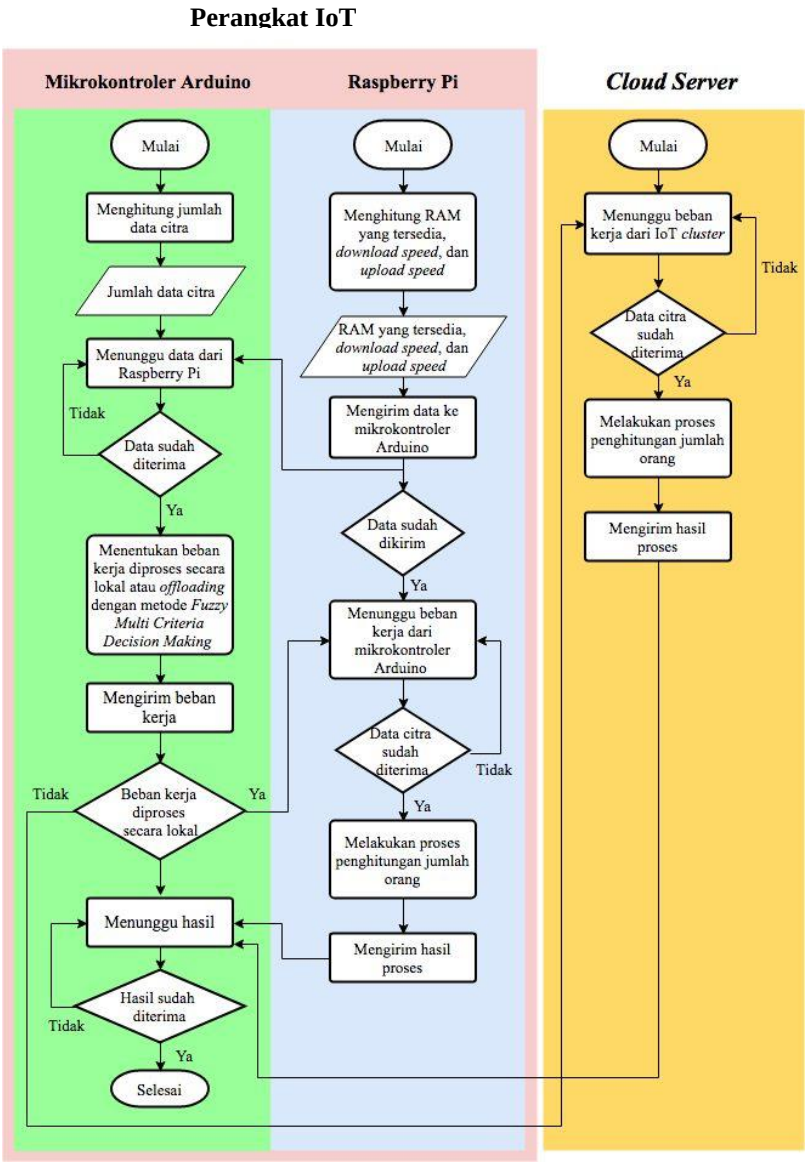
5.2 Skenario Uji Coba

Sebelum melakukan uji coba, perlu ditentukan skenario yang akan digunakan dalam uji coba. Melalui skenario ini, perangkat akan diuji apakah sudah berjalan dengan benar dan bagaimana performa pada masing-masing skenario. Dan membandingkan skenario manakah yang memiliki hasil lebih baik.

Di dalam skenario uji coba, terdapat kondisi faktor yaitu *Available* RAM pada Raspberry Pi, *download speed*, *upload speed*, dan jumlah data citra yang diproses dalam menentukan metode pengeksekusian beban kerja. Kondisi faktor tersebut akan dikategorikan ke dalam syarat tertentu untuk keperluan uji coba.

Masing-masing kondisi faktor dikategorikan menjadi tiga kategori yaitu *Low*, *Medium*, dan *High*. Pembagian kategori tiap kondisi faktor disesuaikan dengan fungsi keanggotaan pada Sub Bab 3.4. pada saat perangkat *client* melakukan proses *image processing*.

Dalam menjalankan eksekusi beban kerja proses *image processing* akan melakukan urutan langkah – langkah kerja yang sudah dijelaskan pada Sub Bab 3.3. Pada Gambar 5.13 akan dijelaskan bagaimana langkah – langkah alur kerja skenario uji coba sistem.



Gambar 5.13 Bagan alur kerja skenario uji coba

Terdapat 3 macam skenario uji coba, yaitu:

1. Perhitungan waktu eksekusi yang dihasilkan oleh eksekusi beban kerja proses penghitungan jumlah orang pada gambar oleh *computation offloading* dengan jumlah 4 gambar sebanyak 15 kali pengujian.
2. Perhitungan waktu eksekusi yang dihasilkan oleh eksekusi beban kerja proses penghitungan jumlah orang pada gambar oleh *computation offloading* dengan jumlah 7 gambar sebanyak 15 kali pengujian.
3. Perhitungan waktu eksekusi yang dihasilkan oleh eksekusi beban kerja proses penghitungan jumlah orang pada gambar oleh *computation offloading* dengan jumlah 10 gambar sebanyak 15 kali pengujian.

5.2.1 Skenario Uji Coba 1

Skenario uji coba 1 adalah pengujian yang dilakukan sebanyak 15 kali dengan kondisi *download speed* (3 *High*, 4 *Medium*, 8 *Low*), kondisi *upload speed* (5 *High*, 3 *Medium*, 7 *Low*), dan jumlah gambar dari masing-masing pengujian sebanyak 4. Perhitungan waktu eksekusi yang dihasilkan oleh eksekusi beban kerja proses penghitungan jumlah orang pada gambar dengan metode lokal dan *offloading* ditunjukkan pada Tabel 5.2.

Tabel 5.2 Waktu eksekusi beban kerja dengan metode lokal dan *offloading* pada Skenario Uji Coba 1

Pengujian	Available RAM	Download Speed	Upload Speed	Waktu Eksekusi	
	MB	(Mbit/s)	(Mbit/s)	Lokal (s)	Offloading (s)
1	169.996	16.8 (*High*)	3.71 (*High*)	89.783	68.38
2	169.748	0.14 (*Low*)	3.02 (*High*)	90.116	96.851
3	169.5	0.15 (*Low*)	0.23 (*Low*)	89.7	113.621

4	54.744	0.14 (*Low*)	0.22 (*Low*)	90.352	109.602
5	45.496	5.52 (*Medium*)	2.88 (*High*)	87.335	69.404
6	224.836	0.06 (*Low*)	0.24 (*Low*)	86.865	108.649
7	223.844	0.16 (*Low*)	0.23 (*Low*)	86.988	108.49
8	223.72	4.76 (*Medium*)	0.27 (*Low*)	85.887	81.562
9	52.884	6.1 (*High*)	2.06 (*Medium*)	87.541	67.68
10	152.456	12.37 (*High*)	3.43 (*High*)	88.396	68.407
11	283.82	0.12 (*Low*)	1.84 (*Medium*)	85.452	95.764
12	283.656	2.68 (*Low*)	0.32 (*Low*)	85.898	90.86
13	75.792	4.27 (*Medium*)	2.26 (*Medium*)	90.238	69.285
14	260.336	4.39 (*Medium*)	3.13 (*High*)	86.87	68.467
15	256.368	0.15 (*Low*)	1.48 (*Low*)	87.174	97.6
Jumlah				1318.595	1314.622

Sedangkan hasil waktu eksekusi dengan *decision making* pada uji coba 1 ditunjukkan pada Tabel 5.3.

Tabel 5.3 Waktu eksekusi beban kerja dengan *decision making* pada Skenario Uji Coba 1

Pengujian	*Fuzzy Multi Criteria Decision Making* (FMCDM) Skor		*Decision Making*	Waktu Eksekusi (s)
	Lokal	Offloading		
1	0.051	0.091	Offloading	68.38
2	0.068	0.073	Offloading	96.851
3	0.086	0.056	Lokal	89.7
4	0.075	0.067	Lokal	90.352
5	0.048	0.094	Offloading	69.404

6	0.086	0.061	Lokal	86.865
7	0.086	0.061	Lokal	86.988
8	0.071	0.076	Offloading	81.562
9	0.048	0.094	Offloading	67.68
10	0.045	0.097	Offloading	68.407
11	0.078	0.071	Lokal	85.452
12	0.086	0.061	Lokal	85.898
13	0.057	0.085	Offloading	69.285
14	0.059	0.088	Offloading	68.467
15	0.086	0.061	Lokal	87.174
Jumlah				1202.47

Berdasarkan hasil waktu eksekusi yang ditunjukkan pada Tabel 5.2 dan Tabel 5.3, diperoleh jumlah waktu eksekusi dengan metode lokal sebesar 1318.595 *seconds*, metode *offloading* sebesar 1314.622 *seconds*, dan dengan *decision making* sebesar 1202.47 *seconds*. Hal ini menunjukkan hasil waktu eksekusi dengan *decision making* 116.125 *seconds* lebih cepat dari metode lokal dan 112.152 *seconds* lebih cepat dari metode *offloading*.

Gambar 5.14 Grafik waktu eksekusi beban kerja Skenario Uji Coba 1

5.2.2 Skenario Uji Coba 2

Skenario uji coba 2 adalah pengujian yang dilakukan sebanyak 15 kali dengan kondisi *download speed* (4 *High*, 2 *Medium*, 9 *Low*), kondisi *upload speed* (6 *High*, 5 *Medium*, 4 *Low*), dan jumlah gambar dari masing-masing pengujian sebanyak 7. Perhitungan waktu eksekusi yang dihasilkan oleh eksekusi beban kerja proses penghitungan jumlah orang pada gambar dengan metode lokal dan *offloading* ditunjukkan pada Tabel 5.4.

Tabel 5.4 Waktu eksekusi beban kerja dengan metode lokal dan *offloading* pada Skenario Uji Coba 2

Pengujian	Available RAM	Download Speed	Upload Speed	Waktu Eksekusi	
	MB	(Mbit/s)	(Mbit/s)	Lokal (s)	Offloading (s)
1	180.004	15.84 (*High*)	3.72 (*High*)	146.992	117.552
2	179.128	0.15 (*Low*)	0.3 (*Low*)	146.785	176.733
3	175.396	0.14 (*Low*)	2.6 (*High*)	146.971	162.218
4	174.812	5.32 (*Medium*)	2.89 (*High*)	146.888	120.292
5	49.944	4.9 (*Medium*)	1.82 (*Medium*)	145.327	118.561
6	274.668	15.5 (*High*)	2.54 (*High*)	146.647	118.788
7	274.312	0.59 (*Low*)	0.61 (*Low*)	146.108	150.427
8	269	0.56 (*Low*)	2.84 (*Medium*)	139.838	125.471
9	268.868	0.57 (*Low*)	1.71 (*Medium*)	143.793	126.716
10	71.12	0.6 (*Low*)	1.64 (*Medium*)	146.996	127.679
11	264.256	0.35 (*Low*)	0.48 (*Low*)	143.675	156.57

12	82.208	0.14 (*Low*)	0.26 (*Low*)	144.2	178.439
13	133.296	0.14 (*Low*)	2.86 (*High*)	143.701	160.269
14	132.924	10.34 (*High*)	1.64 (*Medium*)	140.244	119.391
15	55.796	13.62 (*High*)	3.66 (*High*)	140.377	118.127
Jumlah				2168.542	2077.233

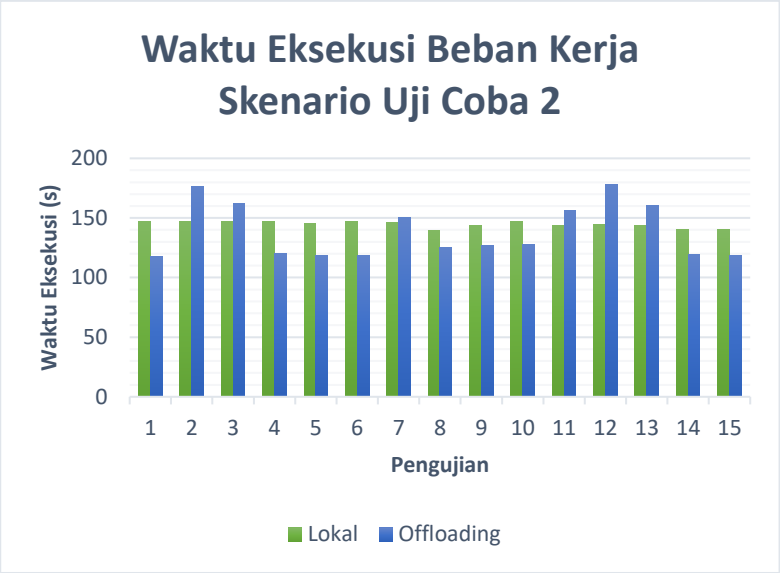
Sedangkan hasil waktu eksekusi dengan *decision making* pada uji coba 2 dapat dilihat pada Tabel 5.5.

Tabel 5.5 Waktu eksekusi beban kerja dengan *decision making* pada Skenario Uji Coba 2

Pengujian	*Fuzzy Multi Criteria Decision Making* (FMCDM) Skor		*Decision Making*	Waktu Eksekusi (s)
	Lokal	Offloading		
1	0.059	0.083	Offloading	117.552
2	0.095	0.047	Lokal	146.785
3	0.077	0.065	Lokal	146.971
4	0.068	0.074	Offloading	120.292
5	0.065	0.077	Offloading	118.561
6	0.059	0.089	Offloading	118.788
7	0.095	0.053	Lokal	146.108
8	0.077	0.071	Lokal	139.838
9	0.086	0.062	Lokal	143.793
10	0.062	0.075	Offloading	127.679
11	0.095	0.053	Lokal	143.675
12	0.083	0.059	Lokal	144.2
13	0.071	0.071	Offloading	160.269
14	0.062	0.08	Offloading	119.391
15	0.047	0.095	Offloading	118.127
Jumlah				2012.029

Berdasarkan hasil waktu eksekusi yang ditunjukkan pada Tabel 5.4 dan Tabel 5.5, diperoleh jumlah waktu eksekusi dengan metode lokal sebesar 2168.542 *seconds*, metode *offloading* sebesar

2077.233 *seconds*, dan dengan *decision making* sebesar 2012.029 *seconds*. Hal ini menunjukkan hasil waktu eksekusi dengan *decision making* 156.513 *seconds* lebih cepat dari metode lokal dan 65.204 *seconds* lebih cepat dari metode *offloading*.



Gambar 5.15 Grafik waktu eksekusi beban kerja Skenario Uji Coba 2

5.2.3 Skenario Uji Coba 3

Skenario uji coba 3 adalah pengujian yang dilakukan sebanyak 15 kali dengan kondisi *download speed* (7 *High*, 5 *Medium*, 3 *Low*), kondisi *upload speed* (7 *High*, 0 *Medium*, 8 *Low*), dan jumlah gambar dari masing-masing pengujian sebanyak 10. Perhitungan waktu eksekusi yang dihasilkan oleh eksekusi beban kerja proses penghitungan jumlah orang pada gambar dengan metode lokal dan *offloading* ditunjukkan pada Tabel 5.6.

Tabel 5.6 Waktu eksekusi beban kerja dengan metode lokal dan *offloading* pada Skenario Uji Coba 3

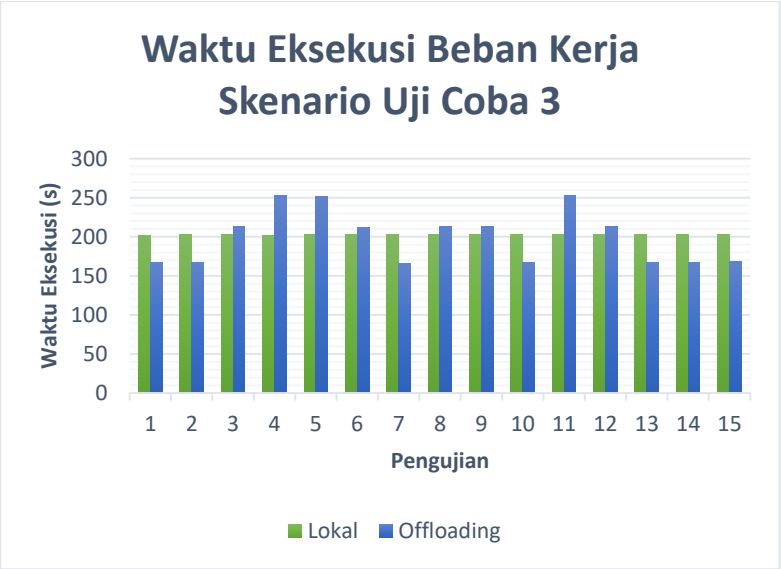
Pengujian	Available RAM	Download Speed	Upload Speed	Waktu Eksekusi	
	MB	(Mbit/s)	(Mbit/s)	Lokal (s)	Offloading (s)
1	69.108	13.94 (*High*)	3.76 (*High*)	202.155	167.37
2	272.02	16.15 (*High*)	3.76 (*High*)	202.821	167.684
3	271.896	5.75 (*Medium*)	0.27 (*Low*)	203.139	212.847
4	267.342	0.15 (*Low*)	0.35 (*Low*)	201.964	252.43
5	277.394	0.32 (*Low*)	0.43 (*Low*)	202.376	251.992
6	139.23	5.22 (*Medium*)	0.42 (*Low*)	203.542	212.428
7	57.342	15.23 (*High*)	3.24 (*High*)	203.434	166.472
8	65.772	5.53 (*Medium*)	0.59 (*Low*)	202.942	213.218
9	268.921	6.82 (*Medium*)	0.45 (*Low*)	202.877	212.924
10	127.546	14.39 (*High*)	3.65 (*High*)	202.754	167.23
11	232.22	0.14 (*Low*)	0.32 (*Low*)	202.455	252.97
12	245.17	6.23 (*Medium*)	0.3 (*Low*)	202.862	213.14
13	44.384	12.4 (*High*)	3.66 (*High*)	203.17	167.42
14	252.96	14.5 (*High*)	3.34 (*High*)	202.54	167.27
15	152.25	12.78 (*High*)	3.5 (*High*)	203.14	167.986
Jumlah				3042.171	2993.381

Sedangkan hasil waktu eksekusi dengan *decision making* pada uji coba 3 dapat dilihat pada Tabel 5.7.

Tabel 5.7 Waktu eksekusi beban kerja dengan *decision making* pada Skenario Uji Coba 3

Pengujian	*Fuzzy Multi Criteria Decision Making* (FMCDM) Skor		*Decision Making*	Waktu Eksekusi (s)
	Lokal	Offloading		
1	0.056	0.095	Offloading	167.37
2	0.067	0.089	Offloading	167.684
3	0.094	0.062	Lokal	203.139
4	0.096	0.045	Lokal	201.964
5	0.096	0.045	Lokal	202.376
6	0.094	0.062	Lokal	203.542
7	0.056	0.095	Offloading	166.472
8	0.086	0.062	Offloading	213.218
9	0.094	0.062	Lokal	202.877
10	0.071	0.077	Offloading	167.23
11	0.096	0.045	Lokal	202.455
12	0.094	0.062	Lokal	202.862
13	0.056	0.095	Offloading	167.42
14	0.067	0.089	Offloading	167.27
15	0.071	0.077	Offloading	167.986
Jumlah				2803.865

Berdasarkan hasil waktu eksekusi yang ditunjukkan pada Tabel 5.6 dan Tabel 5.7, diperoleh jumlah waktu eksekusi dengan metode lokal sebesar 3042.171 *seconds*, metode *offloading* sebesar 2993.381 *seconds*, dan dengan *decision making* sebesar 2803.865 *seconds*. Hal ini menunjukkan hasil waktu eksekusi dengan *decision making* 238.306 *seconds* lebih cepat dari metode lokal dan 189.516 *seconds* lebih cepat dari metode *offloading*.



Gambar 5.16 Grafik waktu eksekusi beban kerja Skenario Uji Coba 3

5.3 Evaluasi Umum Skenario Uji Coba

Hasil waktu eksekusi dari ketiga skenario dapat dilihat pada Tabel 5.8.

Tabel 5.8 Hasil waktu eksekusi pada skenario uji coba

Waktu Eksekusi (s)	Skenario Uji Coba		
	1	2	3
Lokal	1318.595	2168.542	3042.171
Offloading	1314.622	2077.233	2993.381
Decision Making	1202.465	2012.029	2803.865

Berdasarkan skenario uji coba 1 yang telah dilakukan, dapat diketahui waktu eksekusi dengan metode lokal sebesar 1318.595 *seconds*, metode *offloading* sebesar 1314.622 *seconds*,

dan dengan *decision making* sebesar 1202.47 *seconds*. Hal ini menunjukkan hasil waktu eksekusi dengan *decision making* 116.125 *seconds* lebih cepat dari metode lokal dan 112.152 *seconds* lebih cepat dari metode *offloading*. Sedangkan pada skenario uji coba 2 yang telah dilakukan, waktu eksekusi dengan metode lokal sebesar 2168.542 *seconds*, metode *offloading* sebesar 2077.233 *seconds*, dan dengan *decision making* sebesar 2012.029 *seconds*. Hal ini menunjukkan hasil waktu eksekusi dengan *decision making* 156.513 *seconds* lebih cepat dari metode lokal dan 65.204 *seconds* lebih cepat dari metode *offloading*. Sedangkan pada skenario uji coba 3 yang telah dilakukan, waktu eksekusi dengan metode lokal sebesar 3042.171 *seconds*, metode *offloading* sebesar 2993.381 *seconds*, dan dengan *decision making* sebesar 2803.865 *seconds*. Hal ini menunjukkan hasil waktu eksekusi dengan *decision making* 238.306 *seconds* lebih cepat dari metode lokal dan 189.516 *seconds* lebih cepat dari metode *offloading*. Dari setiap skenario uji coba, dengan *decision making* waktu eksekusi beban kerja lebih singkat dibandingkan dengan metode lokal saja dan *offloading* saja.

BAB VI
KESIMPULAN DAN SARAN

Bab ini berisikan kesimpulan yang dapat dapat diambil dari hasil uji coba yang telah dilakukan. Selain kesimpulan, terdapat juga saran yang ditujukan untuk pengembangan perangkat lunak selanjutnya.

6.1 Kesimpulan

Kesimpulan yang didapatkan berdasarkan hasil uji coba peningkatan kinerja pada Perangkat IoT menggunakan *computation offloading* yang diterapkan pada sistem adalah sebagai berikut:

1. Implementasi *computation offloading* dengan metode *Fuzzy Multi Criteria Decision Making* dapat menentukan beban kerja diproses secara lokal atau secara *offloading* , hal ini meningkatkan proses komputasi beban kerja dengan waktu eksekusi beban kerja yang lebih singkat.
2. Pada kasus Tugas Akhir ini, *computation offloading* meningkatkan kinerja proses komputasi dengan hasil waktu eksekusi 116.125 *seconds* lebih cepat dari metode lokal saja dan 112.152 *seconds* lebih cepat dari metode *offloading* saja (Uji Coba 1), 156.513 *seconds* lebih cepat dari metode lokal saja dan 65.204 *seconds* lebih cepat dari metode *offloading* saja (Uji Coba 2), 238.306 *seconds* lebih cepat dari metode lokal dan 189.516 seconds lebih cepat dari metode *offloading* saja (Uji Coba 3).
3. *Fuzzy Multi Criteria Decision Making* sebagai *Decision Maker* pada sistem ini dapat menentukan beban kerja diproses secara lokal atau secara *offloading* secara dinamis dengan memperhitungkan dari nilai faktor-faktor penentu metode *offloading*.

6.2 Saran

Saran yang diberikan terkait pengembangan pada Tugas Akhir ini adalah:

1. Menambah faktor-faktor penentu metode *offloading* pada *Fuzzy Multi Criteria Decision Making* agar penentuan pengeksekusian beban kerja lebih optimal.
2. Menggunakan lebih banyak *expert* pada pemberian variabel linguistik untuk setiap faktor penentu metode *offloading* pada *Fuzzy Multi Criteria Decision Making* agar *decision making* yang dihasilkan lebih tepat.
3. Pengecekan nilai faktor-faktor penentu metode *offloading* agar lebih akurat.

DAFTAR PUSTAKA

[1] K. K. Patel, Sunil Patel. “Internet of Things: Definition, Characteristics, Architecture, Enabling Technologies, Application & Future Challenges ” [Daring]. Tersedia pada: http://ijesc.org/upload/8e9af2eca2e1119b895544fd60c3b857.Internet%20of%20Things-IOT%20Definition,%20Characteristics,%20Architecture,%20Enabling%20Technologies,%20Application%20&%20Future%20Challenges.pdf. [Diakses: 18-February-2018].

[2] “Mikrokontroler Arduino” [Daring]. Tersedia pada: https://id.wikipedia.org/wiki/Arduino. [Diakses: 18-February-2018].

[3] Feri Djuandi. “Pengenalan Arduino” [Daring]. Tersedia pada: http://tobuku.com/docs/Arduino-Pengenalan.pdf. [Diakses: 28-February-2018].

[4] S. J. Johsnon, Simon J Cox. “The Raspberry Pi: A Technology Disrupter, and the Enabler of Dreams” [Daring]. Tersedia pada: http://www.mdpi.com/2079-9292/6/3/51/pdf. [Diakses: 05-Maret-2018].

[5] “Penn-Fudan Database for Pedestrian Detection and Segmentation” [Daring]. Tersedia pada: https://www.cis.upenn.edu/~jshi/ped_html/ [Diakses: 10-Maret-2018].

[6] “Arduino Yun” [Daring]. Tersedia pada: https://store.arduino.cc/usa/arduino-yun. [Diakses: 10-Maret-2018].

[7] “OpenWrt” [Daring]. Tersedia pada: https://en.wikipedia.org/wiki/OpenWrt. [Diakses: 12-Maret-2018].

[8] “Raspberry Pi” [Daring]. Tersedia pada: https://id.wikipedia.org/wiki/Raspberry_Pi. [Diakses: 12-Maret-2018].

[9] “Raspberry Pi 2 Model B” [Daring]. Tersedia pada: https://www.raspberrypi.org/products/raspberry-pi-2-model-b. [Diakses: 14-Maret-2018].
[10] “Raspbian” [Daring]. Tersedia pada: https://en.wikipedia.org/wiki/Raspbian. [Diakses: 20-Maret-2018].
[11] “Computation Offloading” [Daring]. Tersedia pada: https://en.wikipedia.org/wiki/Computation_offloading. [Diakses: 30-Maret-2018].
[12] Arun Nagar. “Development of Fuzzy Multi Criteria Decision Making Method for Selection of Optimum Maintenance Alternative” [Daring]. Tersedia pada: http://www.idc-online.com/technical_references/pdfs/mechanical_engineering/Development%20of%20Fuzzy.pdf. [Diakses: 30-Maret-2018].
[13] “OpenCV library.” [Daring]. Tersedia pada: http://opencv.org/. [Diakses: 31-Mei-2017].
[14] Andrew G. Howard, Menglong Zhu. “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications” [Daring]. Tersedia pada: https://arxiv.org/pdf/1704.04861/. [Diakses: 31-Mei-2017].

LAMPIRAN

(Halaman ini sengaja dikosongkan)

BIODATA PENULIS

Widhi Mahaputra Pande Putu merupakan anak dari pasangan Bapak Pande Made Raka Sumandi dan Ibu Ni Made Putri Widyawati. Lahir di Tabanan pada tanggal 17 Mei 1996. Penulis menempuh pendidikan formal dimulai dari TK Pertiwi Tabanan (2000-2001), SDN Saraswati Tabanan (2001-2007), SMPN 1 Tabanan (2007-20010), SMAN 1 Tabanan (2010-2013) dan S1 Departemen Informatika ITS (2014-2018). Bidang studi yang diambil oleh penulis pada saat berkuliah di Departemen Informatika ITS adalah Komputasi Berbasis Jaringan (KBJ). Penulis aktif dalam organisasi seperti Himpunan Mahasiswa Teknik Computer-Informatika (2015-2016) dan TPKH-ITS (2015-2016). Penulis juga aktif dalam berbagai kegiatan kepanitiaan yaitu SCHEMATICS 2015. Penulis memiliki hobi musik dan bermain game. Penulis dapat dihubungi melalui email: widhimp@gmail.com.